BOUNDED ARITHMETIC AND COMPLEXITY

EMIL JEŘÁBEK

Institute of Mathematics
Czech Academy of Sciences

DSc. Dissertation

Prague, 2021

To my wife

Contents

Chapter 0

Overview

This dissertation comprises eight published papers of the author, each constituting one chapter: [96, 98, 99, 105, 100, 101, 103, 104]. They have been lightly edited to unify the formatting, but otherwise left identical to the versions that were accepted for publication.

1 Bounded arithmetic and its origins

The central subject of this dissertation—bounded arithmetic—arose from the confluence of two seemingly disparate fields: first-order theories of arithmetic, and complexity theory (propositional proof complexity and computational complexity).

The investigation of theories of arithmetic has been one of the primary fields of research in modern logic. Arithmetical theories are interesting for both mathematical and metamathematical reasons. Integers are one of the most fundamental mathematical structures, and their study has been a core mathematical topic for centuries; it is not uncommon for problems from number theory to have simple and easy to understand statements, and yet be extremely hard to solve, which has made this subject intriguing for generations of mathematicians.

The best-known first-order theory of arithmetic is the *Peano arithmetic* (*PA*), an elegant and powerful axiomatic system conceived to capture the valid statements of elementary (first-order) number theory, i.e., true in the structure $\langle \mathbb{N}, 0, S, +, \cdot, \leq \rangle$. It can be seen as a first-order approximation to the original second-order axioms of Peano [149] that characterize the structure of positive integers uniquely up to isomorphism; while first-order Peano arithmetic no longer has this property, it has the important advantage of enjoying an effectively axiomatized proof system, which means that it can be actually employed to prove number-theoretic facts (second-order logic has no complete proof system, hence it is useless in practice for derivation of arithmetic truths—even though in theory, all true arithmetical statements semantically follow from the second-order Peano axioms).

An early impetus for logical investigation of arithmetic and related theories was given by *Hilbert's program*, which sought to provide solid foundations for abstract, infinitary mathematics ("analysis", though for us it is better to think of it as something like set theory) by showing its consistency by *finitary* proof-theoretic methods, as well as its conservativity over finitary mathematics for simple enough statements. While the extent of finitary methods was not exactly specified, it is generally assumed that what Hilbert had in mind could be formalized in *PA*, or even its weaker fragments such as primitive recursive arithmetic; this was certainly true for proof-theoretic arguments employed at around that time e.g. in Presburger's analysis of the theory of $\langle \mathbb{N}, + \rangle$.

The original conception of Hilbert's program was shattered by the seminal work of Gödel [81], who proved that effectively presented theories that can simulate a strong enough fragment of arithmetic, such as *PA*, are incomplete, and cannot even prove their own consistency, let alone the consistency of a much stronger theory. Yet shortly thereafter, Gentzen [79] discovered how to analyze *PA* proofs after all, leading to a proof of consistency of *PA* in a weak fragment of arithmetic augmented with primitive recursive transfinite induction up to the ordinal ε_0 (whereas *PA* can prove transfinite induction up to any strictly smaller ordinal). This, in a sense, pin-points the amount of infinitary reasoning lacking in *PA* that's necessary to prove the consistency of *PA*.

These results set the stage for much of later developments in the investigation of theories of arithmetic. Gödel's theorem led to the understanding that any sufficiently expressive theory comes with a rich web of statements of varying strengths independent of the theory, and that provability in such a theory as well as other associated computational problems are undecidable. But it also showed that finite strings, formulas, proofs, and other syntactic and combinatorial objects, can be adequately encoded by integers, and reasoned about in first-order theories of arithmetic; in particular, arithmetical theories can serve as meta-theories to study logical properties of themselves and of other theories. Gentzen's work led to development of proof theory of sequent calculi, which can be used to gauge the strength of theories of arithmetic by means of ordinal analysis or description of their provably total computable functions, and to prove conservation results between such theories.

While the essential incompleteness of *PA* ensures that it has many different extensions, it turns out that it also has many interesting *subtheories*. A particularly fruitful way of introducing a range of subsystems of *PA* of varying strength is to restrict the scheme of induction only to formulas from some syntactic class, typically constraining the shape of quantifier prefixes. A systematic study of such fragments was initiated by Parsons [147], who introduced the theories now called $I\Sigma_n$ and $B\Sigma_n$, and Parikh [142], who introduced the prototypical theory of *bounded arithmetic* now denoted $I\Delta_0$. In $I\Delta_0$, induction is postulated only for Δ_0 formulas, i.e., formulas (in the basic language of arithmetic) all of whose quantifiers are bounded.

As a part of his original motivation, Parikh relates $I\Delta_0$ to *computational complexity* (then a nascent field). The basic goal of computational complexity is to assess the resources (time, space, ...) required to solve various computational tasks in suitable machine models, and to classify the complexity of such tasks accordingly. Of particular importance are the class P of decision problems solvable (deterministically) in time polynomial in the length of the input, which is taken as an idealized mathematical model of problems that are efficiently solvable on a computer, and the class NP of problems whose positive instances have polynomial-time verifiable witnesses, or equivalently, problems solvable nondeterministically in polynomial time. It is generally assumed that NP-hard problems (problems to which all NP problems can be reduced) cannot be solved in polynomial time; this is the content of the famous $\mathrm{P} \neq \mathrm{NP}$

conjecture. The class NP can be generalized to the polynomial-time hierarchy PH, which is included in the class of problems solvable in polynomial space (PSPACE).

Parikh's reasons for introducing $I\Delta_0$ stems from consideration of *feasibility* (efficient computability) of definable functions and predicates. Strong theories of arithmetic such as PA or $I\Sigma_1$ prove the totality of fast-growing functions, e.g. *exponentiation*, that are infeasible. In contrast, all provably total computable functions in $I\Delta_0$ are bounded by a polynomial (i.e., a term of the language; a similar property holds for theories with bounded induction over richer languages), and evaluation of Δ_0 predicates involves only numbers polynomially bounded in the arguments, i.e., they are computable in deterministic linear space; more precisely, they comprise the linear-time hierarchy.

For various reasons, one obtains a more convenient and more robust theory by extending $I\Delta_0$ with the axiom Ω_1 (introduced in the works of Paris and Wilkie [143, 144, 145, 179]), postulating the totality of the function $\omega_1(x) = x^{\log x}$. The computational complexity reason is that while the provably total computable functions of $I\Delta_0$ are polynomially bounded in value, they are only linearly bounded in terms of the length of the input when written in binary notation (whence the connection to linear-time hierarchy); however, computational classes with polynomial resource bounds, such as P and NP, are more interesting than linear. The $\omega_1(x)$ function grows quadratically (in terms of length in binary), hence its finite iterations give arbitrarily large polynomial bounds; thus, $\Delta_0(\omega_1)$ formulas define predicates from PH, and the provably total computable functions of $I\Delta_0 + \Omega_1$ comprise $\mathrm{FP}^{\mathrm{PH}}$. The logically motivated reason is that when discussing Gödel's theorems and in other contexts involving formalization of logical syntax, we need to be able to substitute a term for a variable in a formula; since this substitution function may quadratically increase the length of the input, it is not provably total in $I\Delta_0$, but it works as expected in $I\Delta_0 + \Omega_1$.

The realization that $\mathrm{P} \neq \mathrm{NP}$ is one of the most fundamental mathematical problems owes much to the seminal paper by Cook [62], who proved the NP-completeness of the propositional satisfiability problem. It is no coincidence that Cook also became one of the founders of propositional proof complexity (Cook and Reckhow [69]), and in a development independent of the work of Wilkie and Paris, he pioneered a form of bounded arithmetic and its connection to propositional proof complexity in Cook [63].

Propositional proof complexity studies proof systems for (usually) classical propositional logic, such as the resolution system, Frege (also known elsewhere as Hilbert) systems, or sequent calculi; in general, a propositional proof system is any sound and complete scheme for certification of propositional tautologies by witnesses ("proofs") that are verifiable in polynomial time (in the lengths of the proof and of the tautology). The basic question is what is the minimal complexity of proofs of a given tautology in a given proof system, with the most important complexity measure being the length of the proof; in particular, we are interested if there are tautologies that require proofs of superpolynomial, or even exponential, length. This is related to computational complexity: there exists a polynomially bounded proof system (i.e., one where every tautology has a polynomial-size proof) if and only if $\mathrm{NP} = \mathrm{coNP}$.

In [63], Cook builds upon Parikh's idea of formalization of feasibly constructive reasoning by introducing an equational theory PV (for "polynomially verifiable"), which has function symbols for polynomial-time algorithms, and a polynomial induction rule. Importantly, he establishes a prototypical *propositional translation* of bounded arithmetic: a true equation in the language of PV can be translated into a sequence of propositional tautologies, and if the equation is provable in PV, its translations have polynomial-time constructible proofs in the extended resolution proof system (equivalent to extended Frege, EF). This result became a precursor to a more general correspondence between theories of arithmetic and propositional proof systems; in a sense, bounded arithmetical theories (or at least their low-complexity fragments) may be considered to be "uniform versions" of propositional proof systems.

A different (though ultimately related) translation of $I\Delta_0(R)$ to bounded-depth Frege was later introduced by Paris and Wilkie [144]. A first-order variant PV_1 of PV was defined by Krajíček, Pudlák, and Takeuti [121].

Buss [37] reformulated $I\Delta_0+\Omega_1$ as the theory T_2 in an expanded language including function symbols $\lfloor x/2 \rfloor$, $|x|$ (meaning $\lceil \log_2(x+1) \rceil$), and $x\#y$ (meaning $2^{|x|\cdot|y|}$). An important advantage of this language is that the individual levels Σ_i^{P} of the polynomial-time hierarchy (including $\Sigma_1^{\mathrm{P}} = \mathrm{NP}$) have transparent syntactic descriptions: they correspond to Σ_i^b formulas of the bounded quantifier alternation hierarchy (ignoring the so-called sharply bounded quantifiers $\exists x \le |t|$, $\forall x \le |t|$). This makes the connections of the theory to computational complexity classes much tighter, and provides a good motivation to shift focus from full bounded arithmetic T_2 to its fragments with induction restricted to Σ_i^b formulas. Buss introduced two main sequences of such fragments: the theories $T_2^i = \Sigma_i^b\text{-}\mathit{IND}$ with usual induction, and $S_2^i = \Sigma_i^b\text{-}\mathit{PIND}$ using the *polynomial induction* schema; they form an intertwined hierarchy $S_2^1 \subseteq T_2^1 \subseteq S_2^2 \subseteq T_2^2 \subseteq S_2^3 \subseteq T_2^3 \subseteq \cdots$. It is often convenient to extend the language of Buss's theories further by including all function symbols of PV, which are Σ_1^b-definable in S_2^1. This unifies this set-up of bounded arithmetic with Cook's PV; the theory $S_2^1(PV)$ is a conservative extension of PV, and in fact, a $\forall\Sigma_1^b$-conservative extension of $PV_1 = T_2^0(PV)$ as a consequence of Buss's witnessing theorem.

Buss's one-sorted ("first-order") theories became one of the two most commonly studied frameworks for theories of bounded arithmetic. The other one is the framework of two-sorted ("second-order") theories with a sort of small/unary integers, and a sort of finite sets of these, also viewed as binary strings, or as large/binary integers. These theories were in fact also originally introduced by Buss [37], but current usage follows the considerably simpler set-up due to Zambella [182]. The one-sorted and two-sorted theories are related by the *RSUV*-isomorphism (Takeuti [170]), which identifies the second (set/string/binary) sort of the two-sorted theories with numbers of the one-sorted theories, and the first (number/unary) sort with logarithmically small numbers. In this way, the theories S_2^i and T_2^i are bi-interpretable with two-sorted theories V^i and TV^i (for $i \ge 1$). While the one-sorted theories are formally simpler, the advantage of two-sorted theories is that one can easily introduce theories corresponding to small complexity classes that do not necessarily include (binary) integer multiplication; in particular, the base theory V^0 corresponds to the important complexity class (DLOGTIME-uniform) AC^0.

2 Our contribution

The work presented in this dissertation is an investigation of several themes in the subject of bounded arithmetic and related complexity theory. While each chapter was published as a separate paper, they are connected in various ways. We will now give brief introductions of the individual topics.

2.1 Approximate counting

The first two chapters of the dissertation (originally published as [96, 98]) are devoted to *approximate counting.* Here, counting refers to determination of the cardinality of a finite set. In the context of theories of arithmetic, we consider definable bounded sets X (i.e., subsets of intervals $[0,a)=\{x:x<a\}$ for some parameters a), and ideally, we would like to define the cardinality $|X|\le a$ in such a way that we can conveniently manipulate it in the theory. Such a notion of definable cardinality can be useful for formalization of all kinds of counting arguments or probabilistic arguments in combinatorics, complexity theory, number theory, etc., and for presentation of randomized algorithms in theories of arithmetic.

It is not difficult to show that the cardinality of all bounded definable sets has a well-behaved definition in PA. In fragments of arithmetic with restricted induction schema, we cannot expect this to work for arbitrary sets, only for sets whose definitions have a sufficiently low complexity; with this caveat, counting still works well in the "strong fragments" $I\Sigma_k$ for $k\ge1$ (where we can count $\Delta_0(\Sigma_k)$-definable sets), down to the theory $I\Delta_0+EXP$, where we can count $\Delta_0(\exp)$-definable sets. Perhaps the best way to think about it is that a very weak fragment of bounded arithmetic (PV_1, or even VTC^0) has a well-behaved definition of counting for sets explicitly encoded as sequences of elements, and then strong fragments of arithmetic prove bounded comprehension principles that ensure that a bounded definable set of suitable complexity can be arranged into a sequence: $I\Delta_0+EXP$ proves this form of comprehension for $\Delta_0(\exp)$ sets, and $I\Sigma_k$ proves comprehension for Σ_k (and consequently, $\Delta_0(\Sigma_k)$) sets.

However, this set-up essentially requires the presence of exponentiation, as subsets of $[0,a)$ need more than a bits to encode, which is exponential in the bit-length of a. In fact, there are good reasons to think that exact counting of, say, polynomial-time bounded sets is not possible to define in *any* reasonable way in theories of bounded arithmetic (say, subtheories of Buss's T_2): if we could define it (in a provably total way) by a Σ_1 formula, then using Parikh's theorem, we could define it by a bounded formula, i.e., Σ^b_i for some i. Thus, it would belong to the complexity class $\mathrm{FP}^{\Sigma^{\mathrm{P}}_i}$. However, counting of polynomial-time sets is #P-complete, and this class is PH-hard by Toda's theorem [174]; thus, we would get the collapse of the polynomial-time hierarchy (and in fact the whole counting hierarchy CH) to $\Delta^{\mathrm{P}}_{i+1}$, which is generally assumed to be quite unlikely. It fact, it is outright *disprovable* for often-considered "relativized" variants of bounded arithmetic with an uninterpreted new predicate. A similar argument applies even to *modular counting*, i.e., determination of $|X|$ modulo a fixed constant m.

This leaves open the possibility that bounded arithmetic can define an *approximation* of $|X|$, up to a polynomially small error ε. Here, we may consider either additive error, i.e., we want to compute s such that $|X|-\varepsilon a\le s\le|X|+\varepsilon a$, where $X\subseteq[0,a)$, or multiplicative error, in which case we want s such that $|X|(1-\varepsilon) \leq s \leq |X|(1+\varepsilon)$. (Counting with multiplicative error is more precise than with additive error, especially when X is rather sparse.) It is known that approximate counting can be accomplished within the polynomial-time hierarchy, hence there is no a priori complexity obstacle to formalization in bounded arithmetic.

The *pigeonhole principle* $\mathit{PHP}^{a+1}_a(f)$ asserts that a function $f\colon [0,a+1) \to [0,a)$ cannot be injective. This amounts to a "passive" form of (exact) counting: the most natural way of witnessing that a bounded set X has size s is to provide a bijection $X \to [0,s)$; while *PHP* does not (at least in an obvious way) imply that such bijections exist, it ensures that *if* they exist for a given X, then s is unique. In a similar way, a passive form of approximate counting is provided by the *weak pigeonhole principle* $\mathit{PHP}^b_a(f)$ where b is much larger than a. (The exact meaning of "much" depends on the context. Common choices include $b = a^2$ or $b = 2a$; below, we will take $b = a(1+1/|a|)$, which corresponds to counting with polynomially small error $\varepsilon \approx 1/|a|$. That is, if Φ is a class of definable functions, $\mathit{WPHP}(\Phi)$ denotes the schema $\forall a\, \mathit{PHP}^{(1+1/|a|)a}_a(f)$ for $f \in \Phi$.)

In bounded arithmetic, *PHP* is as intractable as other forms of exact counting; in particular, the relativized theory $T_2(\alpha)$ does not prove $\mathit{PHP}(\alpha)$ [3, 24]. But crucially, it *does* prove the weak pigeonhole principle, as shown by Paris, Wilkie, and Woods [146]; more precisely, $T^{i+1}_2(\alpha) \vdash \mathit{WPHP}(\Sigma^b_i(\alpha))$ for $i \geq 1$ by Maciel, Pitassi, and Woods [125]. This already shows that bounded arithmetic is capable of approximate counting to some extent. Besides being an interesting principle in its own right, *WPHP* can be used to simulate certain counting arguments in T_2: the very reason it was introduced in [146] was to prove the unboundedness of primes, and for another important example, Pudlák [152] used it to prove Ramsey's theorem.

Although it may look less intuitive at first sight, the *surjective* (or *dual*) weak pigeonhole principle happens to be more useful for formalization of counting arguments in bounded arithmetic than the usual injective principle: for $a < b$, $s\mathit{PHP}^a_b(f)$ says that $f\colon [0,a) \to [0,b)$ cannot be onto, and $s\mathit{WPHP}(\Phi)$ denotes $\forall a\, s\mathit{PHP}^a_{a(1+1/|a|)}$ for each $f \in \Phi$. We are particularly interested in the case of Φ being the set of all *PV*-functions. By [125], we still have $T^2_2 \vdash s\mathit{WPHP}(\mathit{PV})$, and more generally, $T^{i+1}_2 \vdash s\mathit{WPHP}(\Sigma^b_i)$ for $i \geq 1$. From now, we will denote the original injective form of the weak pigeonhole principle as *iWPHP* rather than *WPHP* to distinguish it from *sWPHP*.

The basic idea of emulating counting arguments by *iWPHP* is that we witness $|X| \leq s$ by providing an injective function (efficient, say, computable in polynomial time or by polynomial-size circuits) $X \to [0,s)$, whereas if we work with *sWPHP*, we use a surjection $[0,s) \to X$. It turns out that such counting surjections are easier to construct and manipulate than injections[1].

Another reason suggesting a close connection of *sWPHP* to counting or probabilistic arguments is Wilkie's witnessing theorem (first published in Krajíček [116]): the $\forall\Sigma^b_1$ consequences of $S^1_2 + s\mathit{WPHP}(\mathit{PV})$ can be witnessed in randomized polynomial time (equivalently, the NP-search problems provably total in $S^1_2 + s\mathit{WPHP}(\mathit{PV})$ are included in TFZPP). The basic intuition for

[1]A similar phenomenon occurs in set theory without the axiom of choice, where comparing cardinalities by existence of surjections sometimes works in a more robust way than with injections: e.g., the most natural definition of inaccessible cardinals in *ZF* renders κ being strong limit as "there is no surjection from V_α to κ for any $\alpha < \kappa$" [32, 167].

this result is that if we are given a poly-time function $f\colon [0,a) \to [0,b)$ with $b \gg a$, we can find an element outside the range of f with high probability by considering a random element of $[0,b)$.

In contrast, witnessing $iWPHP(PV)$ is computationally hard: e.g., it is at least as hard as integer factoring [95]. Strictly speaking, $sWPHP(PV)$ and $iWPHP(PV)$ are (presumably) incomparable; nevertheless, $sWPHP(PV)$ is weaker than $iWPHP(PV)$ in that $S^1_2 + sWPHP(PV)$ is $\forall\Sigma^b_1$-conservative over $S^1_2 + iWPHP(PV)$. More precisely, the $\forall\Sigma^b_1$ consequences of $S^1_2 + sWPHP(PV)$ are axiomatized by $PV_1 + rWPHP(PV)$, where the *retraction-pair* weak pigeonhole principle $rWPHP(PV)$ asserts that it is impossible for two PV-functions $f\colon [0,a) \to [0,b)$ and $g\colon [0,b) \to [0,a)$ to satisfy $f \circ g = \mathrm{id}_{[0,b)}$ if $b \gg a$ [172, 93]. Clearly, $rWPHP$ is implied by $iWPHP$.

As we already mentioned, some counting arguments were formalized in bounded arithmetic using variants of $WPHP$ for example in [146, 152]. Both papers rely on ingenious (and, at least in the case of [146], quite complicated) constructions of counting functions designed ad hoc to make the arguments go through. They do not give any suggestions how to turn this into a general method; as a case in point, the *tournament principle* (due to Erdős [77]) can be proved by a simple counting argument quite similar to a proof of Ramsey's theorem, but it stood open for a long time whether it can be proved in a bounded arithmetic (this problem, along with a certain generalization relevant to relating the collapse of Buss's hierarchy to the collapse of the polynomial-time hierarchy, originated in Krajíček, Pudlák, and Takeuti [121]; it was stated explicitly in Clote and Krajíček [55]).

The main goal of Chapters I and II is to develop a systematic framework for formalization of approximate counting and probabilistic arguments in bounded arithmetic using $sWPHP$, including a toolbox of basic facts.

Chapter I (originally [96]) is devoted to approximate counting with *additive error*, working in the theory $PV_1 + sWPHP(PV)$, also called APC_1 in [45]. (It partially builds on [93], which is however not included in this dissertation.) The basic idea is that if $X, Y \subseteq [0, 2^n)$ are sets defined by Boolean circuits (or equivalently, by PV-functions with parameters), we witness that $|X| \le |Y|$ by the existence of a circuit that computes a surjection $Y \twoheadrightarrow X$, but we weaken it in two ways to make it much easier to construct such circuits: first, we actually consider surjections $Y \times [0,v) \twoheadrightarrow X \times [0,v)$ for some $v > 0$, and second, instead of Y, we take its disjoint union with $[0, \varepsilon 2^n)$ for some rational $\varepsilon > 0$. We denote the resulting concept by $X \preceq_\varepsilon Y$, spelled out as *the size of X is approximately less than the size of Y with error ε*. We also write $X \approx_\varepsilon Y$ if $X \preceq_\varepsilon Y$ and $Y \preceq_\varepsilon X$.

The crucial result that makes this definition well behaved is that $PV_1 + sWPHP(PV)$ proves that any set "has a size": that is, given $X \subseteq [0, 2^n)$ as above, and ε at least inverse polynomial in n (or more generally, in a length of something), there exists $s \le 2^n$ such that $X \approx_\varepsilon [0, s)$. (The witnessing surjections also have inverse injections computable by small circuits.) This is shown by formalization of the analysis of the Nisan–Wigderson pseudorandom generator [135]: supplied with the truth-table of a sufficiently hard Boolean function (whose non-uniform existence follows from $sWPHP(PV)$), the NW generator can compute the approximate size s by sampling X, and the proof of "correctness" of this estimate can be turned into a construction of the witnessing counting functions.

Besides basic consequences of the definition (such as monotonicity), we show that it behaves in the expected way with respect to disjoint unions and Cartesian products (the latter generalizes to a formalization of the averaging principle: if $\Pr_{x\in X, y\in Y}[P(x,y)] \geq p$, there exists $x \in X$ such that $\Pr_{y\in Y}[P(x,y)] \geq p$). For more sophisticated counting arguments, we prove a form of the inclusion–exclusion principle and a Chernoff–Hoeffding bound.

In the second half of Chapter I, we apply this machinery to develop the theory of various *randomized complexity classes* in $PV_1 + sWPHP(PV)$: specifically, we look at the classes of FRP and TFRP search problems, BPP languages and promise problems, APP real-valued functions (introduced by Kabanets, Rackoff, and Cook [111]), MA languages and promise problems, and—upgrading the theory by one level of the hierarchy to $T^1_2 + sWPHP(PV_2)$—the classes of AM languages and promise problems. For each class, we indicate how to formally define algorithms from the class in bounded arithmetic using the approximate counting framework, and we prove basic properties of the class in the theory, such as amplification of the success probability, simulation of randomness by nonuniformity, and standard inclusions between the classes. (Along the way, we solve an open problem from [111] on the recursive enumerability of APP, and find a proof of success amplification for APP which is much simpler than the original one as given in [111].)

We now turn to Chapter II (originally published as [98]), devoted to approximate counting with *multiplicative error*. In contrast to Chapter I, we work in the theory $T^1_2 + sWPHP(PV_2)$ (called APC_2 in [45]), which is up one level of the hierarchy from $PV_1 + sWPHP(PV)$. The basic idea is taken from Sipser's coding lemma [166], which employs a universal family of hashing functions (specifically, $\mathbb{F}_2$-linear functions, represented by matrices) to distinguish sets X of size $\leq s$ from sets of size $\Omega(s \log s)$. We consider here bounded sets X definable by Σ^b_1-formulas (i.e., NP/poly). In order to get the error down to εs for a polynomially small ε, we apply Sipser's definition to a suitable Cartesian power X^c in place of X itself; we write $X \precsim_\varepsilon s$ for the resulting notion (note the difference from $\preceq_\varepsilon$). The key result is that, up to relative error ε, $X \precsim_\varepsilon s$ is equivalent to the existence of PV_2-surjections $s^c \twoheadrightarrow X^c$ for some c: we prove this in $T^1_2 + sWPHP(PV_2)$ by formalization of Sipser's lemma, using the machinery from Chapter I for probabilistic reasoning.

Again, we provide a toolbox showing that the definition of $X \precsim_\varepsilon s$ interacts in the expected way with finite unions, Cartesian products, and more generally, unions of parameterized families, i.e., averaging principles. (Due to the asymmetry of the $X \precsim_\varepsilon s$ relation, all these results need to have upper bound and lower bound versions, rather different from each other.) We also prove that any bounded Σ^b_1-definable X has an "almost bijective" increasing enumeration by a PV_2-function, in a suitable sense.

As applications, we show how the general framework can be used to formalize various counting arguments in combinatorics and complexity theory. We prove Ramsey's theorem (using a much simpler proof than Pudlák [152]) and the tournament principle (solving the open problem from [55]). In fact, we prove a multi-dimensional generalization of the tournament principle with several applications. First, we use it to directly formalize in bounded arithmetic the argument from [121] relating collapse of the T_2 hierarchy to collapse of PH, which improves the previously known results in this area [121, 39, 182, 67]: specifically, we show that if $T^i_2 = S^{i+1}_2$, then $T^i_2 = T_2$ proves that $\Sigma^b_\infty = \mathcal{B}(\Sigma^b_{i+1}) \subseteq \Delta^b_{i+1}/\operatorname{poly}$ and $\Sigma^b_\infty \subseteq \Sigma^b_{i+1}/O(1)$. Second, even in the two-dimensional case, our generalized tournament principle applies to arbitrary directed graphs rather than just tournaments; we use this to formalize in $T^1_2 + sWPHP(PV_2)$ the result $\mathrm{S}^P_2 \subseteq \mathrm{ZPP}^{\mathrm{NP}}$ due to Cai [47]. We also use our approximate counting machinery to prove that any interval in a model of T_2 admits a nontrivial approximate Euler characteristic in the sense of Krajíček [118], and we prove in $T^1_2 + sWPHP(PV_2)$ that graph isomorphism is in coAM.

Let us mention some follow-up work. Buss, Kołodziejczyk, and Thapen [45], besides introducing the names APC_1 and APC_2 for the theories $PV_1 + sWPHP(PV)$ and $T^1_2 + sWPHP(PV_2)$, prove $\forall\Sigma^b_1$-separations of several fragments of APC_2 from APC_2 itself and from T^2_2 in the relativized setting. (One case they left open was solved by Atserias and Thapen [15].) The separations are based on the fact that (using the approximate counting machinery and the tournament principle) APC_2 proves the *ordering principle*, which states that any partial order on a nonempty bounded domain has a minimal element.

Using our approximate counting, Buss, Kołodziejczyk, and Zdanowski [46] formalize Toda's theorem on the collapse of $\mathrm{Mod}_p\,\mathrm{PH}$ to $\mathrm{BP}\cdot\oplus_p\mathrm{P}$ in bounded arithmetic relativized with a $\oplus_p\mathrm{P}$ oracle, specifically showing that $APC_2^{\oplus_p\mathrm{P}} = T_2(\oplus_p\mathrm{P})$. Using the Paris–Wilkie translation, they obtain a collapse for propositional proof systems: the constant-depth Frege system with $\oplus_p$ gates is quasipolynomially simulated by its depth 3 fragment (using $\bigwedge$ of $\oplus_p$ of polylogarithmic $\bigwedge$ of literals). We recall that proving superpolynomial lower bounds for constant-depth Frege with $\oplus_p$ gates is one of the longest-standing open problems in proof complexity.

Pich [150] formalizes the exponential PCP theorem in APC_1 using our approximate counting, and proceeds to prove the full PCP theorem (scaled logarithmically down, whence the weaker theory) in PV_1. Müller and Pich [129] employ approximate counting to formalize in APC_1 several prominent super-polynomial circuit lower bounds: AC^0 lower bounds for Parity, $\mathrm{AC}^0[p]$ lower bounds for Mod_q, and monotone lower bounds for Clique. They also formalize the Razborov–Rudich [157] theorem on natural proofs.

2.2 Abelian groups, quadratic residues, and factoring

Chapter III (originally published as [99]) is devoted to a formalization of several inter-related problems from modular arithmetic, elementary number theory, and algebra in suitable fragments of bounded arithmetic. As a follow-up, some of these results are used in Chapter IV (originally published as [105]) to draw consequences in pure computational complexity that are not a priori connected to bounded arithmetic, specifically about the complexity of integer factoring.

The first motivating problem (still unresolved) for Chapter III is whether some fragment of bounded arithmetic T_2 proves *Fermat's little theorem* (FLT): $a^p \equiv a \pmod p$ for all a and prime p. This is a basic principle of modular arithmetic which admits a number of simple elementary proofs (to name a few, using Lagrange's theorem, by counting necklaces, or by induction on a using the binomial theorem), nevertheless all seem to require exact counting or exponential-size sums or objects, and as such resist formalization in bounded arithmetic. On the other hand, there is no evidence that it is hard for bounded arithmetic[2]. We mention that, as shown in [93], a natural formalization of the Rabin–Miller coRP primality testing algorithm in APC_1 using approximate counting has the property that its correctness is provably equivalent to FLT (in other words, APC_1 proves that $\forall a\,\bigl(0 < a < p \to a^{p-1} \equiv 1 \pmod{p}\bigr)$ is a coRP predicate of p).

More generally, we may ask about the structure of the multiplicative groups $\mathbb{F}_p^\times$ for prime p. FLT asserts that these groups have exponent $p-1$. Another important property is that these groups are *cyclic*, which is expressed by the formula $\exists g < p\,\forall a\,\bigl(0 < a < p \to \exists u\, g^u \equiv a \pmod{p}\bigr)$. Like FLT, it is an open problem whether the cyclicity of $\mathbb{F}_p^\times$ for all primes p is provable in T_2. Cyclicity and FLT together are equivalent over $S^1_2 + iWPHP(PV)$ to the statement that primes have Pratt's primality certificates (while the theory unconditionally proves the converse), making primality Σ^b_1.

Another elementary principle related to FLT is *Euler's criterion*, stating that

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}$$

for all a and odd primes p, where $\left(\frac{a}{p}\right)$ or $(a|p)$ denotes the *Legendre symbol*

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } p \nmid a \text{ and } a \text{ is a quadratic residue modulo } p, \\ -1 & \text{if } p \nmid a \text{ and } a \text{ is a quadratic nonresidue modulo } p, \\ 0 & \text{if } p \mid a. \end{cases}$$

Clearly, Euler's criterion implies FLT, and we may ask how much stronger (if at all) it is. This brings us to properties of quadratic residues and of the Legendre symbol. The most fundamental properties of the Legendre symbol are its *multiplicativity*

$$\left(\frac{a}{p}\right)\left(\frac{b}{p}\right) = \left(\frac{ab}{p}\right)$$

(which is implied by Euler's criterion), and the celebrated *quadratic reciprocity theorem* (*QRT*)

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}$$

for odd primes $p \neq q$, together with the *supplementary laws*

$$\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2},$$
$$\left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}.$$

[2]Krajíček and Pudlák [120] show that FLT is not provable in S^1_2 if RSA is secure against polynomial-time attacks, and Thapen [171] points out the same applies to APC_1 assuming security of RSA against randomized polynomial time. In fact, it follows from [93] that the weaker assumption that factoring is not possible in randomized polynomial time suffices. However, all these results are misleading in that they actually show the unprovability of the much weaker statement that any a not divisible by a prime p has a finite order mod p. This is a consequence of FLT, but it is also provable in $S^1_2 + iWPHP(PV)$; thus, the results really show the conditional independence of $iWPHP(PV)$ from APC_1 (which is also mentioned in [95]), and do not say anything about the provability of FLT in T_2, or even in $S^1_2 + iWPHP(PV)$.

These properties also imply the corresponding statements for the *Jacobi symbol* $(a|n)$, which extends the Legendre symbol to all odd $n > 0$ such that it is completely multiplicative in n; one consequence is that it leads to simple polynomial-time algorithms (similar to the Euclidean or binary GCD algorithms) for computing the Jacobi, and therefore Legendre, symbol.

QRT was originally proved by Gauss (who found no less than eight proofs in his lifetime), and until today, well over 300 proofs (not all essentially different, of course) were published by various authors; see Lemmermeyer [123] for the history. In the context of weak theories of arithmetic, Cornaros [70] proved QRT in $I\mathcal{E}^2_*$ (a fragment of arithmetic with induction for LinSpace predicates), and Berarducci and Intrigila [29] proved the supplementary laws in $I\Delta_0$ extended with modular counting principles. (They also proved multiplicativity of the Legendre symbol in $I\Delta_0 + \mathit{WPHP}(\Delta_0)$; their proof works in $PV_1 + i\mathit{WPHP}(PV)$, too.) D'Aquino and Macintyre [73, 74] developed the basic theory of quadratic forms in $I\Delta_0 + \Omega_1$ with a vision of eventually formalizing a proof of QRT along the lines of Gauss's second proof, but so far this did not materialize.

We tackle the problems introduced above in Chapter III in the following ways. As a first step towards clarifying the structure of the multiplicative groups $\mathbb{F}_p^\times$ of prime fields, we look at the structure of *arbitrary* finite abelian groups (more precisely, Σ^b_1-definable groups with a bounded domain): we prove the fundamental theorem that any such group is a direct sum of (essentially unique) cyclic groups of prime power order in the theory $S^2_2 + i\mathit{WPHP}(\Sigma^b_1)$; if the group operation is defined by a PV-function, $S^2_2 + i\mathit{WPHP}(PV)$ is enough.

It is a basic observation that $i\mathit{WPHP}$ implies that any element of a finite abelian group has a finite order, from which it easily follows that any finite abelian group is a direct sum of its p-primary components (this is already noted by D'Aquino and Macintyre [72] for $I\Delta_0+\Omega_1$). On the other hand, S^2_2 (the Σ^b_2-LMAX principle) implies the existence of maximal independent sets in finite structures such as modules, as long as they are a priori of logarithmic size: e.g., this shows in the presence of $i\mathit{WPHP}$ (which gives the logarithmic bound) that any finite $\mathbb{F}_p$-linear space has a basis, as noted by Riis [160]. Usually, proofs of the fundamental theorem for finite abelian groups proceed by first establishing the decomposition to primary components, and then proving the theorem for p-groups by induction on $|G|$ (passing from G to its subgroups or quotients). It's not immediately clear how to set up such inductive arguments so that we do not need to quantify over exponentially large objects (all groups of a given size; recall that the size is given in "binary", thus our groups are too large to be represented by a multiplication table). We manage to overcome this issue by finding a more direct proof, generalizing the maximal independent set argument to show the existence of a "basis" of any finite abelian group, again using $i\mathit{WPHP}$ to get a logarithmic size bound.

Returning to Fermat's little theorem, if p is a prime, then $\mathbb{F}_p^\times$ is a group with domain $[1,p)$; by the structure theorem, there is an isomorphism $f\colon \mathbb{F}_p^\times \to G = \bigoplus_{i<k} C(p_i^{e_i})$ (defined by a PV-function) for some sequence $\langle p_i^{e_i} : i < k\rangle$ of prime powers. Here, G is a group with domain $[0,a)$ where $a = \prod_i p_i^{e_i}$, and it has exponent a, thus so does $\mathbb{F}_p^\times$. The FLT would follow if $a = p-1$. The weak pigeonhole principle applied to f ensures that $a \approx p$, and we know that a is even, but other than that, it seems quite difficult to rule out that, say, $a = p+1$, in which case the FLT spectacularly fails. Thus, we only obtain a proof of FLT if we add to $S^2_2 + i\mathit{WPHP}(PV)$

the *strong* pigeonhole principle $\mathit{PHP}(\mathit{PV})$, which is likely not provable in T_2. All in all, this argument seems to suggest that FLT is *not* provable in T_2, but the evidence is very weak.

Concerning the cyclicity of $\mathbb{F}_p^\times$, the structure theorem implies (over $S^2_2 + i\mathit{WPHP}(\mathit{PV})$) that it is equivalent to there not being too many qth roots of unity in $\mathbb{F}_p$ for any prime $q \neq p$. More precisely, we obtain an interesting dichotomy: either $\mathbb{F}_p^\times$ is cyclic, and for any prime $q \neq p$, $[0,q)$ PV-surjects onto $\{x \in \mathbb{F}_p : x^q = 1\}$; or $\mathbb{F}_p^\times$ is not cyclic, and there exists a prime $q \neq p$ such that $[0,q^2)$ PV-injects into $\{x \in \mathbb{F}_p : x^q = 1\}$. (Both functions also have suitable PV inverses.) Thus, one way to prove the cyclicity of $\mathbb{F}_p^\times$ in bounded arithmetic might be to formalize the principle that a degree-q sparse polynomial (here, $x^q - 1$) may only have at most q roots in a finite field, in the sense of approximate counting. Usual proofs of this fact require the existence of exponentially large objects (e.g., a proof by induction on the degree of the polynomial would need the induction hypothesis to apply to *non-sparse* polynomials as well), but there is a distinct possibility that this can be circumvented somehow.

Next, we look at Euler's criterion. We first show that the Legendre symbol $(-|p)$ is multiplicative whenever $\mathbb{F}_p^\times$ is a torsion group (this improves the result of Berarducci and Intrigila [29] on its being provable from $i\mathit{WPHP}(\mathit{PV})$). We use this to show that Euler's criterion is equivalent over S^1_2 to the conjunction of FLT with the statement $\exists a\, a^{(p-1)/2} \equiv -1 \pmod p$. In particular, Euler's criterion is provable in $S^2_2 + i\mathit{WPHP}(\mathit{PV}) + \mathit{PHP}(\mathit{PV})$. We also observe that, assuming FLT, the assertion $\exists a\, a^{(p-1)/2} \equiv -1 \pmod p$ amounts to the sparse polynomial $x^{(p-1)/2} - 1$ having less than $p-1$ roots, hence we are in a similar situation as with the cyclicity of $\mathbb{F}_p^\times$.

The last section of Chapter III is devoted to quadratic reciprocity. Our starting point is the observation that elementary proofs of QRT often distinctly involve some form of counting modulo 2 (e.g., proofs based on Gauss's or Zolotarev's lemmas, or Eisenstein's proof); in the context of arithmetic, as we already mentioned, Berarducci and Intrigila used counting modulo 4 and 8 (in the form of so-called equipartition principles) to prove the supplementary laws. Since modular counting (much like exact counting) is not available in T_2, this suggests that we should look at some extension of bounded arithmetic, but that perhaps some form of counting principles mod 2 is all we need. The weakest modular counting principle we thought of expresses that we cannot partition an odd-size domain $[0, 2a+1)$ into a disjoint union of two-element sets, where the partition is represented in a very explicit way by a PV-function f that maps each element to its partner. In other words, f is a fixpoint-free involution. Thus, our counting principle $\mathit{Count}_2(\mathit{PV})$ states that every involution on $[0, 2a+1)$ defined by a PV-function has a fixpoint. Interestingly, this principle was used outside the formalism of bounded arithmetic in slick proofs of Fermat's theorem on sums of two squares (related to the first supplementary law of QRT) by Heath-Brown [89] and Zagier [181].

We find a short proof of QRT (as well as the supplementary laws, and multiplicativity of the Legendre symbol) using only simple manipulations of involutions, which can be formalized in $\mathit{PV}_1 + \mathit{Count}_2(\mathit{PV})$ or $I\Delta_0 + \mathit{Count}_2(\Delta_0)$ (more precisely, $\mathit{IE}_1 + \mathit{Count}_2(\nabla_1)$). The proof is loosely based on Gauss's third proof, but we replace the key Gauss's lemma by a formulation with explicit involutions. Strengthening the base theory from PV_1 to S^1_2, we can also prove the corresponding statements about the Jacobi symbol, leading to its being polynomial-time computable (i.e., equivalent to a PV-function).

This brings us to Chapter IV, which is devoted to the computational complexity of integer factoring, and the closely related problem of computing modular square roots. Factoring is one of the most fundamental problems in mathematical computation, going back to classical antiquity. In modern times, it has significant applications in cryptography. In particular, various cryptographic protocols rely on the computational hardness of factoring: it is generally assumed it has no deterministic or randomized algorithms running faster than in exponential time (with polynomial exponent); the best currently known general-purpose factoring algorithm (GNFS) has running time $2^{\widetilde{O}(n^{1/3})}$. It also has an efficient quantum algorithm.

Factoring is closely related to the problem of computing square roots modulo a given integer. There are randomized poly-time algorithms for computation of square roots modulo primes; using Hensel's lifting and the Chinese remainder theorem, this gives a randomized reduction of square roots with general moduli to factoring. One can also give randomized reductions in the opposite direction.

When formulated as a decision problem, factoring sits somewhere inside NP $\cap$ coNP (more precisely, UP $\cap$ coUP) and in BQP. However, here we study the complexity of factoring as a *total* NP*-search problem*, which is arguably a more natural setting than as a decision problem. The influential paper by Papadimitriou [140] introduced several classes of NP-search problems (subclasses of TFNP) that are based on "combinatorial proofs" of totality: in particular, he defined a class PPP corresponding to the (injective) pigeonhole principle, and several classes based on "parity arguments"—we are interested here in the class PPA, whose defining complete problem is, given a circuit representing an undirected graph of degree 2, and a vertex of degree 1, find another such vertex. Papadimitriou posed the question whether FACTORING belongs to some of his classes. The first progress on this problem was made by Buresh-Oppenheim [35], who proved that factoring of integers of a certain special form is in PPA, and has a randomized reduction to a PPP problem.

We prove in Chapter IV that the general FACTORING problem has a randomized reduction to a PPA problem, and to a problem in the subclass PWPP of PPP corresponding to $iWPHP$. We can derandomize the reductions under the assumption of the Riemann hypothesis for quadratic Dirichlet L-functions. Moreover, we prove unconditionally that PPA contains the problems of computing modular square roots, and finding square nonresidues (the latter problem is not really hard, as it can be done in randomized polynomial time; however, an efficient deterministic algorithm is not even known modulo primes, hence a deterministic reduction to PPA is nontrivial).

Our basic strategy is to apply a witnessing theorem to the results of Chapter III on provability of the quadratic reciprocity theorem. First, it is easy to show that the theory $S^1_2 + Count_2(PV)$ corresponds to PPA, in the sense that NP-search problems witnessing its $\forall\Sigma^b_1$ consequences are in PPA (the $Count_2$ principle is virtually identical to an alternative PPA-complete problem LONELY from Beame et al. [23]). Since the theory proves that the Jacobi symbol $(a|n)$ is computable by a PV-function, say $J(a,n)$, it also proves the $\forall\Sigma^b_1$ sentence expressing "if $J(a,n)=1$, then a is a quadratic residue mod n, unless n is composite". Thus, PPA contains the following problem FACROOT: given a,n such that $(a|n)=1$, find a square root of a mod n, or a proper divisor of n.

In order to make the original paper self-contained and accessible to an audience of complexity researchers not necessarily familiar with bounded arithmetic, we include a direct combinatorial proof of FacRoot $\in$ PPA. Its main part is a rather complicated dynamic programming algorithm. We believe the bounded arithmetic proof is much more transparent; this seems to be a not-so-common case where witnessing applied to a bounded arithmetic proof yields a genuinely new algorithm (the more usual situation is that one needs to know an algorithm in the first place before formalizing a result in bounded arithmetic; the typical use case for witnessing theorems is to show *unprovability* in arithmetical theories).

Now, to reduce Factoring to FacRoot, it suffices to choose a at random: we show easily that if n is odd and not a prime power, then with a probability $\geq 1/4$, we have $(a|n) = 1$, but a is not a quadratic residue, hence FacRoot(a, n) must split n. On the other hand, we can show that FacRoot can be used (deterministically) to compute square roots mod n. While our reduction of general factoring to PPA is randomized, we exhibit a class of special cases of factoring that are deterministically in PPA; this generalizes the original result of Buresh-Oppenheim.

The reduction of Factoring to PWPP we also include is based on the proof of multiplicativity of the Legendre symbol in $PV_1 + iWPHP(PV)$, which gives the following search problem: given odd n and a, b, compute a square root of one of a, b, or ab modulo n, or split n. Again, this leads to a factoring algorithm just by choosing a and b at random.

2.3 Sorting networks and monotone sequent calculus

The material in Chapters V and VI (originally published as [100, 101]) is primarily motivated by a problem from propositional proof complexity. As we already mentioned, one of the most fundamental proof systems is the *Frege system.* This system is quite robust in that it can be presented in a variety of ways which turn out all to be p-equivalent: as a system operating with formulas (over any finite functionally complete set of connectives whose choice does not matter) that allows derivation by means of a finite set of schematic axioms and rules (whose choice does not matter as long as they are sound and implicationally complete), as a natural deduction system (again, using a finite complete set of schematic rules), or as a Gentzen-style *sequent calculus LK*. It also does not matter whether proofs are required to be tree-like, or allowed to be dag-like (i.e., sequences).

An interesting variant of the sequent calculus is the *monotone sequent calculus MLK* (introduced by Pudlák [153]): it operates with two-sided sequents that only use monotone formulas, i.e., formulas using the connectives $\wedge$, $\vee$, $\perp$, and $\top$ (but not $\neg$ or $\rightarrow$); the calculus includes the usual derivation rules of the sequent calculus (including the cut rule) pertaining to the restricted language. There is a simple tautologicity-preserving reduction from arbitrary Boolean formulas (or sequents) to monotone sequents, showing the latter to be coNP-complete, and justifying the role of MLK as a fully fledged propositional proof system. A natural question to ask (dubbed the *Think Positively Conjecture* by Atserias [12]) is whether MLK is p-equivalent to the usual sequent calculus LK, in the sense that given an LK proof of a monotone sequent, we can construct its MLK proof in polynomial time. MLK is also included in the intuitionistic sequent calculus *LJ*, and we can similarly ask if *LJ* p-simulates *LK* for monotone sequents.

A part of the original motivation for studying *MLK* was that, unlike general circuits, we know exponential lower bounds on the size of monotone circuits for explicit natural problems, which suggests that perhaps we could prove unconditional exponential lower bounds on the size of *MLK* proofs, separating it from *LK*. However, it soon turned out that *MLK* is quite close to *LK*: first, Atserias, Galesi, and Gavaldà [13] proved that *MLK* has quasipolynomial proofs of the pigeonhole principle, and then Atserias, Galesi, and Pudlák [14] proved that in general, *MLK* (and therefore *LJ*) quasipolynomially simulates *LK* for all monotone sequents. A question remained whether we can improve this simulation to polynomial.

For *LJ*, the question was resolved by Jeřábek [97], who presented a simple p-simulation of *LK*-proofs of monotone sequents in *LJ*.

For *MLK* itself, the basic idea of [14] was to use the monotone *threshold* (*slice*) *functions*

$$\theta^n_k(x_0,\dots,x_{n-1}) = \begin{cases} 1 & \text{if } \bigl|\{i<n : x_i = 1\}\bigr| \ge k, \\ 0 & \text{otherwise.} \end{cases}$$

In particular, if we assume that exactly k of the variables $x_0,\dots,x_{n-1}$ are true, we can express $\neg x_i$ by the monotone function $\theta^n_k(x_0,\dots,x_{i-1},\bot,x_{i+1},\dots,x_{n-1})$, which can be used to make formulas in a proof monotone. There is a straightforward construction of quasipolynomial-size monotone formulas for θ^n_k, and this allowed [14] to prove a quasipolynomial simulation of *LK* by *MLK*. They observed that if we could find a *polynomial* construction of monotone formulas for θ^n_k such that certain basic properties of these formulas have polynomial *MLK* proofs, we would obtain a polynomial simulation of *LK* by *MLK*. But surprisingly, they also showed that the same conclusion holds if we only assume that the properties of θ^n_k have polynomial *LK proofs*, using a sort of boot-strapping argument.

Polynomial-size monotone formulas for θ^n_k do, in fact, exist: first, Ajtai, Komlós, and Szemerédi [5, 4] proved that there are *sorting networks* of depth $O(\log n)$, which also gives monotone formulas of depth $O(\log n)$ for θ^n_k, and second, Valiant [175] gave a simple probabilistic construction of such formulas. However, in both cases it's quite unclear how to prove properties of the formulas efficiently in *LK*: Valiant's construction is randomized, hence it does not even give a uniformly constructible sequence of θ^n_k formulas, and gives no handle how to reason about such formulas in *LK*; the Ajtai–Komlós–Szemerédi (AKS) sorting network is explicit, but it is immensely complicated, hence proving the relevant properties in *LK* is a major undertaking, and it relies on an expander graph construction, whose formalization is a separate difficult issue on its own.

The goal of Chapters V and VI is to formalize the AKS sorting network proper (i.e., minus the expander construction) in a suitable theory of bounded arithmetic, which then yields monotone θ^n_k formulas whose defining properties have polytime-constructible *LK* proofs by means of propositional translation of bounded arithmetic, modulo an assumption that the theory can prove the existence of the necessary expanders.

The most natural theory that translates to *LK* (Frege) is VNC^1 of Cook and Morioka [61], which is a basic theory corresponding to fully uniform NC^1 (i.e., U_E-uniform NC^1 in the terminology of Ruzzo [162], or equivalently, ALOGTIME); in particular, its provably total Σ^B_1-

definable functions are exactly the uniform NC^1 functions. As such, the theory can prove that we can evaluate $O(\log n)$-depth (bounded fan-in) Boolean circuits that are presented by their *extended connection language* (*ecl*) (as defined in [162]). Unfortunately, the intricate construction of the AKS network, where elements are shuffled around by means of expanders in hard-to-predict paths, does not seem to lend itself to an efficient description of the extended connection language; we only have the direct connection language (dcl) available. This precludes formalization in VNC^1, as evaluation of $O(\log n)$-depth circuits presented by their dcl is (likely) not computable in uniform NC^1.

In order to solve this problem, we have to find a suitable theory extending VNC^1 where the formalization can go through, but such that it still translates to polynomial-size Frege proofs. This is the purpose of Chapter V. We introduce a theory VNC^1_*, axiomatized using a derivation rule that ensures that we can evaluate any $O(\log n)$-depth circuit whose dcl is definable by a formula without second-order parameters which is VNC^1_*-provably Δ^B_1. We also consider its universal conservative extension $\overline{VNC}^1_*$, whose terms correspond to the Σ^B_1-definable functions of VNC^1_*. We develop both theories and establish their basic properties. In particular, we show that the provably total computable functions of VNC^1_* form a class that includes fully uniform NC^1 functions, and is included among L-uniform NC^1 functions; we prove that $VNC^1_* + \exists\Sigma^B_1\text{-}AC$ is $\forall\exists\Sigma^B_1$-conservative over VNC^1_*; and crucially, we establish that propositional translations of $\forall\Sigma^b_0$ theorems of VNC^1_* (even in the richer language of $\overline{VNC}^1_*$) have L-uniform polynomial-size *LK* proofs.

In Chapter VI, we proceed to formalize the AKS sorting network in VNC^1_*. Our argument is actually based on the somewhat simplified construction of Paterson [148] rather than the original network from [5, 4]; we modified some inessential details to facilitate the formalization, and we stream-lined the presentation. The whole formalization is done under an assumption (left to future work) that VNC^1_* can prove the existence of suitable expander graphs.

We then apply translation of bounded arithmetic to propositional logic: the $\overline{VNC}^1_*$-function that defines the AKS network translates to an L-uniform sequence of monotone $O(\log n)$-depth formulas for the θ^n_k functions, and our VNC^1_* proof that the network correctly sorts translates to L-uniform Frege proofs establishing the defining properties of the θ^n_k formulas. Thus, all in all, we obtain a proof that *MLK* polynomially simulates *LK* on monotone sequents (the Think Positively Conjecture), modulo our assumption on the existence of expanders in VNC^1_*.

This assumption was subsequently proved (even in the weaker theory VNC^1) by Buss, Kabanets, Kolokolova, and Koucký [44], hence the polynomial simulation of *LK* by *MLK* is now fully settled. (A rudimentary form of some of their results [114] circulated already before our work here.) By results of Jeřábek [102], this also extends to a p-simulation of *LK* (on arbitrary sequents) by the proof system *MCLK* which allows arbitrary sequents in the proof, but restricts the *cut rule* to monotone cut formulas (thus, *MCLK* coincides with *MLK* when proving monotone sequents).

We mention that it remains an open problem if *tree-like MLK* p-simulates *MLK* (or equivalently, if tree-like *MCLK* p-simulates *LK*).

2.4 Induction in TC^0 theories and root finding

The last two chapters of this dissertation investigate the power of theories of bounded arithmetic corresponding to (DLOGTIME-uniform) TC^0. First, TC^0 is not just a random complexity class, but it has fundamental significance as describing the complexity of *elementary arithmetic operations*. The basic integer operations $+$, $-$, $\cdot$, $/$, and the $<$ relation, are computable in TC^0; while $+$, $-$, and $<$ are even in the subclass $\mathrm{AC}^0 \subseteq \mathrm{TC}^0$, the operations $\cdot$ and $/$ are TC^0-*complete* under AC^0 Turing reductions. We can also compute in TC^0 iterated addition $\sum_{i<n} X_i$ and iterated multiplication $\prod_{i<n} X_i$; apart from integers, we can also do the corresponding operations in rationals, Gaussian rationals $\mathbb{Q}(i)$, or number fields, as well as other structures such as polynomial rings. Using iterated addition and multiplication, we can compute approximations of analytic functions given by sufficiently nice power series, such as sin, arctan, exp, or log.

It is worth pointing out that the TC^0-computability of integer division and iterated multiplication (and other above-mentioned functions that depend on these) is a quite nontrivial result of Hesse, Allender, and Barrington [90] (building on Beame, Cook, and Hoover [25], who showed that these problems are reducible to each other, and are in P-uniform TC^0, and Chiu, Davida, and Litow [52], who proved they are in L-uniform TC^0).

The basic theory of bounded arithmetic corresponding to TC^0 is the Zambella-style two-sorted theory VTC^0 introduced by Nguyen and Cook [133]. We may interpret provability in VTC^0 as a formalization of *feasible reasoning about elementary arithmetic operations* $+, \cdot, <$: what can we prove about them while only referring to concepts that do not exceed their complexity? (Note that we are concerned here with operations on *binary* integers, i.e., the second sort of VTC^0; operations on unary integers have much lower complexity.) More precisely, we ask what sentences in the basic language of arithmetic $\{+, \cdot, <\}$ are provable in VTC^0 if we interpret them over the binary integer sort. (This is a particular case of the $RSUV$ isomorphism.)

We are particularly interested if VTC^0 proves any nontrivial instances of induction for binary integers. (Of course, VTC^0 includes Σ^B_0-induction for unary integers, but a priori it does not seem to prove any induction on the second sort.) The specific question we tackle in Chapters VII and VIII is whether VTC^0 (or some extension thereof that still corresponds to TC^0) proves *open* (i.e., *quantifier-free*) *induction*, that is, the $RSUV$-translation of the theory *IOpen* introduced by Shepherdson [164].

We first observe that the provability of *IOpen* in VTC^0, even extended with arbitrary true universal (i.e., $\forall\Sigma^B_0$) sentences, has nontrivial computational consequences: if $f(X)$ is any polynomial (with integer or rational coefficients given by second-sort parameters), induction for the formula $f(X) < 0$ is a $\forall\Sigma^B_1$ statement, where the witness to the existential quantifier solves the following search problem: given a (fixed-degree) polynomial f and an integer $X > 0$ such that $f(0) < 0 \le f(X)$, find an integer $Y < X$ such that $f(Y) < 0 \le f(Y+1)$. Thus, if this instance of induction is provable in $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$, the corresponding search problem is computable by a TC^0 function. We can then easily manipulate it to obtain, for each constant d, a TC^0 *root approximation algorithm* for degree-d univariate polynomials: given such a polynomial f and a rational $\varepsilon > 0$, compute rational approximations within additive error ε of all real roots of f, or even Gaussian rational approximations of all complex roots of f. (One can show that this is, in fact, *equivalent* to provability of *IOpen* in $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$.)

Another way to look at it is to consider Shepherdson's criterion stating that (the non-negative part of) a discretely ordered ring D satisfies *IOpen* iff D is an integer part of a real-closed field, specifically of the real closure of (the fraction field of) D. Here, an integer part of a real-closed field R is a discrete subring $D \subseteq R$ such that every element of R is within distance 1 of D. Since elements of the real closure $\mathrm{rcl}(D)$ are just the "real" roots of polynomials from $D[X]$, this says that such roots can be arbitrarily well approximated in the fraction field of D, and witnessing functions for $\forall\exists$ statements expressing this property amount to constant-degree root approximation algorithms.

We tackle the computational complexity problem as a first step: in Chapter VII (originally published as [103]), we prove that TC^0 degree-d root approximation algorithms exist for any constant d. The argument uses tools from complex analysis. The basic idea is that if a is "close" to a root α of a polynomial f, then f has an analytic inverse function g on a neighbourhood of $f(a)$ including 0, and $g(0) = \alpha$. The coefficients of the power series of g can be determined by the *Lagrange inversion formula* (*LIF*), which makes them TC^0-computable, and then $g(0)$ can be approximated in TC^0 by computing a partial sum of the power series.

The exact meaning of a being "close" to α can be quantified using the Cauchy integral formula; the criterion is based on the ratio of $|a - \alpha|$ to the distance from a to the set of critical points of f (i.e., roots of f'). Using this, and bounds on the roots, we can set up a polynomial-size set of sample points a (in patterns centred around critical points of f, which can be approximated by induction on d) such that each root of f is close enough to some sample point; thus, locally inverting f (as explained above) near all sample points in parallel, we obtain a TC^0 algorithm that computes approximations of all roots of f.

The existence of such algorithms implies that the *RSUV* translation of *IOpen* is provable in $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$. The argument does not allow to restrict the usage of true universal statements to something provable in a reasonable fragment of bounded arithmetic, as it relied on fancy tools from complex analysis that would be difficult to even formulate, let alone prove, in bounded arithmetic.

One consequence of our results is that for any algebraic constant α, we can compute the nth digit of α in TC^0 given n in unary (hence we can compute it in the counting hierarchy CH when n is given in binary).

The provability of *IOpen* (and more) in a mild extension of VTC^0 is demonstrated in Chapter VIII (originally published as [104]). The reason it does not go through in VTC^0 proper is that we need iterated multiplication (and division) all over the place, but formalization of the Hesse, Allender, and Barrington algorithm is a serious problem in itself that's mostly tangential to the question of constant-degree root finding. Thus, we work in the theory $VTC^0 + IMUL$, where the *IMUL* axiom is a suitable formalization of the totality of iterated integer multiplication.

Again, one idea we use is to locally invert polynomials by power series whose coefficients are given by LIF. We can prove a suitable version of LIF in $VTC^0 + IMUL$ by direct manipulation of multinomial coefficients; in absence of other complex-analytic tools, this allows us to formalize root approximation for polynomials f such that, roughly speaking, the constant coefficient of f is very small w.r.t. the remaining coefficients.

We complement this with a model-theoretic argument based on properties of *valued fields*. Any ordered field F, such as the fraction field of a model M of arithmetic, carries a natural valuation; the completion $\hat{F}$ of F as a valued field coincides with the *Scott completion* of F, which is the largest ordered field extension of F in which F is dense. Using Shepherdson's criterion, $M \vDash \mathit{IOpen}$ iff $\hat{F}$ is a real-closed field. By basic properties of valued fields, one can show that F has a real-closed completion iff its value group is divisible, its residue field is real-closed, and F is *almost henselian* (i.e., all proper quotients of its valuation ring are henselian). In the case of F induced from a model of $\mathit{VTC}^0 + \mathit{IMUL}$, the divisibility of the value group is easy, and we can arrange the residue field to be $\mathbb{R}$ if the model is sufficiently saturated. Crucially, the condition of F being almost henselian follows from root approximation of polynomials with small constant coefficients that we proved earlier using LIF.

In this way, we prove IOpen in $\mathit{VTC}^0 + \mathit{IMUL}$. However, we can actually leverage the argument to get quite a bit more: we can formalize in $\mathit{VTC}^0 + \mathit{IMUL}$ a suitable version of a result of Mantzivis [127] on the structure of sets defined by sharply bounded (Σ^b_0) formulas, using root approximation for constant-degree polynomials for the base case of atomic formulas; thus, $\mathit{VTC}^0 + \mathit{IMUL}$ proves the RSUV translation of induction and minimization for Σ^b_0 formulas in Buss's language (and even in certain extensions of the language).

We remark that Jeřábek [106] recently succeeded to formalize a suitable version of the Hesse–Allender–Barrington algorithm in the base TC^0-theory VTC^0, showing that VTC^0 proves IMUL. Thus, by the results of Chapter VIII, the RSUV translation of Σ^b_0-MIN (including IOpen) is provable in VTC^0.

Chapter I

Approximate counting in bounded arithmetic

Abstract

We develop approximate counting of sets definable by Boolean circuits in bounded arithmetic using the dual weak pigeonhole principle ($sWPHP(PV)$), as a generalization of results from [93]. We discuss applications to formalization of randomized complexity classes (such as BPP, APP, MA, AM) in $PV_1 + sWPHP(PV)$.

1 Introduction

One of the most important aspects of bounded arithmetic is its close connection to computational complexity. There is a correspondence between arithmetical theories, and complexity classes: Buss's theories S^i_2 and T^i_2 [37] correspond to levels of the polynomial-time hierarchy, and various second-order theories were constructed for weak classes such as TC^0; Cook [65] presents a uniform way of constructing "minimal theories" associated to complexity classes below P. Consequently, fundamental problems from complexity theory are tied to similar questions about the arithmetical theories; for instance, the hierarchy of Buss's theories collapses if and only if bounded arithmetic proves the collapse of the polynomial hierarchy.

Our main motivation for studying approximate counting is the problem whether we can associate theories to randomized complexity classes, like BPP or AM. The problem is a loose research program rather than an exact question. On one hand, the concept of correspondence between theories and complexity classes does not admit a general definition; the way in which T^1_2 corresponds to P^{NP} is rather different from the correspondence of U^1 to NC. On the other hand, many probabilistic classes like BPP are "semantic classes", which means that attempts to characterize them as provably total functions of some kind in a recursively axiomatized theory are bound to failure. Nevertheless, we will try to provide evidence that $PV_1 + sWPHP(PV)$ (i.e., PV_1 extended by the dual (surjective) weak pigeonhole principle for poly-time computable functions) is the "right" theory for reasoning about randomized algorithms.

The connection of $sWPHP(PV)$ to probabilistic computation was first noticed by A. Wilkie, who proved that Σ^b_1-consequences of $S^1_2 + sWPHP(PV)$ are witnessed by TFRP-functions, and in particular, predicates provably Δ^b_1 in $S^1_2 + sWPHP(PV)$ are in ZPP (the result was published in Krajíček [116]). Jeřábek [93] considered the converse problem of formalizing probabilistic algorithms in $S^1_2 + sWPHP(PV)$, and introduced a way to define FRP-functions in $S^1_2 + sWPHP(PV)$ which covered at least the witnessing functions from Wilkie's theorem; however, the method used was seemingly ad hoc, and it was not clear how it could be generalized to other complexity classes like BPP.

In this paper, we will show that the dual weak pigeonhole principle is strong enough to provide a general method of approximating probabilities. More precisely, if X is a subset of an interval $[0, a)$ definable by a PV-formula, we can estimate $\Pr_{x<a}(x \in X)$ within a polynomially small error in $PV_1 + sWPHP(PV)$, and events of higher complexity can be dealt with by appropriate relativization. This allows us to treat various randomized classes like BPP, APP, AM, in a uniform and intuitive way—in fact, once we have a reasonable notion of (approximate) probability, the usual definitions of these classes can be formalized almost literally. As we have already mentioned, provably total functions are not an appropriate standard for establishing correspondence of theories to probabilistic complexity classes: for semantic classes there is no hope, and as we will see, for syntactic classes the problem is either meaningless or trivial (with the notable exception of APP). Instead, we will show that $PV_1 + sWPHP(PV)$ proves basic properties of the relevant probabilistic algorithms, such as amplification of success, or simulation of randomness by nonuniformity.

Estimating probabilities in uniform distributions is only a fancy name for approximate counting of bounded sets. Approximate counting has other applications besides randomized algorithms; most importantly, counting arguments are often used to prove various combinatorial theorems. We will provide basic counting tools like the inclusion-exclusion principle, but the overall utility of our methods in this area seems rather limited. Proofs of combinatorial statements such as the Ramsey theorem or the tournament principle typically rely on counting of sparse sets, which is impossible in our setup. We can only approximate the size of a set $X \subseteq [0, 2^n)$ within a polynomial fraction of 2^n, whereas here we would need to approximate it within a polynomial fraction of $|X|$.

The paper is organized as follows. In Section 2 we provide elementary background on basic arithmetic, and fix notational conventions. In Section 3 we introduce approximate counting of sets defined by circuits in $PV_1 + sWPHP(PV)$, and formalize a toolbox of counting principles. In Section 4 we discuss in detail the development of several randomized complexity classes (FRP, BPP, APP, MA, and promise variants) in $PV_1 + sWPHP(PV)$. In Section 5 we indicate how to relativize our approach, and we discuss the class AM.

2 Preliminaries

We assume some degree of familiarity with first-order bounded arithmetic, however the basic definitions are summarized below. More background can be found in [116, 40, 86].

Buss's S^i_2 and T^i_2 [37] are first-order theories with equality in the language $L = \langle 0, S, +, \cdot, \leq, \#, |x|, \lfloor \frac{x}{2} \rfloor \rangle$, where the function $|x|$ is intended to designate $\lceil \log_2(x + 1) \rceil$ (the number of digits in the binary representation of x), and $x \# y$ is $2^{|x|\cdot|y|}$. *Bounded quantifiers* are expressions of the form

$$\exists x \le t \ldots := \exists x\,(x \le t \wedge \ldots),$$
$$\forall x \le t \ldots := \forall x\,(x \le t \to \ldots),$$

where t is a term without an occurrence of x. A bounded quantifier is *sharply bounded*, if t has the form $|s|$ for some term s. A formula φ is sharply bounded, if all quantifiers in φ are sharply bounded. The hierarchy of Σ^b_i- and Π^b_i-formulas is defined inductively: $\Sigma^b_0 = \Pi^b_0$ is the set of sharply bounded formulas, Σ^b_{i+1} is the closure of Π^b_i under bounded existential and sharply bounded universal quantifiers, and Π^b_{i+1} is the closure of Σ^b_i under bounded universal and sharply bounded existential quantifiers. Bounded formulas capture the polynomial-time hierarchy (PH). More precisely, for any $i \ge 1$ the class Σ^P_i coincides with sets of natural numbers definable by Σ^b_i-formulas in $\mathbb{N}$ (the standard model of arithmetic), and dually $\Pi^P_i = \Pi^b_i(\mathbb{N})$, in particular $\mathrm{NP} = \Sigma^b_1(\mathbb{N})$.

The theory S^i_2 consists of a finite list of open axioms denoted by *BASIC*, and the polynomial induction schema

$$(\Sigma^b_i\text{-}\mathit{PIND}) \qquad \varphi(0) \wedge \forall x \le a\,(\varphi(\lfloor \tfrac{x}{2} \rfloor) \to \varphi(x)) \to \varphi(a),$$

where $\varphi \in \Sigma^b_i$. The theory T^i_2 is axiomatized by *BASIC* and the induction schema

$$(\Sigma^b_i\text{-}\mathit{IND}) \qquad \varphi(0) \wedge \forall x \le a\,(\varphi(x) \to \varphi(x+1)) \to \varphi(a).$$

PV is a purely equational theory introduced by Cook [63]. Its language contains a few basic function symbols, and it is inductively expanded by symbols for functions defined from previously introduced functions by composition, and limited recursion on notation. *PV* is axiomatized by equations defining all the function symbols, and a derivation rule similar to open *PIND*. In the standard model, *PV*-functions define exactly the class of polynomial-time computable functions (FP). We will slightly abuse the notation and denote by *PV* also the language of *PV* (the set of all *PV*-functions).

PV_1 (also called *QPV*) is an extension of *PV* to first-order logic [121, 39, 64]. It has an axiomatization by purely universal sentences, and it is conservative over *PV*. The hierarchy of $\Sigma^b_i(PV)$- and $\Pi^b_i(PV)$-formulas is defined similarly to Σ^b_i and Π^b_i, but in the language of *PV*. PV_1 proves *IND* and *PIND* for $\Sigma^b_0(PV)$-formulas.

$S^1_2(PV)$ is the combination of S^1_2 and PV_1: i.e., it has the language of *PV*, and it is axiomatized by *PV* and $\Sigma^b_1(PV)$-*PIND*. All *PV*-functions have well-behaved provably total Δ^b_1-definitions in S^1_2; it follows that $S^1_2(PV)$ is an extension of S^1_2 by definitions, and in particular, $S^1_2(PV)$ is conservative over S^1_2. Thus there is little practical difference between S^1_2 and $S^1_2(PV)$, and we will simply identify these two theories. Buss's witnessing theorem [37] implies that S^1_2 is Σ^b_1-conservative over PV_1, and in fact, we may identify PV_1 with $\forall\Sigma^b_1(S^1_2)$.

The theories PV_{i+1} for $i > 0$, introduced in [121], are defined similarly to PV_1, except that the basic functions of their language include the characteristic functions of all Σ^b_i-predicates, thus PV_{i+1}-functions correspond to $\mathrm{FP}^{\Sigma^P_i}$ in the standard model. PV_{i+1} is a conservative extension of T^i_2 (contrary to popular belief, essentially the same also holds for $i = 0$ [94]), and

$S^1_2(PV_{i+1})$ is a conservative extension of S^{i+1}_2. S^{i+1}_2 is Σ^b_{i+1}-conservative over PV_{i+1} and T^i_2 by Buss's witnessing theorem.

All these theories can be *relativized*. We consider the language $L(\alpha) = L \cup \{\alpha\}$, where α is a new predicate, and define $\Sigma^b_i(\alpha)$ and $\Pi^b_i(\alpha)$ in the same way as Σ^b_i and Π^b_i, but extended to the new language. The theories $S^i_2(\alpha)$ and $T^i_2(\alpha)$ are axiomatized by $BASIC$ and $\Sigma^b_i(\alpha)$-$PIND$ resp. $\Sigma^b_i(\alpha)$-IND, with no other axioms about α. $PV(\alpha)$ and $PV_i(\alpha)$ can be defined similarly (the characteristic function of α is allowed to appear in functions constructed by limited recursion on notation). $PV(\alpha)$-functions correspond to polynomial-time algorithms with an oracle. We write φ^α and f^α when we want to stress the dependence of an $L(\alpha)$-formula or $PV(\alpha)$-function on α; in that case, φ^ψ or f^ψ denotes the result of substitution of a formula ψ for α. We may generalize $L(\alpha)$ by allowing an arbitrary set of new predicates and function symbols instead of α; in the case of functions, we have to include axioms enforcing an explicit polynomial bound on the length of the output of the function.

For any function f we define the formula

$$sPHP^x_y(f) := \exists v < y \, \forall u < x \, f(u) \neq v,$$

where f may involve other parameters not explicitly shown. The *dual* (or *surjective*) *weak pigeonhole principle* for f, written as $sWPHP(f)$, is the universal closure of the formula

$$x > 0 \to sPHP^{x|y|}_{x(|y|+1)}(f),$$

and if Γ is a set of functions, $sWPHP(\Gamma)$ denotes the schema $\{sWPHP(f) \mid f \in \Gamma\}$. We will mostly work with $sWPHP(PV)$, i.e., the dual weak pigeonhole principle for poly-time functions. $sWPHP(PV)$ is over S^1_2 equivalent to the more usual schema

$$x > 1 \to sPHP^x_{x^2}(f),$$

but it is not clear whether this reduction also works over PV_1. $sWPHP(PV)$ is provable in T^2_2 [146, 116, 125], but $sWPHP(\alpha)$ is not provable in $S^2_2(\alpha)$ [160]. The schema $sWPHP(PV)$ is finitely axiomatizable: PV_1 proves that any PV-function is computable by a poly-size circuit on any bounded domain, thus $sWPHP(PV)$ is equivalent to its instance $sWPHP(\mathrm{eval})$, where $\mathrm{eval}(C, x)$ is a two-place PV-function which evaluates a circuit C on an input x.

We will often work with *bounded definable sets*, which are collections of numbers of the form

$$X = \{x < a \mid \varphi(x)\},$$

where φ is a formula. Bounded sets are *not* genuine objects in our arithmetical theories, but a figure of speech: $x \in X$ is an abbreviation for $x < a \wedge \varphi(x)$. When used in a context which asks for a set, a number a is assumed to represent the integer interval $[0, a)$; thus, for example, $X \subseteq a$ means that all elements of X are less than a. We will use simple set-theoretic operations, whose meaning should be generally clear from the context; for example, if $X \subseteq a$ and $Y \subseteq b$, we may define

$$X \times Y := \{bx + y \mid x \in X, y \in Y\} \subseteq ab,$$
$$X \mathbin{\dot\cup} Y := X \cup \{y + a \mid y \in Y\} \subseteq a + b.$$

The sets we will encounter most often will be defined by *Boolean circuits*: a circuit $C\colon 2^n \to 2$ defines the set $\{x < 2^n \mid C(x) = 1\}$. (Here again, 2^n denotes the interval $[0, 2^n)$, which may be identified with the set of binary strings of length n; thus C is a circuit with n Boolean input variables.)

We will use the shorthand notation

$$x \in \mathrm{Log} \leftrightarrow \exists y\, x = |y|,$$
$$x \in \mathrm{LogLog} \leftrightarrow \exists y\, x = ||y||.$$

If f is a function of two variables, $f(a, -)$ denotes the function of one variable which results from f by fixing its first argument to a. The set of natural numbers will be denoted by ω (in the metatheory).

We will also work with rational numbers in PV, which are assumed to be represented by pairs of integers in the natural way. The expression $x^{-1} \in \mathrm{Log}$ is a shorthand notation meaning that x is a positive rational number, whose inverse is bounded from above by a natural number $n \in \mathrm{Log}$.

Many of our results take place *inside* formal theories like $PV_1 + sWPHP(PV)$. If T is a theory, a parenthesized expression "in T" after the heading of a definition or theorem indicates that the definition is introduced in T, or that the theorem is formulated and proved inside T. However, we will slightly abuse this convention for reasons of compactness: when we write e.g. "for every PV-function f ..." in a formalized context, it is assumed that the quantification over PV-functions takes place in the metatheory, and only *parameters* of the function are quantified inside T. Formulas, definable sets, and other non-first-order objects are treated similarly. Expressions like "a pair of PV-functions $\langle f, g\rangle$" also fit in this category; inside T, no actual pairing operation is involved.

3 Counting

Our definition of approximate counting in bounded arithmetic is based on the following observation: if X and Y are sets, and there exists a circuit which maps X onto Y, then the cardinality of Y is at most the cardinality of X. We need to make sure that such a definition is well-behaved, i.e., that it satisfies common properties we expect from a cardinality function. In particular, it is conceivable that a large but complicated set X cannot be disentangled by a polynomial-size circuit and mapped onto an interval $[0, s)$ approaching its size; we must show that such cases do not happen. The natural way to guarantee sufficient precision of these counting circuits is to consider a two-sided comparison: if we find a mapping of X onto $[0, s - e)$, and a mapping of $[0, s + e)$ onto X, we know that the size of X is s within error e.

It turns out that an extra complication is necessary: rather than mapping X onto Y directly, we will take several copies of both sets, i.e., map $v \times X$ onto $v \times Y$ for some $v > 0$. With this modification, we are able to prove in $PV_1 + sWPHP(PV)$ that there exists a pair of counting circuits which estimates the size of X within a polynomially small error (relative to the size of the ambient interval containing X), for any X defined by a circuit. We will construct such counting circuits by analysis of the Nisan-Wigderson pseudorandom generator [135]; formalization of the

Nisan-Wigderson generator in $S^1_2 + sWPHP(PV)$ was already considered in [93] for a different goal. We start by overview of the relevant concepts.

Definition 3.1 (in PV_1) Let $f\colon 2^k \to 2$ be a truth-table of a Boolean function (f is encoded as a string of 2^k bits, hence $k \in \mathrm{LogLog}$). We say that f is *(worst-case) ε-hard,* written as $\mathrm{Hard}_\varepsilon(f)$, if there does not exist a circuit C of size at most $2^{\varepsilon k}$ which computes f. The function f is *average-case ε-hard,* written as $\mathrm{Hard}^A_\varepsilon(f)$, if there does not exist a circuit C of size at most $2^{\varepsilon k}$ such that

$$\bigl|\{u < 2^k \mid C(u) = f(u)\}\bigr| \geq \bigl(\tfrac{1}{2} + 2^{-\varepsilon k}\bigr)2^k.$$

Notice that $\mathrm{Hard}_\varepsilon(f)$ and $\mathrm{Hard}^A_\varepsilon(f)$ are Π^b_1-formulas.

Lemma 3.2 ([93]) *For every constant $\varepsilon < 1/3$ there exists a constant c such that $PV_1 + sWPHP(PV)$ proves: for every $k \in \mathrm{LogLog}$ such that $k \geq c$, there exist average-case ε-hard functions $f\colon 2^k \to 2$.*

Moreover, there exists a PV-function $g\colon 2^{n-m} \to 2^n$ such that any $f < 2^n$ outside the range of g is average-case ε-hard, where $n = 2^k$, and $m \geq n^{1-2\varepsilon}$.

Definition 3.3 ([135]) (in PV_1) Let $k, \ell, t, m \in \mathrm{Log}$, $k \leq \ell \leq t$. A *$\langle k, \ell, t, m\rangle$-design* is a sequence $\langle S_i\rangle_{i<m}$ of subsets $S_i \subseteq t$, such that $|S_i| = \ell$ and $|S_i \cap S_j| \leq k$ for all $i < j < m$.

Lemma 3.4 ([93]) *Let $0 < \gamma < 1$. There are constants $\delta > 0$, $c > 1$, and a PV-function d such that*

$$PV_1 \vdash d(x) \text{ is a } \langle \gamma\ell, \ell, c\ell, 2^{\delta\ell}\rangle\text{-design, where } \ell = ||x||.$$

Definition 3.5 ([135]) (in PV_1) Let $x < 2^t$, and $X \subseteq t$, $|X| = \ell$. Let $\{s_i\}_{i<\ell}$ be the increasing enumeration of the set X. Then we put $x \restriction X := y$, where $y < 2^\ell$ and $\mathrm{bit}(y, i) = \mathrm{bit}(x, s_i)$ for all $i < \ell$.

If $f\colon 2^\ell \to 2$, and $S = \langle S_i\rangle_{i<m}$ is a $\langle k, \ell, t, m\rangle$-design, the *Nisan-Wigderson generator* is a function $NW_{f,S}\colon 2^t \to 2^m$ defined by

$$\mathrm{bit}(NW_{f,S}(x), i) = f(x \restriction S_i).$$

Definition 3.6 (in PV_1) We adopt a few conventions on functions computed by circuits. Let $C\colon 2^n \to 2^m$ be a circuit, and X and Y definable sets. We say that C *computes a function from X to Y*, written as

$$C\colon X \to Y,$$

if $X \subseteq 2^n$, $Y \subseteq 2^m$, and $C[X] \subseteq Y$. We write

$$C\colon X \hookrightarrow Y$$

if, in addition, the function computed by C is injective on X.

We write

$$C\colon X \twoheadrightarrow Y$$

if $X \subseteq 2^n$, $Y \subseteq 2^m$, and $C[X] \supseteq Y$. Notice that this does *not* imply $C\colon X \to Y$. An equivalent condition is $C\colon X' \to Y$ and $C[X'] = Y$ for some $X' \subseteq X$.

(This way of introducing $\twoheadrightarrow$ is mostly a technicality, needed to overcome the annoying fact that a non-empty set cannot be mapped onto the empty set.)

We are ready for the main theorem of this section, which guarantees the existence of suitable counting circuits. It is an extension of [93, Prop. 4.7].

Theorem 3.7 (*in $PV_1 + sWPHP(PV)$*) *Let $C\colon 2^n \to 2$ be a Boolean circuit, and $\varepsilon^{-1} \in \mathrm{Log}$. Denote*

$$X := \{x < 2^n \mid C(x) = 1\}.$$

There exist $s \le 2^n$, $v \le \mathrm{poly}(n\varepsilon^{-1}|C|)$, and circuits G_ξ, H_ξ, $\xi = 0, 1$, of size $\mathrm{poly}(n\varepsilon^{-1}|C|)$ such that

$$\begin{aligned} &G_0\colon v(s + \varepsilon 2^n) \twoheadrightarrow v \times X && H_0\colon v \times X \hookrightarrow v(s + \varepsilon 2^n) \\ &G_1\colon v \times (X \mathbin{\dot\cup} \varepsilon 2^n) \twoheadrightarrow vs && H_1\colon vs \hookrightarrow v \times (X \mathbin{\dot\cup} \varepsilon 2^n) \end{aligned}$$

and such that

$$G_\xi \circ H_\xi = \mathrm{id}$$

on their respective domains.

Proof: Let δ and c be the constants from Lemma 3.4 for $\gamma := 1/12$. Put

$$\ell := \max\bigl\{4|n\varepsilon^{-1}|, 12|n|, \tfrac{1}{\delta}|n|, 4(||C|| + 1)\bigr\},$$

and $k := \gamma\ell$, $t := c\ell$, $v := 2^t$. As $n \le 2^{\delta\ell}$, there exists a $\langle k, \ell, t, n\rangle$-design $S = \langle S_0, \dots, S_{n-1}\rangle$. By Lemma 3.2, there exists an average-case 1/4-hard Boolean function $f\colon 2^\ell \to 2$. We define

$$\begin{aligned} Y &:= \{x < 2^t \mid C(NW_{f,S}(x)) = 1\}, \\ s &:= 2^{n-t}|Y|. \end{aligned}$$

(We may count $|Y|$ directly, as $t \in \mathrm{LogLog}$.)

For any $i \le n$, we define

$$M_i = \{\langle \vec r, x\rangle \in 2^n \times 2^t \mid C(f(x \restriction S_0), \dots, f(x \restriction S_{i-1}), r_i, \dots, r_{n-1}) = 1\}.$$

Notice that $M_0 = X \times 2^t$, and $M_n = 2^n \times Y$. Suppose we find a sequence of circuits $G_{\xi,i}$, $H_{\xi,i}$, where $\xi = 0, 1$ and $i < n$, such that

$$\begin{aligned} &G_{0,i}\colon M_{i+1} \mathbin{\dot\cup} (i+1)a2^{n+t-\ell} \twoheadrightarrow M_i \mathbin{\dot\cup} ia2^{n+t-\ell} \\ &H_{0,i}\colon M_i \mathbin{\dot\cup} ia2^{n+t-\ell} \hookrightarrow M_{i+1} \mathbin{\dot\cup} (i+1)a2^{n+t-\ell} \\ &G_{1,i}\colon M_i \mathbin{\dot\cup} (n-i)a2^{n+t-\ell} \twoheadrightarrow M_{i+1} \mathbin{\dot\cup} (n-i-1)a2^{n+t-\ell} \\ &H_{1,i}\colon M_{i+1} \mathbin{\dot\cup} (n-i-1)a2^{n+t-\ell} \hookrightarrow M_i \mathbin{\dot\cup} (n-i)a2^{n+t-\ell} \\ &G_{\xi,i} \circ H_{\xi,i} = \mathrm{id} \end{aligned}$$

where $a = 2^{3\ell/4}$. Then we can define

$$G_0 = G_{0,0} \circ G_{0,1} \circ \cdots \circ G_{0,n-1} \qquad H_0 = H_{0,n-1} \circ H_{0,n-2} \circ \cdots \circ H_{0,0}$$
$$G_1 = G_{1,n-1} \circ G_{1,n-2} \circ \cdots \circ G_{1,0} \qquad H_1 = H_{1,0} \circ H_{1,1} \circ \cdots \circ H_{1,n-1}$$

Notice that $v\varepsilon 2^n \geq na2^{n+t-\ell}$, as $n\varepsilon^{-1} \leq 2^{\ell/4}$. For any $x \in X \times 2^t$ and $y \in 2^n \times Y$, we can show

$$((G_{0,0} \circ G_{0,1} \circ \cdots \circ G_{0,i}) \circ (H_{0,i} \circ \cdots \circ H_{0,1} \circ H_{0,0}))(x) = x,$$
$$((G_{1,n-1} \circ G_{1,n-2} \circ \cdots \circ G_{1,n-i}) \circ (H_{1,n-i} \circ \cdots \circ H_{1,n-2} \circ H_{1,n-1}))(y) = y$$

by straightforward induction on i, in particular $G_\xi \circ H_\xi = \mathrm{id}$, which also implies that G_ξ are surjective, and H_ξ are injective.

It thus suffices to construct $G_{\xi,i}$ and $H_{\xi,i}$. There exists an easily computable bijection between pairs $\langle y, u\rangle \in 2^{t-\ell} \times 2^\ell$, and numbers $x \in 2^t$, so that x maps to $\langle x \restriction (t \smallsetminus S_i), x \restriction S_i)\rangle$. If $j < n$, $y < 2^{t-\ell}$, $u < 2^\ell$, and $x < 2^t$ is such that $\langle y, u\rangle = \langle x \restriction (t \smallsetminus S_i), x \restriction S_i)\rangle$, we define $f_j^{i,y}(u) = f(x \restriction S_j)$. Notice that $f_i^{i,y}(u) = f(u)$. Then

$$M_i \approx 2^i \times M_i',$$

where

$$M_i' := \{\langle r_{i+1}, \ldots, r_{n-1}, y, r, u\rangle \in 2^{n-i-1} \times 2^{t-\ell} \times 2 \times 2^\ell \mid$$
$$C(f_0^{i,y}(u), \ldots, f_{i-1}^{i,y}(u), r, r_{i+1}, \ldots, r_{n-1}) = 1\},$$

and $A \approx B$ means that there exists a bijection g of A onto B such that g and g^{-1} are computable by a polynomial-size circuit. In a similar way we have

$$M_{i+1} \approx 2^i \times M_{i+1}',$$

where

$$M_{i+1}' := \{\langle r_{i+1}, \ldots, r_{n-1}, y, r, u\rangle \mid C(f_0^{i,y}(u), \ldots, f_{i-1}^{i,y}(u), f(u), r_{i+1}, \ldots) = 1\}.$$

Fix $y < 2^{t-\ell}$, and $r_{i+1}, \ldots, r_{n-1} < 2$. Define

$$\begin{aligned}
U^{\vec{r},y} &:= \{\langle r, u\rangle \in 2 \times 2^\ell \mid C(f_0^{i,y}(u), \ldots, f_{i-1}^{i,y}(u), r, r_{i+1}, \ldots, r_{n-1}) = 1\} \\
&= \{\langle r, u\rangle \in 2 \times 2^\ell \mid \langle \vec{r}, y, r, u\rangle \in M_i'\}, \\
V^{\vec{r},y} &:= \{\langle r, u\rangle \in 2 \times 2^\ell \mid C(f_0^{i,y}(u), \ldots, f_{i-1}^{i,y}(u), f(u), r_{i+1}, \ldots, r_{n-1}) = 1\} \\
&= \{\langle r, u\rangle \in 2 \times 2^\ell \mid \langle \vec{r}, y, r, u\rangle \in M_{i+1}'\}, \\
A_\eta(u) &:= C(f_0^{i,y}(u), \ldots, f_{i-1}^{i,y}(u), \eta, r_{i+1}, \ldots, r_{n-1}),
\end{aligned}$$

where $\eta < 2$. As $\ell \in \mathrm{LogLog}$, we can directly count the sets $U^{\vec{r},y}$ and $V^{\vec{r},y}$; an easy calculation shows

$$\begin{aligned}|V^{\vec{r},y}| - |U^{\vec{r},y}| &= 2\big|\{u \mid f(u) \wedge A_1(u)\}\big| + 2\big|\{u \mid \neg f(u) \wedge A_0(u)\}\big| \\ &\quad - \big|\{u \mid A_1(u)\}\big| - \big|\{u \mid A_0(u)\}\big| \\ &= \big|\{u \mid f(u) \wedge A_1(u)\}\big| - \big|\{u \mid \neg f(u) \wedge A_1(u)\}\big| \\ &\quad + \big|\{u \mid \neg f(u) \wedge A_0(u)\}\big| - \big|\{u \mid f(u) \wedge A_0(u)\}\big| \\ &= \big|\{u \mid f(u) \wedge A_1(u)\}\big| + \big|\{u \mid \neg f(u) \wedge \neg A_1(u)\}\big| \\ &\quad + \big|\{u \mid \neg f(u) \wedge A_0(u)\}\big| + \big|\{u \mid f(u) \wedge \neg A_0(u)\}\big| - 2^\ell \\ &= \big|\{u \mid f(u) \leftrightarrow A_1(u)\}\big| - \big|\{u \mid f(u) \leftrightarrow A_0(u)\}\big|\end{aligned}$$

On the other hand, for any $j \neq i$, $f_j^{i,y}(u)$ depends only on $|S_i \cap S_j| \leq k$ variables of u, and is thus computable by a circuit of size 2^k. Therefore, A_η and $\neg A_\eta$ are computable by circuits of size at most

$$1 + |C| + i2^k \leq |C| + n2^k \leq 2^{\ell/4-1} + 2^{\ell/12}2^{\ell/12} \leq 2^{\ell/4}.$$

As f is average-case 1/4-hard, we have

$$\big||\{u \mid A_\eta(u) = f(u)\}| - 2^{\ell-1}\big| \leq 2^{\ell-\ell/4} = a,$$

thus

$$\big||V^{\vec{r},y}| - |U^{\vec{r},y}|\big| \leq 2a.$$

We may arrange the sets $U^{\vec{r},y}$ and $V^{\vec{r},y}$ in increasing sequences, match their initial parts, and pad to get functions

$$\begin{aligned} &g_0^{\vec{r},y} \colon U^{\vec{r},y} \dot\cup 2a \twoheadrightarrow V^{\vec{r},y} & &h_0^{\vec{r},y} \colon V^{\vec{r},y} \hookrightarrow U^{\vec{r},y} \dot\cup 2a \\ &g_1^{\vec{r},y} \colon V^{\vec{r},y} \dot\cup 2a \twoheadrightarrow U^{\vec{r},y} & &h_1^{\vec{r},y} \colon U^{\vec{r},y} \hookrightarrow V^{\vec{r},y} \dot\cup 2a\end{aligned}$$

such that $g_\xi^{\vec{r},y} \circ h_\xi^{\vec{r},y} = \mathrm{id}$. As this construction is uniform in $\vec{r}$ and y, we may construct polynomial-size circuits

$$\begin{aligned} &G'_0 \colon M'_i \dot\cup a2^{n-i+t-\ell} \twoheadrightarrow M'_{i+1} & &H'_0 \colon M'_{i+1} \hookrightarrow M'_i \dot\cup a2^{n-i+t-\ell} \\ &G'_1 \colon M'_{i+1} \dot\cup a2^{n-i+t-\ell} \twoheadrightarrow M'_i & &H'_1 \colon M'_i \hookrightarrow M'_{i+1} \dot\cup a2^{n-i+t-\ell}\end{aligned}$$

and from these we obtain $G_{\xi,i}$, $H_{\xi,i}$ as required. □

We formally introduce the concept of approximate size comparison, as described in the introductory paragraph of this section. Notice that the definition applies to a more general situation than what is permitted by Theorem 3.7. The main reason is that we will occasionally need to express that a set is exponentially small, even though Theorem 3.7 cannot provide counting with exponential precision.

Definition 3.8 (in $PV_1 + sWPHP(PV)$) Let $X, Y \subseteq 2^n$ be definable sets, and $\varepsilon \leq 1$. We say that *the size of X is approximately less than the size of Y with error ε*, written as

$$X \preceq_\varepsilon Y,$$

if there exists a circuit G, and $v \neq 0$, such that

$$G\colon v \times (Y \dot\cup \varepsilon 2^n) \twoheadrightarrow v \times X.$$

The sets X and Y have *approximately the same size with error* ε, written as

$$X \approx_\varepsilon Y,$$

if $X \preceq_\varepsilon Y$ and $Y \preceq_\varepsilon X$.

We recall that we identify a number s with the interval $[0, s)$, thus as a special case, $X \approx_\varepsilon s$ means that the size of X is equal to s with error ε.

Remark 3.9 In this definition, "error ε" is somewhat a misnomer. The counting is not exact even if we take $\varepsilon = 0$, there is always some error present due to the fact that only the weak pigeonhole principle is available. In fact, we will often conveniently use $\preceq_0$ for approximate size comparisons.

The lemma below summarizes elementary properties of Definition 3.8.

Lemma 3.10 (*in PV_1*) *Let $X, Y, X', Y', Z \subseteq 2^n$ and $W, W' \subseteq 2^m$ be definable sets, and $\varepsilon, \delta \leq 1$.*

(i) $X \preceq_\varepsilon Y$, $\varepsilon \leq \delta \Rightarrow X \preceq_\delta Y$.

(ii) $X \subseteq Y \Rightarrow X \preceq_0 Y$.

(iii) $X \preceq_\varepsilon Y$, $Y \preceq_\delta Z \Rightarrow X \preceq_{\varepsilon+\delta} Z$.

(iv) *If $X \preceq_\varepsilon X'$, $Y \preceq_\delta Y'$, and X' and Y' are separable by a circuit, then $X \cup Y \preceq_{\varepsilon+\delta} X' \cup Y'$.*

(v) $X \preceq_\varepsilon X'$, $W \preceq_\delta W' \Rightarrow X \times W \preceq_{\varepsilon+\delta+\varepsilon\delta} X' \times W'$.

Proof: Exercise. □

The next lemma exploits consequences of Theorem 3.7.

Lemma 3.11 (*in $PV_1 + sWPHP(PV)$*) *Let $X, Y \subseteq 2^n$ be definable by circuits, $s, t, u \leq 2^n$, $\varepsilon, \delta, \eta, \xi \leq 1$, $\xi^{-1} \in$* Log.

(i) *There exists $s \leq 2^n$ such that $X \approx_\xi s$.*

(ii) $s \preceq_\varepsilon X \preceq_\delta t \Rightarrow s \leq t + (\varepsilon + \delta + \xi) 2^n$.

(iii) $X \preceq_\xi Y$ *or* $Y \preceq_\xi X$.

(iv) $X \preceq_\varepsilon Y \Rightarrow 2^n \smallsetminus Y \preceq_{\varepsilon+\xi} 2^n \smallsetminus X$.

(v) $X \approx_\varepsilon s$, $Y \approx_\delta t$, $X \cap Y \approx_\eta u \Rightarrow X \cup Y \approx_{\varepsilon+\delta+\eta+\xi} s + t - u$.

Proof:

(i) follows from Theorem 3.7.

(ii): by transitivity, it suffices to show that $s \preceq_0 t$ implies $s \le t + \xi 2^n$, which follows from $sWPHP(PV)$.

(iii) follows from (i) and the linearity of $\le$.

(iv): let $\zeta = \xi/11$, and choose s, t, s', t' such that $X \approx_\zeta s$, $Y \approx_\zeta t$, $2^n \smallsetminus X \approx_\zeta s'$, $2^n \smallsetminus Y \approx_\zeta t'$. We have $s \le t + (\varepsilon + 3\zeta)2^n$ by (ii). As $t + t' \preceq_{2\zeta} 2^n$ by Lemma 3.10 (iv), we have also $t' \le 2^n - t + 3\zeta 2^n$ by (ii), and in a similar way, $2^n - s \le s' + 3\zeta 2^n$. This implies $t' \le s' + (\varepsilon + 9\zeta)2^n$, thus $2^n \smallsetminus Y \preceq_{\varepsilon + 11\zeta} 2^n \smallsetminus X$.

(v): fix r such that $X \smallsetminus Y \approx_{\xi/2} r$. By Lemma 3.10 (iv), we have $X \approx_{\eta + \xi/2} r + u$, and $X \cup Y \approx_{\delta + \xi/2} r + t$. The former implies $s \approx_{\varepsilon + \eta + \xi/2} r + u$, thus $s + t - u \approx_{\varepsilon + \eta + \xi/2} r + t$, and $s + t - u \approx_{\varepsilon + \delta + \eta + \xi} X \cup Y$. □

The definition of $\preceq_\varepsilon$ is problematic, if we wish to use it in induction formulas in more sophisticated arguments. As it stands, it is an unbounded $\exists\Pi^b_2$-formula; even if we restrict its usage to the case covered by Theorem 3.7, and include the relevant bounds, we cannot do much better than Σ^b_2. We can solve this problem by working in a suitable conservative extension of $PV_1 + sWPHP(PV)$, introduced in [93].

Definition 3.12 The theory $HARD^A$ is an extension of $PV_1(\alpha) + sWPHP(PV(\alpha))$ by the axioms

$$\alpha(x) \text{ is a truth-table of a Boolean function in } ||x|| \text{ variables,}$$
$$x \ge c \to \mathrm{Hard}^A_{1/4}(\alpha(x)),$$
$$||x|| = ||y|| \to \alpha(x) = \alpha(y),$$

where c is the constant from Lemma 3.2.

Theorem 3.13 *$HARD^A$ is a conservative extension of $PV_1 + sWPHP(PV)$. More generally, for any $i \ge 1$, $HARD^A + S^i_2(\alpha)$ and $HARD^A + T^i_2(\alpha)$ are conservative extensions of $S^i_2 + sWPHP(PV)$ and $T^i_2 + sWPHP(PV)$, respectively.*

Proof: This was shown in [93] with S^1_2 as a base theory. It is easy to modify the proof so that it works over PV_1. □

We note that the axiom $sWPHP(PV(\alpha))$ is redundant in $HARD^A + S^1_2(\alpha)$; i.e., the existence of functions hard on average implies $sWPHP(PV)$ over S^1_2 [93]. We do not know whether this also holds over PV_1.

Lemma 3.14 *There is a $PV(\alpha)$-function* Size *such that $HARD^A$ proves: if $X \subseteq 2^n$ is definable by a circuit C, then*

$$X \approx_\varepsilon \mathrm{Size}(C, 2^n, e),$$

where $\varepsilon = |e|^{-1}$. The "witnessing circuits" G_ξ, H_ξ from Theorem 3.7 are also constructible by $PV(\alpha)$-functions.

Proof: By inspection of the proof of Theorem 3.7, we see that the only non-uniformity was in the choice of the hard function f. $\square$

We will abuse the notation and write $\mathrm{Size}(X, \varepsilon)$ instead of $\mathrm{Size}(C, 2^n, e)$.

The advantage of $HARD^A$ is that the complexity of approximate counting drops from Σ^b_2 to $PV(\alpha)$, which means that we can use approximate counting freely in induction, and we can count parametric families of sets uniformly. Some of the results below illustrate these techniques. We begin by showing that the size of the disjoint union of a sequence of sets is the sum of sizes of the sets.

Proposition 3.15 (Disjoint union) (*in $PV_1 + sWPHP(PV)$*) *Let $\{X_i \mid i < m\}$ be subsets of 2^n, defined by a sequence of circuits. Let $\varepsilon, \xi \leq 1$, $\xi^{-1} \in \mathrm{Log}$, and $\{s_i \mid i < m\}$ a sequence of numbers such that $X_i \preceq_\varepsilon s_i$ for every $i < m$. Then*

$$\dot{\bigcup_{i<m}} X_i \preceq_{\varepsilon+\xi} \sum_{i<m} s_i,$$

where the disjoint sum $\dot\bigcup_{i<m} X_i := \bigcup_{i<m}(X_i \times \{i\}) \subseteq 2^n \times m$ is considered a subset of $2^{n+|m|}$.

The same holds for $\succeq$ in place of $\preceq$.

Proof: We may work in $HARD^A$ by Theorem 3.13. First, notice that the error in $\preceq$ is relative to the ambient set size, thus if we reconsider X_i as a subset of $2^n \times m$, we have $X_i \preceq_{\varepsilon/m} s_i$. Put $\zeta = \xi/(3m+1)$. We will show

$$\mathrm{Size}\Big(\dot{\bigcup_{i<k}} X_i, \zeta\Big) \leq \sum_{i<k} s_i + (\varepsilon/m + 3\zeta)k$$

by induction on $k \leq m$. Assume that the statement is true for k. We have

$$\dot{\bigcup_{i<k}} X_i \approx_\zeta \mathrm{Size}\Big(\dot{\bigcup_{i<k}} X_i, \zeta\Big) \preceq_\delta \sum_{i<k} s_i,$$

where $\delta = (\varepsilon/m + 3\zeta)k$. As $X_k \preceq_{\varepsilon/m} s_k$, we obtain

$$\mathrm{Size}\Big(\dot{\bigcup_{i\leq k}} X_i, \zeta\Big) \approx_\zeta \dot{\bigcup_{i\leq k}} X_i \preceq_{\varepsilon/m+\zeta+\delta} \sum_{i\leq k} s_i$$

by Lemma 3.10 (iv), thus

$$\mathrm{Size}\Big(\dot{\bigcup_{i\leq k}} X_i, \zeta\Big) \leq \sum_{i\leq k} s_i + (\varepsilon/m + 3\zeta)(k+1)$$

by Lemma 3.11 (ii).

For $k = m$, we get

$$\dot{\bigcup_{i<m}} X_i \approx_\zeta \mathrm{Size}\Big(\dot{\bigcup_{i<m}} X_i, \zeta\Big) \preceq_{\varepsilon+3m\zeta} \sum_{i<m} s_i. \qquad \square$$

We can apply Proposition 3.15 only to sequences of sets encoded by a number, in particular, the length of the sequence is in Log. We present a variant which applies to larger families of sets, whose sizes are uniformly bounded. We can also read it contrapositively as an averaging argument: if we have a family of at most t sets, such that the size of their union is more than st, then one of the sets must be larger than $st/t = s$.

Proposition 3.16 (Averaging) *(in $PV_1 + sWPHP(PV)$) Let $X \subseteq 2^n \times 2^m$ and $Y \subseteq 2^m$ be definable by circuits, $Y \preceq_\delta t$, and $X_y \preceq_\varepsilon s$ for every $y \in Y$, where $X_y := \{x \mid \langle x, y\rangle \in X\}$. Then*

$$X \cap (2^n \times Y) \preceq_{\varepsilon+\delta+\varepsilon\delta+\xi} st$$

for any $\xi^{-1} \in \mathrm{Log}$.

Proof: By Lemma 3.14, there are $PV(\alpha)$-functions f, v such that

$$f(y,-)\colon v(y) \times (\mathrm{Size}(X_y, \xi) + \xi 2^n) \twoheadrightarrow v(y) \times X_y.$$

We may easily arrange $v(y) = v$ to be independent on y, while increasing the error slightly. Also, if $y \in Y$, we have $\mathrm{Size}(X_y, \xi) \le s + (\varepsilon + \xi)2^n$, thus we obtain a function f' such that

$$f'(y,-)\colon v \times (s + (\varepsilon + 3\xi)2^n) \twoheadrightarrow v \times X_y$$

for every $y \in Y$. There is a function g and number w such that

$$g\colon w \times (t + \delta 2^m) \twoheadrightarrow w \times Y,$$

and suitable composition of g with f' gives a function

$$vw(t + \delta 2^m)(s + 3\xi 2^n) \twoheadrightarrow vw \times (X \cap (2^n \times Y)).$$

We have

$$(t + \delta 2^m)(s + 3\xi 2^n) \le st + (\varepsilon + \delta + \varepsilon\delta + 6\xi)2^{n+m},$$

thus $X \cap (2^n \times Y) \preceq_{\varepsilon+\delta+\varepsilon\delta+6\xi} st$. □

The next task is to formalize a suitable version of Chernoff's bound, which is *sine qua non* for development of randomized algorithms. The proof consists of two parts. The number-theoretic part is a bound on certain sums of binomial coefficients; we reduce it to a special case which was formalized in [93]. The combinatorial part of Chernoff's bound relies on the fact that we can construct counting circuits for a set X and its complement $2^n \smallsetminus X$ so that the sizes approximately add up to 2^n.

Lemma 3.17 *There is a constant c such that PV_1 proves: for any $n > 0$, $x > 0$, $y \le x$, and $\delta, \varepsilon \in [0,1]$, such that $n \in \mathrm{Log}$,*

$$\sum_{j \le n\left(\frac{y}{x} - \delta\right)} \binom{n}{j} (y + \varepsilon x)^j (x - y + \varepsilon x)^{n-j} \le c\, x^n 4^{n(c\varepsilon - \delta^2)}.$$

Proof: Put

$$k := \left\lfloor n\frac{y+\varepsilon x}{(1+2\varepsilon)x}\right\rfloor, \qquad i := k - \left\lfloor n\left(\frac{y}{x}-\delta\right)\right\rfloor.$$

We assume $k > i \geq 0$, the remaining borderline cases are left as an exercise. The left-hand side is at most

$$S := \sum_{j\leq k-i}\binom{n}{j}(k+1)^j(n-k)^{n-j}\left(\frac{x(1+2\varepsilon)}{n}\right)^n \leq c\,x^n\left(1+\frac{1}{k}\right)^k(1+2\varepsilon)^n 4^{-i^2/n}$$

by [93, Prop. A.5]. We also have

$$\left(1+\frac{1}{k}\right)^k \leq 4.$$

Assume for simplicity $\varepsilon \leq 1/4$, and put $\ell := \lfloor 1/(2\varepsilon)\rfloor$. Then

$$(1+2\varepsilon)^n \leq \left(1+\frac{1}{\ell}\right)^n \leq \left(1+\frac{1}{\ell}\right)^{\ell\lceil n/\ell\rceil} \leq 4^{\lceil n/\ell\rceil},$$

and

$$\frac{n}{\ell} \leq \frac{n}{1/(2\varepsilon)-1} = \frac{2\varepsilon n}{1-2\varepsilon} \leq 4\varepsilon n,$$

thus $(1+2\varepsilon)^n \leq 4\cdot 4^{4\varepsilon n}$.

We have

$$n\frac{y}{x} - k \leq 1 + n\left(\frac{y}{x} - \frac{y+\varepsilon x}{(1+2\varepsilon)x}\right) = 1 + n\frac{\varepsilon(2y-x)}{(1+2\varepsilon)x} \leq 1 + n\varepsilon,$$

thus

$$i \geq k - n\left(\frac{y}{x}-\delta\right) = \delta n - \left(n\frac{y}{x} - k\right) \geq \delta n - (1+\varepsilon n),$$

and

$$-\frac{i^2}{n} \leq -\frac{(\delta n - (1+\varepsilon n))^2}{n} \leq -\delta^2 n + 2\delta(1+\varepsilon n) \leq 2 - \delta^2 n + 2\varepsilon n.$$

Putting everything together, we have

$$S \leq 4^4 c\,x^n 4^{6\varepsilon n - \delta^2 n}. \qquad \square$$

Proposition 3.18 (Chernoff's bound) (*in* $PV_1 + sWPHP(PV)$) *Let* $X \subseteq 2^n$ *be defined by a circuit,* $m \in \mathrm{Log}$, $0 \leq \varepsilon, \delta, p \leq 1$, *and* $X \succeq_\delta p2^n$. *Then*

$$\left\{w \in (2^n)^m \;\middle|\; |\{i < m \mid w_i \in X\}| \leq m(p-\varepsilon)\right\} \preceq_0 c4^{m(c\delta-\varepsilon^2)}2^{nm}$$

for some constant c, *where* w *is treated as a sequence of* m *numbers less that* 2^n, *and* w_i *is its* i*th member.*

Proof: Let $\xi = 1/m$, and $s = \mathrm{Size}(X, \xi)$. There is a $v > 0$ and functions f, g such that

$$f\colon v(s + \xi 2^n) \twoheadrightarrow v \times X,$$
$$g\colon v(2^n - s + \xi 2^n) \twoheadrightarrow v \times (2^n \smallsetminus X).$$

We can construct a function h by taking f and g coordinatewise so that

$$h(I, -)\colon v^m (s + \xi 2^n)^j (2^n - s + \xi 2^n)^{m-j} \twoheadrightarrow v^m \times \{w \mid I = \{i < m \mid w_i \in X\}\}$$

for every $I \subseteq m$ of size j. (The straightforward way of showing the surjectivity of h uses $BB\Sigma^b_1$ to collect preimages under f or g into a sequence. The choice schema $BB\Sigma^b_1$ is not available in PV_1, but we can avoid it as f and g have coretractions computable by poly-size circuits by Theorem 3.7.) We combine h with enumeration of small subsets of m, and obtain a function

$$v^m \sum_{j \le m(p-\varepsilon)} \binom{m}{j} (s + \xi 2^n)^j (2^n - s + \xi 2^n)^{m-j}$$
$$\twoheadrightarrow v^m \times \bigl\{w \in (2^n)^m \bigm| |\{i < m \mid w_i \in X\}| \le m(p - \varepsilon)\bigr\}.$$

Notice that $p2^n \le s + (\delta + 2\xi)2^n$ by Lemma 3.11 (ii). We invoke Lemma 3.17 with "x" $= 2^n$, "y" $= s + (\delta + 2\xi)2^n$, and "δ" $= 3\xi + \delta$, which gives

$$\sum_{j \le m(p-\varepsilon)} \binom{m}{j} (s + \xi 2^n)^j (2^n - s + \xi 2^n)^{m-j} \le c2^{nm} 4^{m(3c/m + c\delta - \varepsilon^2)} = 64^c c 2^{nm} 4^{m(c\delta - \varepsilon^2)}.$$

□

Another widely used property of counting is the inclusion-exclusion principle, which we formalize below. Notice that the assumptions on k and m are necessary so that the bounded sum in the statement of the principle is well-defined; thus it is not an additional restriction on applicability of the principle.

Proposition 3.19 (Inclusion-exclusion principle) (*in* $PV_1 + sWPHP(PV)$) *Let* $\{X_i \mid i < m\}$ *be subsets of* 2^n*, defined by a sequence of circuits. Let* $k \le m$ *be such that* $k \in \mathrm{LogLog}$ *and* $(m/k)^k \in \mathrm{Log}$*. Assume*

$$\bigcap_{i \in I} X_i \approx_{\varepsilon_I} s_I$$

for every $I \subseteq m$ *of size at most* k*, and define*

$$s = \sum_{\substack{I \subseteq m \\ 0 < |I| \le k}} (-1)^{|I|+1} s_I, \qquad \varepsilon = \sum_{\substack{I \subseteq m \\ 0 < |I| \le k}} \varepsilon_I.$$

Then

$$\bigcup_{i<m} X_i \succeq_{\varepsilon + \xi} s$$

if k *is even, and*

$$\bigcup_{i<m} X_i \preceq_{\varepsilon + \xi} s$$

if k *is odd, for any* $\xi^{-1} \in \mathrm{Log}$.

Proof: The sums are well-defined, as

$$\binom{m}{\le k} := \sum_{i\le k}\binom{m}{i} \le (4m/k)^k \in \mathrm{Log}$$

can be shown by easy induction on k, using $(1+1/k)^k \le 4$. For any $i \le \ell < m$, we define

$$X_i^\ell := \begin{cases} X_i, & i<\ell, \\ \bigcup_{j=\ell}^{m-1} X_j, & i=\ell. \end{cases}$$

Assume $k>0$ is even, the case of odd k is similar. Let $\eta^{-1} \in \mathrm{Log}$. We will show

$$\mathrm{Size}\Big(\bigcup_{i<m} X_i, \eta\Big) + 5\eta\binom{\ell}{\le k}2^n \ge \sum_{\substack{I\subseteq \ell+1\\ 0<|I|\le k}} (-1)^{|I|+1}\,\mathrm{Size}\Big(\bigcap_{i\in I} X_i^\ell, \eta\Big)$$

by induction on $\ell < m$. The base case $\ell = 0$ is trivial. Assume that the statement holds for $\ell - 1$. We have

$$\begin{aligned}(*) &:= \sum_{\substack{I\subseteq \ell\\ 0<|I|\le k}} (-1)^{|I|+1}\,\mathrm{Size}\Big(\bigcap_{i\in I} X_i^{\ell-1}, \eta\Big)\\ &= \sum_{\substack{I\subseteq \ell-1\\ 0<|I|\le k}} (-1)^{|I|+1}\,\mathrm{Size}\Big(\bigcap_{i\in I} X_i, \eta\Big) - \sum_{\substack{I\subseteq \ell-1\\ 0\le|I|<k}} (-1)^{|I|+1}\,\mathrm{Size}\Big(\bigcap_{i\in I} X_i \cap X_{\ell-1}^{\ell-1}, \eta\Big).\end{aligned}$$

By Lemma 3.11 (v), we have

$$\begin{aligned}\mathrm{Size}\Big(\bigcap_{i\in I} X_i \cap X_{\ell-1}^{\ell-1}, \eta\Big) &= \mathrm{Size}\Big(\big(\bigcap_{i\in I\cup\{\ell-1\}} X_i^\ell\big) \cup \big(\bigcap_{i\in I\cup\{\ell\}} X_i^\ell\big), \eta\Big)\\ &= \mathrm{Size}\Big(\bigcap_{i\in I\cup\{\ell-1\}} X_i^\ell, \eta\Big) + \mathrm{Size}\Big(\bigcap_{i\in I\cup\{\ell\}} X_i^\ell, \eta\Big)\\ &\qquad - \mathrm{Size}\Big(\bigcap_{i\in I\cup\{\ell-1,\ell\}} X_i^\ell, \eta\Big) \pm 5\eta 2^n,\end{aligned}$$

thus

$$\begin{aligned}(*) + 5\eta\binom{\ell-1}{\le k-1}2^n &\ge \sum_{\substack{I\subseteq \ell+1\\ 0<|I|\le k}} (-1)^{|I|+1}\,\mathrm{Size}\Big(\bigcap_{i\in I} X_i^\ell, \eta\Big) + (-1)^k \sum_{\substack{I\subseteq \ell-1\\ |I|=k-1}} \mathrm{Size}\Big(\bigcap_{i\in I\cup\{\ell-1,\ell\}} X_i^\ell, \eta\Big)\\ &\ge \sum_{\substack{I\subseteq \ell+1\\ 0<|I|\le k}} (-1)^{|I|+1}\,\mathrm{Size}\Big(\bigcap_{i\in I} X_i^\ell, \eta\Big).\end{aligned}$$

Using the induction hypothesis, we get

$$\operatorname{Size}\Big(\bigcup_{i<m} X_i, \eta\Big) + 5\eta\left(\binom{\ell-1}{\le k} + \binom{\ell-1}{\le k-1}\right)2^n \ge \sum_{\substack{I\subseteq \ell+1\\ 0<|I|\le k}} (-1)^{|I|+1}\operatorname{Size}\Big(\bigcap_{i\in I} X_i^\ell, \eta\Big),$$

and we can easily derive

$$\binom{\ell-1}{\le k} + \binom{\ell-1}{\le k-1} = \binom{\ell}{\le k}$$

from $\binom{\ell}{i+1} = \binom{\ell-1}{i+1} + \binom{\ell-1}{i}$.

We take $\ell = m-1$. We have

$$\sum_{\substack{I\subseteq m\\ 0<|I|\le k}} (-1)^{|I|+1}\operatorname{Size}\Big(\bigcap_{i\in I} X_i, \eta\Big) + \left(\varepsilon + 2\eta\binom{m}{\le k}\right)2^n \ge s$$

by Lemma 3.11 (ii), thus

$$\bigcup_{i<m} X_i \succeq_{\varepsilon+\xi} s,$$

where $\xi \le 7\eta\binom{m}{\le k}$. As $\binom{m}{\le k} \in \mathrm{Log}$, we can make ξ arbitrarily small by choosing a suitable $\eta^{-1} \in \mathrm{Log}$. □

Approximate counting, and estimation of probability with respect to the uniform distribution are two sides of the same coin, thus we can introduce probabilities in $PV_1 + sWPHP(PV)$ as in the following definition. All the results of Section 3 can be naturally restated in probabilistic terms, which we leave to the reader's imagination.

Definition 3.20 (in $PV_1 + sWPHP(PV)$) Let X be a definable subset of $2^{|t|}$, and $0 \le \varepsilon, p \le 1$. We define

$$\Pr_{x<t}(x \in X) \preceq_\varepsilon p \iff X \cap t \preceq_\varepsilon pt,$$

and similarly for $\succeq$, $\approx$. If X is defined by a circuit and $\varepsilon^{-1} \in \mathrm{Log}$, we put

$$\Pr_{x<t}(x \in X)_\varepsilon := \frac{1}{t}\operatorname{Size}(X \cap t, \varepsilon).$$

4 Randomized algorithms

Our main application of approximate counting is in the formalization of probabilistic algorithms in $PV_1 + sWPHP(PV)$. We will consider in turn the classes FRP, BPP, APP, MA, including their promise versions (prBPP, prMA). For each class we present a natural way to define algorithms from the class in $PV_1 + sWPHP(PV)$ (and its extensions), and we prove in $PV_1 + sWPHP(PV)$ basic properties of the class (such as success amplification, or simulation by circuits). We also discuss the problem whether all algorithms from the class can be defined in $PV_1 + sWPHP(PV)$: in general, algorithms from "syntactic classes" (like prBPP or APP) are always definable, whereas "semantic classes" (like BPP) cannot be shown to be captured by $PV_1 + sWPHP(PV)$ (or in fact, any recursively axiomatizable theory), without nontrivial progress in their derandomization. In the case of semantic classes we pinpoint the problem by showing that definability of any particular algorithm is equivalent to provability of a $\forall\Sigma^b_1$-sentence. (We show that the class APP is recursively enumerable, thus it can be considered a syntactic class even if that is not apparent from its definition.)

4.1 NP search problems

The first class of algorithms we mention are probabilistic solvers to NP search problems.

Definition 4.1 An NP *search problem* S is given by a poly-time computable relation $R(x, y)$ such that

$$R(x, y) \Rightarrow |y| \leq p(|x|)$$

for some polynomial p. Any y such that $R(x, y)$ is a *solution* of S for x, and x is called an *instance* of S in this context. The search problem S is *total* if every instance of S has a solution.

A deterministic algorithm *solves* S if it computes a solution for any given solvable instance of S. A probabilistic algorithm A *solves* S if

$$\Pr(A(x) \text{ is a solution for } x) \geq 1/2$$

for every solvable instance x. (The constant $1/2$ is rather arbitrary.)

The class of NP search problems solvable in probabilistic polynomial time is called FRP. The class of total search problems from FRP is denoted TFRP.

Notice that we may require without loss of generality that an algorithm solving an NP search problem rejects all unsolvable instances. The class of randomized poly-time algorithms which solve NP search problems under this requirement can be defined directly, without any reference to search problems: a probabilistic algorithm A computes an FRP-function, if for every input x, either $A(x)$ rejects with probability 1, or accepts and outputs a value with probability at least $1/2$. FRP can thus be thought of as a class of partial multifunctions. Notice that a language L is in ZPP iff its characteristic function is in FRP, and $L \in$ RP iff it is the domain of an FRP-function, thus FRP generalizes the classes ZPP and RP.

Formalization of FRP in $PV_1 + sWPHP(PV)$ was studied in [93]. We can restate the main definition of [93] in the present notation as follows.

Definition 4.2 (in $PV_1 + sWPHP(PV)$) A *β-definable randomized algorithm* is given by a pair of PV-functions $\langle A, r\rangle$ such that

$$\exists w < r(\vec{x})\, A(\vec{x}, w) \neq * \to \Pr_{w<r(\vec{x})}(A(\vec{x}, w) = *) \preceq_0 \beta,$$

where $*$ is a special symbol signalling a rejecting computation, and $0 < \beta < 1$. If unspecified, we take $\beta = 1/2$.

Various properties of FRP were proved in $PV_1 + sWPHP(PV)$ in [93]. We will not repeat these here, but instead we will concentrate on the question of which FRP-algorithms are definable in $PV_1 + sWPHP(PV)$. This is actually two questions: Which FRP-functions are *provably* $1/2$-*definable* in $PV_1 + sWPHP(PV)$, and which TFRP-functions are *provably total* in $PV_1 + sWPHP(PV)$. We begin with the latter.

For any NP search problem S, the statement "S is total" is a $\forall\Sigma^b_1$-sentence. Conversely, for any $\forall\Sigma^b_1$-sentence φ, we can construct an NP search problem S_φ such that φ holds iff S is total, thus description of provably total NP search problems of a theory is equivalent to characterization of its Σ^b_1-consequences. Wilkie's witnessing theorem (see [116]) states that provably total NP search problems of $PV_1 + sWPHP(PV)$ (or $S^1_2 + sWPHP(PV)$) are in TFRP, and it was shown in [93] that these witnessing TFRP-functions are definable and provably total in $PV_1 + sWPHP(PV)$:

Theorem 4.3 ([93]) *Assume $S^1_2 + sWPHP(PV) \vdash \forall x\,\exists y\,\varphi(x, y)$ with $\varphi \in \Sigma^b_1$, and let S be the corresponding search problem. There exists a probabilistic algorithm A such that PV_1 proves*

(i) *A is $1/2$-definable,*

(ii) *A solves S,*

and $PV_1 + sWPHP(PV)$ proves that A is total.

It is not clear whether all TFRP-functions are provably total in $PV_1 + sWPHP(PV)$, or in any its r.e. extension for that matter, even if we restrict ourselves to univalued functions with values in $\{0, 1\}$, i.e., ZPP-predicates. On one hand, such a result cannot be shown by a relativizing technique: it would imply that ZPP has a complete language due to Thapen [172], and there exist oracles A such that ZPP^A has no complete language [34]. On the other hand, TFRP is widely believed to coincide with FP, in which case all TFRP-functions (but not necessarily all TFRP-algorithms) are trivially definable in PV_1.

We can obtain a more precise characterization of provably total search problems of $PV_1 + sWPHP(PV)$, if we consider "nonintensional" representations instead of particular TFRP-algorithms.

Definition 4.4 A PV-formula φ *represents* a search problem S, if the following hold (in the standard model):

(i) if $\varphi(x, y)$, then y is a solution of S for x,

(ii) if x is a solvable instance of S, then $\exists y\,\varphi(x, y)$.

WPHPWIT is the following NP search problem: given a pair of circuits $G\colon 2^n \to 2^{2n}$ and $H\colon 2^{2n} \to 2^n$, find an $x < 2^{2n}$ such that $G(H(x)) \neq x$.

Let S and S' be NP search problems. S is *reducible* to S', if there are poly-time functions f and g such that:

(i) if x is a solvable instance of S, then $f(x)$ is a solvable instance of S',

(ii) if y is a solution of S' for $f(x)$, then $g(x, y)$ is a solution of S for x.

Theorem 4.5 *Let S be an* NP *search problem. The following are equivalent:*

(i) *S has a provably total representation in $PV_1 + sWPHP(PV)$.*

(ii) *S is reducible to $WPHPWIT$.*

Proof:
(i) $\to$ (ii) follows from Thapen's proof of Wilkie's witnessing theorem [173].

(ii) $\to$ (i): assume that S is given by a poly-time relation $R(x,y)$, and f and g form a reduction of S to $WPHPWIT$. We may easily modify f so that its output $f(x) = \langle G_x, H_x\rangle$ consists of a pair of circuits as in Definition 4.4, provably in $PV_1 + sWPHP(PV)$. Put

$$\varphi(x,y) = R(x,y) \vee \big(G_x(H_x(y)) \neq y \wedge \neg R(x, g(x,y))\big).$$

The second disjunct never holds in the standard model by the definition of reduction, thus φ represents S. $PV_1 + sWPHP(PV)$ proves $\forall x\, \exists y\, \varphi(x,y)$, as $G_x(H_x(y)) \neq y$ implies $\varphi(x, g(x,y))$ or $\varphi(x,y)$. □

As noticed in [93], $WPHPWIT$ can also be used as an axiomatic description of Σ^b_1-theorems of $PV_1 + sWPHP(PV)$, which is again implicit in Thapen's proof of Wilkie's witnessing theorem.

Proposition 4.6 *The statement "$WPHPWIT$ is total" axiomatizes $\forall\Sigma^b_1$-consequences of PV_1+ $sWPHP(PV)$ over PV_1.*

We return to the question which FRP-algorithms (not necessarily total) are definable in a given theory T. Perhaps surprisingly, this question is essentially equivalent to a $\forall\Sigma^b_1$-sentence, it thus reduces to the problem of the provably total TFRP-functions discussed above. (The constants 1/2 and 2/3 below are arbitrary.)

Theorem 4.7 *Let A be a* FRP*-algorithm with error* $1/2$*. There exists a true $\forall\Sigma^b_1$-sentence φ such that $PV_1 + sWPHP(PV)$ proves*

(i) *if φ, then A is* $2/3$*-defined,*

(ii) *if A is* $1/2$*-defined, then φ.*

Moreover, the (*total*) NP *search problem S_φ associated with φ is in* TFRP*: there exists a randomized algorithm B such that $PV_1 + sWPHP(PV)$ proves*

(iii) *if A is* $1/2$*-defined, then B is* $1/2$*-defined, total, and solves S_φ.*

Proof: The idea is to consider the $PV(\alpha)$-formula

$$\psi^\alpha(x,y) = \big(y < r(x) \wedge A(x,y) \neq * \to \Pr_{w<r(x)}(A(x,w) = *)_{1/50} \leq 5/8\big),$$

where $*$ is as in Definition 4.2. Clearly, $HARD^A$ proves

$$\begin{aligned} \forall x, y\, \psi^\alpha(x,y) &\to\ A \text{ is } 2/3\text{-defined},\\ A \text{ is } 1/2\text{-defined}\ &\to \forall x, y\, \psi^\alpha(x,y). \end{aligned}$$

We need to eliminate α from the formula. In the proof of Theorem 3.7 (resp. Lemma 3.14), the exact choice of the function f is not relevant: the behaviour of $\Pr(\dots)_{1/50}$ is preserved if we replace α by any average-case 1/4-hard Boolean function f in the right number of variables. We thus define

$$\varphi'(x,y,f) = \big((f\colon 2^{\ell(x)} \to 2) \wedge \mathrm{Hard}^A_{1/4}(f) \to \psi^f(x,y)\big),$$

where $\ell(x) \in \mathrm{LogLog}$ is chosen as in Theorem 3.7. Then φ' is a Σ^b_1-formula, and $PV_1 + sWPHP(PV)$ proves

$$\forall x,y,f\, \varphi'(x,y,f) \to \ A \text{ is 2/3-defined},$$
$$A \text{ is 1/2-defined } \to \forall x,y,f\, \varphi'(x,y,f).$$

We use a witnessing argument to show that S_φ is solvable in randomized polynomial time. Notice that the only non-sharply bounded existential quantifier in φ' is the one from $\neg\,\mathrm{Hard}^A_{1/4}(f)$. $PV_1 + sWPHP(PV)$ proves the Σ^b_1-formula

$$(f,g\colon 2^{\ell(x)} \to 2) \wedge \Pr\nolimits^f_{w<r(x)}(A(x,w)=*)_{1/50} > 5/8 \wedge \Pr\nolimits^g_{w<r(x)}(A(x,w)=*)_{1/50} \le 9/16$$
$$\to \neg\,\mathrm{Hard}^A_{1/4}(f) \vee \neg\,\mathrm{Hard}^A_{1/4}(g).$$

By Wilkie's witnessing theorem there exists a probabilistic algorithm $h(x,y,f,g) \in \mathrm{TFRP}$ such that

$$(f,g\colon 2^{\ell(x)} \to 2) \wedge \Pr\nolimits^f_{w<r(x)}(A(x,w)=*)_{1/50} > 5/8 \wedge \Pr\nolimits^g_{w<r(x)}(A(x,w)=*)_{1/50} \le 9/16$$
$$\to \mathrm{Wit}_{\neg\,\mathrm{Hard}^A_{1/4}(f)}(h(x,y,f,g)) \vee \mathrm{Wit}_{\neg\,\mathrm{Hard}^A_{1/4}(g)}(h(x,y,f,g))$$

holds with high probability. As A has error at most 1/2, the implication

$$y < r(x) \wedge A(x,y) \ne * \wedge \mathrm{Hard}^A_{1/4}(g) \to \Pr\nolimits^g_{w<r(x)}(A(x,w)=*)_{1/50} \le 9/16$$

is true. Let $B(x,y,f)$ be the probabilistic algorithm which generates a random function g, and applies $h(x,y,f,g)$. As most Boolean functions are average-case 1/4-hard, we have

$$y < r(x) \wedge A(x,y) \ne * \wedge (f\colon 2^{\ell(x)} \to 2) \wedge \Pr\nolimits^f_{w<r(x)}(A(x,w)=*)_{1/50} > 5/8$$
$$\to \mathrm{Wit}_{\neg\,\mathrm{Hard}^A_{1/4}(f)}(B(x,y,f))$$

with high probability. This construction can be easily formalized in $PV_1 + sWPHP(PV)$, using Theorem 4.3 and Lemma 3.2. □

4.2 The classes BPP and promise BPP

BPP, introduced by Gill [80], is arguably the most popular randomized complexity class. It is generally considered a good approximation to the class of problems which are efficiently solvable in practice.

Definition 4.8 A language L is in BPP, if there exists a probabilistic poly-time decision algorithm A such that for every x,

$$x \in L \Rightarrow \Pr(A(x)) \geq 3/4,$$
$$x \notin L \Rightarrow \Pr(A(x)) \leq 1/4.$$

A *promise problem* is a pair $L = \langle L^+, L^- \rangle$ of disjoint languages. An ordinary language L is identified with the promise problem $\langle L, \{0,1\}^{<\omega} \smallsetminus L \rangle$. A promise problem L is in *promise* BPP ($L \in$ prBPP for short), if there exists a probabilistic poly-time algorithm A such that for every x,

$$x \in L^+ \Rightarrow \Pr(A(x)) \geq 3/4,$$
$$x \in L^- \Rightarrow \Pr(A(x)) \leq 1/4.$$

Formalizing the definition of prBPP in $PV_1 + sWPHP(PV)$ is a straightforward application of the approximate counting machinery.

Definition 4.9 (in $PV_1 + sWPHP(PV)$) Let β be a PV-function with values in $(0, 1/2)$, A a PV-predicate, and r a PV-function. The pair $\langle A, r \rangle$ *β-defines the* prBPP *problem* $L_{A,r,\beta} = \langle L^+_{A,r,\beta}, L^-_{A,r,\beta} \rangle$, where

$$x \in L^+_{A,r,\beta} \iff \Pr_{w<r(x)}(\neg A(x,w)) \preceq_0 \beta(x),$$
$$x \in L^-_{A,r,\beta} \iff \Pr_{w<r(x)}(A(x,w)) \preceq_0 \beta(x).$$

More generally, if L^+, L^- are disjoint definable sets, the promise problem $L = \langle L^+, L^- \rangle$ is *β-defined* by $\langle A, r \rangle$ if $L^+ \subseteq L^+_{A,r,\beta}$ and $L^- \subseteq L^-_{A,r,\beta}$.

The pair $\langle A, r \rangle$ *β-defines a* BPP *language* if $\forall x\,(x \in L^+_{A,r,\beta} \vee x \in L^-_{A,r,\beta})$.

If unspecified, we take $\beta = 1/4$.

Lemma 4.10 (*in $PV_1 + sWPHP(PV)$*) *Let L be a definable* prBPP*-problem, and $n \in$* Log*. There exists a Boolean circuit $C\colon 2^n \to 2$ such that*

$$x \in L^+ \Rightarrow C(x) = 1,$$
$$x \in L^- \Rightarrow C(x) = 0,$$

for every $x < 2^n$.

Proof: Work in $HARD^A$. By Lemma 3.14, there is a $PV(\alpha)$-predicate $P(x)$ such that

$$x \in L^+ \Rightarrow P(x),$$
$$x \in L^- \Rightarrow \neg P(x).$$

We may compute P on a bounded interval by an oracle-free circuit, as $\alpha(x)$ only depends on the length of x. $\square$

Proposition 4.11 (*in $PV_1 + sWPHP(PV)$*) *Let t, s be PV-functions such that $t(x), s(x) > 0$, and $1/s(x) + 1/|t(x)| \le 1/2$. Let $L = \langle L^+, L^- \rangle$ be a promise problem. The following are equivalent.*

(i) *L is a $(1/2 - 1/|t|)$-definable* prBPP*-problem,*

(ii) *L is a $1/4$-definable* prBPP*-problem,*

(iii) *L is a $1/s$-definable* prBPP*-problem.*

Proof: The only interesting implication is (i) $\to$ (iii). Assume that L is $(1/2 - 1/|t|)$-defined by $\langle A, r \rangle$. Let c be the constant from Proposition 3.18, put $m(x) = |t(x)|^2|cs(x)|$, $r'(x) = r(x)^{m(x)}$, and

$$A'(x, w') \leftrightarrow \big(|\{i < m(x) \mid A(x, w_i)\}| \ge m(x)/2\big),$$

where $w' < r'(x)$ is viewed as a sequence $\langle w_i \mid i < m(x) \rangle$ of numbers less than $r(x)$. Then L is $1/s$-defined by $\langle A', r' \rangle$ due to Chernoff's bound (Proposition 3.18). $\square$

Notice that prBPP is defined by a purely syntactic condition: in other words, every pair $\langle A, r \rangle$ of PV-functions (provably) defines a prBPP-problem.

Corollary 4.12 *Every* prBPP*-algorithm is definable in $PV_1 + sWPHP(PV)$.*

Definable BPP-languages are essentially "provably total" prBPP-problems. As in the case of TFRP, we do not know whether all BPP-languages are definable in $PV_1 + sWPHP(PV)$ or its r.e. extension; again, relativizing techniques cannot work, as Thapen's result is applicable to BPP, and an oracle with respect to which BPP does not have a complete language was constructed in [88]. We show that the totality of a BPP-algorithm is essentially equivalent to a $\forall\Sigma^b_1$-sentence, thus the characterization of the BPP-languages definable in a particular theory can be reduced to the characterization of its provably total TFRP-functions.

Theorem 4.13 *Let A be a* BPP*-algorithm. There exists a true $\forall\Sigma^b_1$-sentence φ such that $PV_1 + sWPHP(PV)$ proves*

(i) *if φ, then A $1/3$-defines a* BPP*-language,*

(ii) *if A $1/4$-defines* BPP*-language, then φ.*

Moreover, the NP *search problem S_φ associated with φ is in* TFRP*. There is a randomized algorithm B such that $PV_1 + sWPHP(PV)$ proves*

(iii) *if A $1/4$-defines* BPP*-language, then B is $1/2$-defined, total, and solves S_φ.*

Proof: We define

$$\varphi = \forall x\, \forall f\, \big((f\colon 2^{\ell(x)} \to 2) \wedge \mathrm{Hard}^A_{1/4}(f) \\ \to \Pr^f_{w<r(x)}(A(x,w))_{1/50} < 7/24 \vee \Pr^f_{w<r(x)}(\neg A(x,w))_{1/50} < 7/24\big)$$

with suitably chosen $\ell(x) \in \mathrm{LogLog}$, and proceed as in the proof of 4.7.

There is a minor complication in the construction of the probabilistic solver to S_φ: the algorithm cannot directly decide which of the disjuncts in φ should hold, as we do not know whether BPP = ZPP. The solution is to try both possibilities, and check whether either of them leads to a correct witness for $\neg\,\mathrm{Hard}^A_{1/4}(f)$. □

A similar argument can be used to prove that prBPP lies on the second level of the polynomial hierarchy. The original result (formulated for BPP only) is due to Sipser and Gács [166], and it was simplified by Lautemann [122]. We follow an alternative proof due to Nisan and Wigderson [135].

Proposition 4.14 *Let A be a PV-predicate, and r a PV-functions. There are Σ^b_2-formulas $\sigma^+(x), \sigma^-(x)$ and Π^b_2-formulas $\pi^+(x), \pi^-(x)$ such that $PV_1 + sWPHP(PV)$ proves*

$$x \in L^+_{A,r,1/4} \to \pi^+(x) \to \sigma^+(x) \to x \in L^+_{A,r,1/3},$$
$$x \in L^-_{A,r,1/4} \to \pi^-(x) \to \sigma^-(x) \to x \in L^-_{A,r,1/3}.$$

In particular, any definable BPP*-language is in $\Sigma^b_2 \cap \Pi^b_2$.*

Proof: It suffices to define

$$\pi^+(x) = \forall f\,\big(f\colon 2^{\ell(x)} \to 2 \wedge \mathrm{Hard}^A_{1/4}(f) \to \mathrm{Pr}^f_{w<r(x)}(\neg A(x,w))_{1/50} \le 7/24\big),$$
$$\sigma^+(x) = \exists f\,\big(f\colon 2^{\ell(x)} \to 2 \wedge \mathrm{Hard}^A_{1/4}(f) \wedge \mathrm{Pr}^f_{w<r(x)}(\neg A(x,w))_{1/50} \le 7/24\big),$$
$$\pi^-(x) = \forall f\,\big(f\colon 2^{\ell(x)} \to 2 \wedge \mathrm{Hard}^A_{1/4}(f) \to \mathrm{Pr}^f_{w<r(x)}(A(x,w))_{1/50} \le 7/24\big),$$
$$\sigma^-(x) = \exists f\,\big(f\colon 2^{\ell(x)} \to 2 \wedge \mathrm{Hard}^A_{1/4}(f) \wedge \mathrm{Pr}^f_{w<r(x)}(A(x,w))_{1/50} \le 7/24\big).$$

The quantifiers over f are bounded as $f \le 2^{2^{\ell(x)}}$ and $\ell(x) = O(||x||)$. □

To complete the picture we mention an elegant alternative description of definable BPP-languages, based on implicit definability in (extensions of) $HARD^A$. The intuition behind this characterization stems from the well-known result BPP = almost-P (cf. [27, 135]).

Definition 4.15 Let T be a simple extension of $PV_1 + sWPHP(PV)$, and $T^+(\alpha) := T + HARD^A$. A $PV(\alpha)$-predicate $P^\alpha(x)$ is a *T^+-definable implicitly poly-time predicate*, if

$$T^+(\alpha) + T^+(\beta) \vdash P^\alpha(x) \leftrightarrow P^\beta(x).$$

Theorem 4.16 *Let T be a simple extension of $PV_1 + sWPHP(PV)$.*

(i) *Every T-provably total* BPP*-language is in T^+ equivalent to a T^+-definable implicitly poly-time predicate.*

(ii) *Every T^+-definable implicitly poly-time predicate is in T^+ equivalent to a T-provably total* BPP*-language.*

Proof:

(i): let L be a definable BPP-language. By Lemma 3.14, there exists a $PV(\alpha)$-predicate P^α such that

$$T^+(\alpha) \vdash P^\alpha(x) \leftrightarrow x \in L.$$

Then clearly

$$T^+(\alpha), T^+(\beta) \vdash P^\alpha(x) \leftrightarrow P^\beta(x).$$

(ii): assume that

$$T^+(\alpha), T^+(\beta) \vdash P^\alpha(x) \leftrightarrow P^\beta(x).$$

Let c be a constant such that $P^\alpha(x)$ only accesses the value of $\alpha(y)$ for $||y|| \le c||x||$. Work in $T^+(\alpha)$. Fix x, let $f = \langle f_i \mid i \le c||x|| \rangle$ be a sequence of average-case 1/4-hard functions $f_i \colon 2^i \to 2$, and define

$$\beta(y) = \begin{cases} f_{||y||}, & ||y|| \le c||x||, \\ \alpha(y), & \text{otherwise.} \end{cases}$$

Then β defines a (parametric) interpretation of $T^+(\beta)$ in $T^+(\alpha)$, and consequently $P^\alpha(x) \leftrightarrow P^f(x)$.

We thus have

$$T^+(\alpha) \vdash \forall i \le c||x|| \bigl(f_i \colon 2^i \to 2 \wedge \mathrm{Hard}^A_{1/4}(f_i)\bigr) \to (P^\alpha(x) \leftrightarrow P^f(x)).$$

Let A be the formalization of the following randomized algorithm: on input x, generate a random sequence $f = \langle f_i \mid i \le c||x|| \rangle$ of functions $f_i \colon 2^i \to 2$, and output $P^f(x)$. By [93, L. 4.10], $PV_1 + sWPHP(PV)$ proves

$$Pr_f\bigl(\neg\forall i \le c||x|| \ \mathrm{Hard}^A_{1/4}(f_i)\bigr) \preceq_0 1/4,$$

thus

$$T^+(\alpha) \vdash \Pr_f\bigl(A(x,f) \leftrightarrow \neg P^\alpha(x)\bigr) \preceq_0 1/4.$$

In particular,

$$T^+(\alpha) \vdash \Pr_f(A(x,f)) \preceq_0 1/4 \vee \Pr_f(\neg A(x,f)) \preceq_0 1/4,$$

i.e., A is a 1/4-defined BPP-algorithm in $T^+(\alpha)$, and by Theorem 3.13, also in T. If L denotes the BPP-language defined by A, clearly

$$T^+(\alpha) \vdash P^\alpha(x) \leftrightarrow x \in L$$

as required. □

4.3 The class APP

The class APP is a generalization of BPP introduced by Kabanets, Rackoff, and Cook [111]. It comprises a representative class of algorithms which can be derandomized using the current methods for proving $P = \mathrm{BPP}$ (viz. hardness-randomness tradeoffs), and unlike BPP, it is known to have a complete problem. A unique feature of APP is that it does not consist of languages (or promise problems), but functions with real values in the interval $[0,1]$.

Definition 4.17 A real-valued function $f\colon \omega \to [0,1]$ is in APP, if there exists a probabilistic poly-time function $g(x,y)$ with values in $[0,1]_{\mathbb{Q}}$ such that

$$\Pr\bigl(|f(x) - g(x, 2^k)| \le 1/k\bigr) \ge 3/4$$

for all x and k.

We cannot directly talk about real numbers in bounded arithmetic, we thus have to formalize APP-algorithms without an explicit reference to the functions which they compute. The idea is similar to methods used in constructive analysis (cf. [31]).

Definition 4.18 (in $PV_1 + sWPHP(PV)$) Let $\beta(x,y)$ be a PV-function with rational values in $(0,1/2)$. A *β-definable* APP-*algorithm* is given by a pair of PV-functions $g(x,y,w)$ and $r(x,y)$, where r has positive integer values, g has rational values in $[0,1]$, and

$$\forall x\,\forall k,\ell \in \mathrm{Log}\ \exists a \in [0,1]\ \bigl(\Pr_{w<r(x,2^k)}\bigl(|g(x,2^k,w) - a| > 1/k\bigr) \preceq_0 \beta(x,2^k) \\ \wedge \Pr_{w<r(x,2^\ell)}\bigl(|g(x,2^\ell,w) - a| > 1/\ell\bigr) \preceq_0 \beta(x,2^\ell)\bigr).$$

When unspecified, we take $\beta = 1/4$.

Let $\langle g', r'\rangle$ be a β'-definable APP-algorithm. We say that $\langle g, r\rangle$ and $\langle g', r'\rangle$ *compute the same function* if

$$\forall x\,\forall k \in \mathrm{Log}\ \exists a \in [0,1]\ \bigl(\Pr_{w<r(x,2^k)}\bigl(|g(x,2^k,w) - a| > 1/k\bigr) \preceq_0 \beta(x,2^k) \\ \wedge \Pr_{w<r'(x,2^k)}\bigl(|g'(x,2^k,w) - a| > 1/k\bigr) \preceq_0 \beta'(x,2^k)\bigr).$$

Proposition 4.19 *(in $PV_1 + sWPHP(PV)$) Let $t(x,y)$ and $s(x,y)$ be PV-functions with positive integer values. If $\langle g, r\rangle$ is a $(1/2 - 1/|t|)$-definable* APP-*algorithm, there exists a $1/s$-definable* APP-*algorithm $\langle g', r'\rangle$ which computes the same function as $\langle g, r\rangle$.*

Proof: Let c be the constant from Proposition 3.18, and let $m(x,y) := |cs(x,y)||t(x,y)|^2 \in \mathrm{Log}$. Put

$$r'(x,y) = r(x,y)^{m(x,y)}, \\ g'(x,y,w') = \mathrm{Median}(g(x,y,w_0), \dots, g(x,y,w_{m-1})),$$

where $w' < r'(x,y)$ is considered as a sequence of $m = m(x,y)$ numbers $w_i < r(x,y)$. Fix x, $k \in \mathrm{Log}$, and $a \in [0,1]$ such that

$$\Pr_{w<r}\bigl(|g(x,2^k,w) - a| > 1/k\bigr) \preceq_0 1/2 - 1/|t|.$$

By Proposition 3.18 (Chernoff's bound), we have

$$\Pr_{w'<r'}\Bigl(\bigl|\{i < m \mid |g(x,2^k,w_i) - a| > 1/k\}\bigr| \ge m/2\Bigr) \preceq_0 c4^{-m/|t|^2} \le 1/s.$$

The median of a set of numbers falls into the interval $I = [a - 1/k, a + 1/k]$ whenever more than half of the numbers are in I, thus

$$\Pr_{w'<r'}\bigl(|g'(x,2^k,w') - a| > 1/k\bigr) \preceq_0 1/s. \qquad \square$$

Definition 4.20 An APP-function $f\colon \omega \to [0,1]$ is *representable in a theory T*, if there exists a pair of PV-functions $\langle g, r\rangle$ which, provably in T, 1/4-defines an APP-algorithm, and for any x and k,

$$\Pr_{w<r(x,2^k)}\bigl(|g(x,2^k,w) - f(x)| > 1/k\bigr) \le 1/4$$

is true in $\mathbb{N}$.

We want to show that all APP-functions are representable in $PV_1 + sWPHP(PV)$. Notice that for any reasonable model of computation (such as APP), the class of algorithms representable in a given recursively axiomatizable theory is recursively enumerable. We thus need to establish recursive enumerability of APP as a necessary prerequisite (it was left as an open problem in [111]).

Definition 4.21 Let $f, g\colon \omega \to [0,1]$ be real-valued functions. We say that f is (*poly-time many-one approximately*) *reducible* to g, if there is a poly-time function r such that for every x and k,

$$|f(x) - g(r(x,2^k))| \le 1/k.$$

The *Circuit Acceptance Probability Problem* (*CAPP*) is the real-valued function f_{CAPP} such that for every Boolean circuit $C\colon 2^n \to 2$,

$$f_{CAPP}(C) = \Pr_{u<2^n}(C(u) = 1).$$

Theorem 4.22 ([111]) *A function f is in* APP *if and only if f is reducible to f_{CAPP}.*

Theorem 4.23 *The class* APP *is recursively enumerable. I.e., there exists a recursive sequence $\{A_e \mid e \in \omega\}$ such that*

- *each A_e is a description of an* APP*-algorithm approximating a function f_e,*
- *for every $f \in$* APP*, there is an e such that $f = f_e$.*

Proof: Let $\{g_e \mid e \in \omega\}$ be a recursive enumeration of all clocked poly-time algorithms $g(x,y)$, such that the output of $g(x,y)$ is a description of a Boolean circuit. Let Cut_p^q be the cut-off function

$$\mathrm{Cut}_p^q(x) := \max\{p, \min\{q, x\}\} = \begin{cases} q, & x \ge q, \\ x, & p \le x \le q, \\ p, & x \le p. \end{cases}$$

Let $A_e(x, 2^k)$ be the algorithm described in Figure 4.1. Clearly, A_e is a probabilistic poly-time algorithm. Fix e and x, and define

$$\begin{aligned} C_i &:= g_e(x, 2^i), \\ a_i &:= \Pr_u(C_i(u) = 1), \\ b_k &:= \mathrm{Cut}_0^1\Bigl(a_1 + \sum_{i=2}^{k} \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(a_i - a_{i-1})\Bigr). \end{aligned}$$

```
input: $x$, $2^k$
for $i = 1, \dots, k$ do:
    $C_i \leftarrow g_e(x, 2^i)$
    whp, compute $c_i$ such that $|c_i - \Pr_u(C_i(u) = 1)| \le 1/(4k^2)$ by random sampling
output $\mathrm{Cut}_0^1\Big(c_1 + \sum_{i=2}^k \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(c_i - c_{i-1})\Big)$
```

Figure 4.1: The APP-algorithm A_e.

For any $k < \ell$, we have

$$|b_\ell - b_k| \le \Big|\sum_{i=k+1}^{\ell} \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(a_i - a_{i-1})\Big| \le \sum_{i=k+1}^{\ell} \frac{1}{2i^2} \le \frac{1}{2}\sum_{i=k+1}^{\infty} \frac{1}{(i-1)i} = \frac{1}{2k},$$

thus the sequence $\{b_k \mid k \in \omega\}$ is Cauchy, and converges to a number $f_e(x) := b \in [0,1]$ such that

$$|b - b_k| = \lim_{\ell \to \infty} |b_\ell - b_k| \le \frac{1}{2k}.$$

Fix k, and consider a computation of A_e on input $\langle x, 2^k \rangle$. For all $i = 1, \dots, k$, let $c_i \in [0,1]$ be as in Figure 4.1. With high probability, we have

$$|c_i - a_i| \le \frac{1}{4k^2}$$

for every i. Let

$$d := \mathrm{Cut}_0^1\Big(c_1 + \sum_{i=2}^{k} \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(c_i - c_{i-1})\Big)$$

be the output of the algorithm. Addition, subtraction, and the cut-off function are 1-Lipschitz, thus

$$\begin{aligned} |d - b_k| &\le |c_1 - a_1| + \sum_{i=2}^{k} |(c_i - c_{i-1}) - (a_i - a_{i-1})| \\ &\le |c_1 - a_1| + \sum_{i=2}^{k} \big(|c_i - a_i| + |c_{i-1} - a_{i-1}|\big) \\ &\le (2k-1)\frac{1}{4k^2} \le \frac{1}{2k}, \end{aligned}$$

and

$$|d - b| \le |d - b_k| + |b_k - b| \le \frac{1}{k}.$$

This means that A_e is an APP-algorithm for f_e.

Let f be an arbitrary APP-function. By APP-completeness of CAPP, there is a poly-time function g such that for any x and k, $C := g(x, 2^k)$ is a Boolean circuit satisfying

$$|f(x) - \Pr_u(C(u) = 1)| \le \frac{1}{k}.$$

Choose e such that

$$g_e(x, 2^k) = g(x, 2^{4(k+1)^2}).$$

Fix x, and define the sequences C_i, a_i, and b_i as above. We have

$$|a_i - a_{i-1}| \le \frac{1}{4(i+1)^2} + \frac{1}{4i^2} \le \frac{1}{2i^2},$$

thus

$$b_k = \mathrm{Cut}_0^1\Big(a_1 + \sum_{i=2}^{k}(a_i - a_{i-1})\Big) = \mathrm{Cut}_0^1(a_k) = a_k,$$

which means

$$f_e(x) = \lim_{k\to\infty} b_k = \lim_{k\to\infty} a_k = f(x).$$

As x was arbitrary, $f_e = f$. □

Lemma 4.24 *CAPP is representable in $PV_1 + sWPHP(PV)$.*

Proof: Let $\langle g, r\rangle$ be the formalization of the following algorithm: given C and 2^k, choose a random Boolean function f in a suitable number of variables, and output $\Pr_u^f(C(u) = 1)_{1/(3k)}$.

Fix $C\colon 2^n \to 2$, $k < \ell \in \mathrm{Log}$, and let $\xi = 1/(3\ell)$, $\Pr_u(C(u) = 1) \approx_\xi a$. As $PV_1 + sWPHP(PV)$ proves

$$\Pr_f(\neg\,\mathrm{Hard}^A_{1/4}(f)) \preceq_0 1/4$$

(Lemma 3.2), we have

$$\Pr_f\big(|g(C, 2^k, f) - a| > 1/(3k) + \xi + \xi\big) \preceq_0 1/4$$

and

$$\Pr_f\big(|g(C, 2^\ell, f) - a| > 1/(3\ell) + \xi + \xi\big) \preceq_0 1/4$$

by Lemma 3.11 (ii). □

We remark that the combinatorial core of Theorem 4.22 can also be formalized in $PV_1 + sWPHP(PV)$ with no difficulty. However, we do not know how to sensibly *formulate* the statement of Theorem 4.22 in $PV_1 + sWPHP(PV)$, due to absence of real numbers in bounded arithmetic.

Theorem 4.25 *Every* APP*-function f is representable in $PV_1 + sWPHP(PV)$.*

Proof: The basic idea is to partially formalize Theorem 4.23 in $PV_1 + sWPHP(PV)$.

As in Theorem 4.23, choose a PV-function $h(x, y)$ such that for every x and k we have

$$|f(x) - \Pr_u(C(u) = 1)| \le 1/(8k^2),$$

where $C = h(x, 2^k)$. Let $\langle g_{CAPP}, r_{CAPP}\rangle$ be the representation of CAPP from Lemma 4.24, amplified by Proposition 4.19 so that the error on input $\langle C, 2^k\rangle$ is at most $1/k$. We may assume that $r_{CAPP}(x, 2^k)$ is always a power of 2. Define $g(x, 2^k, w)$ as in Figure 4.2, and let $r(x, 2^k)$ be a power of 2 large enough to accommodate all calls to g_{CAPP} inside g. The functions $\langle g, r\rangle$

```
input: $x$, $2^k$, $w$
for $i = 1, \dots, k$ do:
    $C_i \leftarrow h(x, 2^i)$
    $c_i \leftarrow g_{CAPP}\big(C_i, 2^{1/(8k^2)}, \big(w \bmod r_{CAPP}(C_i, 2^{1/(8k^2)})\big)\big)$
output $\mathrm{Cut}_0^1\Big(c_1 + \sum_{i=2}^{k} \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(c_i - c_{i-1})\Big)$
```

Figure 4.2: The function g, formalizing A_e.

represent f by the proof of Theorem 4.23, it remains to prove in $PV_1 + sWPHP(PV)$ that $\langle g, r\rangle$ is a 1/4-defined APP-algorithm.

Work in $HARD^A$. Fix x, and $k < \ell \in \mathrm{Log}$. Define

$$\begin{aligned} C_i &:= h(x, 2^i), \\ a_i &:= \Pr_u(C_i(u) = 1)_{1/(10\ell^2)}, \\ a &:= \mathrm{Cut}_0^1\Big(a_1 + \sum_{i=2}^{\ell} \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(a_i - a_{i-1})\Big), \\ a' &:= \mathrm{Cut}_0^1\Big(a_1 + \sum_{i=2}^{k} \mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(a_i - a_{i-1})\Big) \end{aligned}$$

for every $i \le \ell$. Consider first the computation of $d := g(x, 2^\ell, w)$ on a random input w, and let c_i be as in Figure 4.2. For every $i \le \ell$ and suitably chosen small ξ, we have

$$|c_i - a_i| \le \frac{1}{8\ell^2} + \frac{1}{10\ell^2} + \xi \le \frac{1}{4\ell^2}$$

with probability at least $1 - 1/(8\ell^2)$, thus

$$\forall i \le \ell\, |c_i - a_i| \le 1/(4\ell^2)$$

with probability $1 - \ell/(8\ell^2) - \xi \ge 3/4$ by Proposition 3.15. When this happens, we have

$$|d - a| \le |c_1 - a_1| + \sum_{i=2}^{\ell}\big(|c_i - a_i| + |c_{i-1} - a_{i-1}|\big) \le \frac{2\ell - 1}{4\ell^2} < \frac{1}{2\ell}$$

as in Theorem 4.23, thus

$$\Pr_w\big(|g(x, 2^\ell, w) - a| > 1/\ell\big) \preceq_0 1/4.$$

Now consider the computation of $d' := g(x, 2^k, w)$. We have

$$|d' - a'| < \frac{1}{2k}$$

with probability at least 3/4 by the same reasoning as above. Moreover,

$$|a' - a| \le \sum_{i=k+1}^{\ell} \big|\mathrm{Cut}_{-1/(2i^2)}^{1/(2i^2)}(a_i - a_{i-1})\big| \le \sum_{i=k+1}^{\ell} \frac{1}{2(i-1)i} = \frac{1}{2}\left(\frac{1}{k} - \frac{1}{\ell}\right),$$

where the last equality follows by induction on ℓ. Consequently

$$|g(x, 2^k, w) - a| \le \frac{1}{2k} + \frac{1}{2k} - \frac{1}{2\ell} < \frac{1}{k}$$

holds with probability at least 3/4. □

4.4 The classes MA and promise MA

Babai [16] (cf. [17]) introduced a hierarchy of complexity classes based on public-coin randomized interactive proof systems, *Arthur-Merlin games.* The game is played by the omniscient but untrustworthy wizard Merlin, and king Arthur, who may flip coins, but otherwise his computational power is polynomially limited. The players exchange messages in turn, and the goal for Merlin is to convince mistrustful Arthur to accept the input string. MA is the lowest level of the hierarchy, where the game is restricted to one round, with Merlin playing first.

Definition 4.26 A promise problem L is in *promise* MA (prMA for short), if there exists a probabilistic poly-time algorithm $A(x, y)$ such that

$$A(x, y) \Rightarrow |y| \le p(|x|)$$

for some polynomial p, and

$$x \in L^+ \Rightarrow \exists y \; \Pr(A(x, y)) \ge 3/4,$$
$$x \in L^- \Rightarrow \forall y \; \Pr(A(x, y)) \le 1/4.$$

A language is in MA if the corresponding promise problem is in prMA.

Definition 4.27 (in $PV_1 + sWPHP(PV)$) Let β be a PV-function with values in $(0, 1/2)$, A a PV-predicate, and q, r PV-functions. The triple $\langle A, q, r\rangle$ *β-defines a* prMA-*problem* $L = \langle L^+, L^-\rangle$ if $L^+ \supseteq L^{+\exists}_{A,q,r,\beta}$ and $L^- \supseteq L^{-\forall}_{A,q,r,\beta}$, where

$$x \in L^{+\exists}_{A,q,r,\beta} \iff \exists y \le q(x) \; \Pr_{w<r(x)}(\neg A(x, y, w)) \preceq_0 \beta(x),$$
$$x \in L^{-\forall}_{A,q,r,\beta} \iff \forall y \le q(x) \; \Pr_{w<r(x)}(A(x, y, w)) \preceq_0 \beta(x).$$

$\langle A, r, s\rangle$ β-defines an MA-language, if $\forall x\, (x \in L^{+\exists}_{A,r,s,\beta} \vee x \in L^{-\forall}_{A,r,s,\beta})$. If unspecified, we take $\beta = 1/4$.

Corollary 4.28 (*in* $PV_1 + sWPHP(PV)$) *Let t and s be as in Proposition 4.11, and let $L = \langle L^+, L^-\rangle$ be a promise problem. The following are equivalent.*

(i) *L is a $(1/2 - 1/|t|)$-definable* prMA-*problem,*

(ii) *L is a $1/4$-definable* prMA-*problem,*

(iii) *L is a $1/s$-definable* prMA-*problem.*

Moreover, every definable prMA-*problem is in* (*the natural formalization of*) prNP/poly.

Proof: This follows from Proposition 4.11 and Lemma 4.10, as the definable prMA-problems are just existentially quantified definable prBPP-problems. □

Trivially, every prMA-problem is representable in $PV_1 + sWPHP(PV)$. For MA-languages, we again have a reduction to a Σ^b_1-problem.

Proposition 4.29 *Let A be an* MA*-algorithm. There exists a true $\forall\Sigma^b_1$-sentence φ such that $PV_1 + sWPHP(PV)$ proves*

(i) *if φ, then A 1/3-defines an* MA*-language,*

(ii) *if A 1/4-defines an* MA*-language, then φ.*

Proof: We may take the formula

$$\varphi = \forall x \, \forall f \, \forall y \, \exists z \, \big(\neg \operatorname{Hard}^A_{1/4}(f) \vee \operatorname{Pr}^f_w(A(x,y,w))_{1/50} \le 7/24 \vee \operatorname{Pr}^f_w(\neg A(x,z,w))_{1/50} \le 7/24\big)$$

as in Theorem 4.13. □

We do not know whether the NP search problem associated with φ is solvable in probabilistic polynomial time. It holds at least for languages from $\mathrm{NP}^{\mathrm{BPP}} \subseteq \mathrm{MA}$.

Proposition 4.30 *Let A, q and r be PV-functions. There are Σ^b_2-formulas $\sigma^+(x), \sigma^-(x)$ and Π^b_2-formulas $\pi^+(x), \pi^-(x)$ such that $PV_1 + sWPHP(PV)$ proves*

$$x \in L^{+\exists}_{A,q,r,1/4} \to \pi^+(x) \to \sigma^+(x) \to x \in L^{+\exists}_{A,q,r,1/3},$$
$$x \in L^{-\forall}_{A,q,r,1/4} \to \pi^-(x) \to \sigma^-(x) \to x \in L^{-\forall}_{A,q,r,1/3}.$$

In particular, any definable MA*-language is in $\Sigma^b_2 \cap \Pi^b_2$.*

Proof: Similar to Proposition 4.14. The extra quantifiers do no harm:

$$\begin{aligned}
\pi^+(x) &= \forall f \, \big(\neg \operatorname{Hard}^A_{1/4}(f) \vee \exists y \le q(x) \, \operatorname{Pr}^f_{w<r(x)}(\neg A(x,y,w))_{1/50} \le 7/24\big),\\
\sigma^+(x) &= \exists f \, \exists y \le q(x) \, \big(\operatorname{Hard}^A_{1/4}(f) \wedge \operatorname{Pr}^f_{w<r(x)}(\neg A(x,y,w))_{1/50} \le 7/24\big),\\
\pi^-(x) &= \forall f \, \forall y \le q(x) \, \big(\neg \operatorname{Hard}^A_{1/4}(f) \vee \operatorname{Pr}^f_{w<r(x)}(A(x,y,w))_{1/50} \le 7/24\big),\\
\sigma^-(x) &= \exists f \, \big(\operatorname{Hard}^A_{1/4}(f) \wedge \forall y \le q(x) \, \operatorname{Pr}^f_{w<r(x)}(A(x,y,w))_{1/50} \le 7/24\big),
\end{aligned}$$

where f is bounded as in Proposition 4.14. □

5 Relativization and AM

The content of Section 3 can be relativized in a straightforward way: we work with $PV(R)$ instead of PV, where R is a new predicate, and we replace circuits with oracle circuits. The relativized version of Theorem 3.7 then provides approximate counting of sets defined by oracle circuits in $PV_1(R) + sWPHP(PV(R))$. The other results relativize in a similar way.

In particular, counting of sets higher in the polynomial hierarchy may be achieved by substitution of Σ^b_i-predicates for R. Namely, approximate counting of $\mathrm{P}^{\Sigma^b_i}$-definable sets (or more generally, sets defined by circuits with Σ^b_i oracles) is possible in $T^i_2 + sWPHP(\mathrm{FP}^{\Sigma^b_i}) \subseteq T^{i+2}_2$.

Relativization of Section 4 provides the formalization of $\mathrm{FRP}^{\Sigma^b_i}$, $\mathrm{prBPP}^{\Sigma^b_i}$, $\mathrm{APP}^{\Sigma^b_i}$, and $\mathrm{prMA}^{\Sigma^b_i}$ in $T^i_2 + sWPHP(\mathrm{FP}^{\Sigma^b_i})$.

Approximate counting of NP sets also permits formalization of Babai's class AM [16], which is defined by one-round Arthur-Merlin games where Arthur plays first.

Definition 5.1 A promise problem L is in *promise* AM (prAM for short), if there exists a probabilistic poly-time algorithm $A(x, y)$ such that

$$A(x, y) \Rightarrow |y| \le p(|x|)$$

for some polynomial p, and

$$\begin{aligned} x \in L^+ &\Rightarrow \Pr(\exists y\, A(x, y)) \ge 3/4,\\ x \in L^- &\Rightarrow \Pr(\exists y\, A(x, y)) \le 1/4. \end{aligned}$$

A language is in AM if the corresponding promise problem is in prAM.

Definition 5.2 (in $T^1_2 + sWPHP(PV_2)$) Let β be a PV-function with values in $(0, 1/2)$. A pair $\langle\varphi, r\rangle$, where $\varphi(x, w)$ is a Σ^b_1-formula, and r is a PV-function, *β-defines a* prAM *problem* $L = \langle L^+, L^-\rangle$ if $L^+ \supseteq L^+_{\varphi,r,\beta}$ and $L^- \supseteq L^-_{\varphi,r,\beta}$, where

$$\begin{aligned} x \in L^+_{\varphi,r,\beta} &\iff \Pr_{w<r(x)}(\neg\varphi(x, w)) \preceq^1_0 \beta(x),\\ x \in L^-_{\varphi,r,\beta} &\iff \Pr_{w<r(x)}(\varphi(x, w)) \preceq^1_0 \beta(x), \end{aligned}$$

and $\preceq^i_\varepsilon$ denotes $\preceq_\varepsilon$ relativized with a Σ^b_i-complete oracle. The pair $\langle\varphi, r\rangle$ *β-defines an* AM-*language,* if $\forall x\, (x \in L^+_{\varphi,r,\beta} \vee x \in L^+_{\varphi,r,\beta})$.

If unspecified, we take $\beta = 1/4$.

Proposition 5.3 (*in* $T^1_2 + sWPHP(PV_2)$) *Let t and s be as in Proposition 4.11, and let $L = \langle L^+, L^-\rangle$ be a promise problem. The following are equivalent:*

(i) *L is a $(1/2 - 1/|t|)$-definable* prAM-*problem,*

(ii) *L is a $1/4$-definable* prAM-*problem,*

(iii) *L is a $1/s$-definable* prAM-*problem.*

Proof: As $\mathrm{prAM} \subseteq \mathrm{prBPP}^{\mathrm{NP}}$, the result follows from relativization of Proposition 4.11, observing that the formula $\varphi'(x, w)$ defined by

$$\bigl|\{i < m(x) \mid \varphi(x, w_i)\}\bigr| \ge m(x)/2$$

is Σ^b_1, as it is equivalent to

$$\exists I \subseteq m(x)\, \bigl(|I| \ge m(x)/2 \wedge \forall i \in I\, \varphi(x, w_i)\bigr). \qquad \square$$

Babai's Collapse Theorem [16] states that AM coincides with the class of languages recognized by an Arthur-Merlin protocol with a bounded number of rounds. It is not clear how to define general Arthur-Merlin games in bounded arithmetic; the next theorem shows that prMAM = prAM, which implies that any class obtained by a constant number of applications of the $\exists$ and BP operators to prP is contained in prAM.

Theorem 5.4 *(in $T^1_2 + sWPHP(PV_2)$) Let $L = \langle L^+, L^-\rangle$ be a 1/4-definable* prAM*-problem, and q a PV-function. Define a promise problem $L^\exists = \langle L^{+\exists}, L^{-\forall}\rangle$ by*

$$\begin{aligned} x \in L^{+\exists} &\iff \exists y < q(x)\,\langle x, y\rangle \in L^+, \\ x \in L^{-\forall} &\iff \forall y < q(x)\,\langle x, y\rangle \in L^-. \end{aligned}$$

Then $L^\exists$ is a 1/4-definable prAM*-problem.*

In particular, every definable prMA*-problem is a definable* prAM*-problem.*

Proof: By Proposition 5.3, there exists a $1/(4q(x))$-definition $\langle \varphi, r\rangle$ of L. Define

$$\varphi'(x, w) \iff \exists y < q(x)\,\varphi(x, y, w).$$

Then $\langle \varphi', r\rangle$ is a 1/4-definition of $L^\exists$: if $x \in L^{+\exists}$, there is $y < q(x)$ such that $\Pr_w(\neg\varphi(x, y, w)) \preceq^1_0 1/(4q(x))$, and a fortiori

$$\Pr_w(\neg\exists y < q(x)\,\varphi(x, y, w)) \preceq^1_0 1/(4q(x)) \leq 1/4.$$

Assume $x \in L^{-\forall}$. Then $\Pr_w(\varphi(x, y, w)) \preceq^1_0 1/(4q(x))$ for every $y < q(x)$, and we would like to argue that

$$\Pr_w(\exists y < q(x)\,\varphi(x, y, w)) \preceq^1_0 1/4.$$

We cannot do it directly (say, by application of Proposition 3.19), as $q(x) \notin \mathrm{Log}$ in general, but we can explore the fact that the proof of Proposition 5.3 is sufficiently uniform.

We work in the relativized version of $HARD^A$, which we denote $HARD^A(\Sigma^b_1)$. Let $\langle \psi, s\rangle$ be a 1/6-definition of L, and we assume that $\langle \varphi, r\rangle$ was constructed from $\langle \psi, s\rangle$ as in Proposition 5.3. Keep x fixed. By the relativization of Lemma 3.14 there exist v and $\mathrm{FP}^{\alpha,\Sigma^b_1}$-functions f, g, h such that

$$\begin{gathered} f(y) < 1/6, \\ g(y, -)\colon v(1/50 + f(y))s(x) \twoheadrightarrow v \times \{w < s(x) \mid \psi(x, y, w)\}, \\ h(y, -)\colon v(1/50 + 1 - f(y))s(x) \twoheadrightarrow v \times \{w < s(x) \mid \neg\psi(x, y, w)\} \end{gathered}$$

for all $y < q(x)$. By the proof of Proposition 5.3 and the relativization of Proposition 3.18 there exists a v' and an $\mathrm{FP}^{\alpha,\Sigma^b_1}$-function g' such that

$$g'(y, -)\colon v'(r(x)/(4q(x))) \twoheadrightarrow v' \times \{w < r(x) \mid \varphi(x, y, w)\}$$

for all $y < q(x)$. We define $g''(u) = g'(u \bmod q(x), \lfloor \frac{u}{q(x)} \rfloor)$, and observe that

$$g''\colon v'(r(x)/4) \twoheadrightarrow v' \times \{w < r(x) \mid \exists y < q(x)\,\varphi(x, y, w)\},$$

thus

$$\Pr_w(\exists y < q(x)\,\varphi(x, y, w)) \preceq^1_0 1/4. \qquad \square$$

Proposition 5.5 (*in $T^1_2 + sWPHP(PV_2)$*) *$1/4$-definable* prAM*-problems are in* prNP/poly. *I.e., if $L = \langle L^+, L^-\rangle$ is a $1/4$-definable* prAM*-problem, and $n \in \mathrm{Log}$, then there exists a poly-size nondeterministic circuit $C\colon 2^n \to 2$ such that*

$$x \in L^+ \Rightarrow C(x) = 1,$$
$$x \in L^- \Rightarrow C(x) = 0$$

for every $x < 2^n$.

Proof: Let $\langle\varphi, r\rangle$ be a $1/4$-definition of L. Using twice the relativized version of Lemma 4.10, there exists a circuit $D\colon 2^n \to \{0, 1, *\}$ with an NP-oracle such that

$$x \in L^+_{\varphi,r,1/4} \to D(x) = 1 \to x \in L^+_{\varphi,r,1/3},$$
$$x \in L^-_{\varphi,r,1/4} \to D(x) = 0 \to x \in L^-_{\varphi,r,1/3}.$$

for every $x < 2^n$. Let $\langle\psi, s\rangle$ be a 2^{-2n}-definition of $L_{\varphi,r,1/3}$, available by Proposition 5.3. For simplicity, we may assume that $s(x) = s$ is constant for all $x < 2^n$. Then

$$\Pr_{w<s}\big((D(x) = 1 \wedge \neg\psi(x, w)) \vee (D(x) = 0 \wedge \psi(x, w))\big) \preceq^1_0 2^{-2n}$$

for every $x < 2^n$. Using the uniformity of the proof of Propositions 5.3 and 3.18, we obtain

$$\Pr_w\big(\exists x < 2^n\, (D(x) = 1 \wedge \neg\psi(x, w)) \vee (D(x) = 0 \wedge \psi(x, w))\big) \preceq^1_0 1/2$$

by the same reasoning as in Theorem 5.4. By $sWPHP(\mathrm{FP}^{\Sigma^b_1})$ there exists $w < s$ such that

$$D(x) = 1 \to \psi(x, w),$$
$$D(x) = 0 \to \neg\psi(x, w)$$

for every $x < 2^n$, and then it suffices to define $C(x) \leftrightarrow \psi(x, w)$. □

As AM $\subseteq$ BPP$^{\mathrm{NP}}$, the relativized version of Proposition 4.14 implies that every definable AM-predicate is in $\Sigma^b_3 \cap \Pi^b_3$. We will formalize the stronger result AM $\subseteq$ coRP$^{\mathrm{NP}[1]} \subseteq \Pi^b_2$ from [16]. The proof is based on [122].

Theorem 5.6 (*in $T^1_2 + sWPHP(PV_2)$*) *Let $L = \langle L^+, L^-\rangle$ be a $1/4$-definable* prAM*-problem. There exists a Σ^b_1-formula $\varphi(x, y)$ and a PV-function $r(x)$ such that for every x,*

$$x \in L^+ \Rightarrow \forall y \le r(x)\, \varphi(x, y),$$
$$x \in L^- \Rightarrow \Pr_{y\le r(x)}(\varphi(x, y)) \preceq^1_0 1/2.$$

Proof: In Proposition 5.3, the number of random bits increases polynomially in the number of iterations, but the probability of error decreases exponentially. Thus there exists $\langle\psi, s\rangle$ which is a $1/(4|s(x)|)$-definition of L. We may assume $s(x)$ is a power of two. We define

$$r(x) := s(x)^{|s(x)|},$$
$$\varphi(x, y) \leftrightarrow \exists w < s(x)\, \forall i < |s(x)|\, \psi(x, w \oplus y_i),$$

where y is decomposed as a sequence of $|s(x)|$ numbers $y_i < s(x)$, and $\oplus$ is bitwise XOR.

Let $x \in L^+$, and fix $y < r(x)$. We have $\Pr_{w<s(x)}(\neg\psi(x,w)) \preceq^1_0 1/(4|s(x)|)$, and $- \oplus y_i$ is a poly-time computable involution on $2^{|s(x)|}$, thus

$$\Pr_w(\neg\psi(x, w \oplus y_i)) \preceq^1_0 \frac{1}{4|s(x)|}$$

for every $i < |s(x)|$. We obtain

$$\Pr_w(\exists i < |s(x)|\, \neg\psi(x, w \oplus y_i)) \preceq^1_{1/4} \frac{1}{4}$$

from relativization of Proposition 3.19, thus $\varphi(x, y)$ by $s\mathit{WPHP}(\mathrm{FP}^{\Sigma^b_1})$.

Let $x \in L^-$, and write $s = s(x)$. We have $\Pr_w(\psi(x,w)) \preceq^1_0 1/(4|s|) \le 1/4$, thus there exists a circuit C_1 with an NP oracle such that

$$C_1 \colon v(s/4) \twoheadrightarrow v \times \{u < s \mid \psi(x,u)\}$$

for some $v > 0$. As $w \oplus -$ is a poly-time involution, we have a circuit C_2 such that for any $w < s(x)$,

$$C_2(w,-) \colon v(s/4) \twoheadrightarrow v \times \{u < s \mid \psi(x, w \oplus u)\}.$$

We apply $|s|$ copies of C_2 in parallel to obtain a circuit C_3 such that

$$C_3(w,-) \colon v^{|s|}(s/4)^{|s|} \twoheadrightarrow v^{|s|} \times \{y < s^{|s|} \mid \forall i < |s|\, \psi(x, w \oplus y_i)\},$$

and rearranging the domain yields a circuit C_4 such that

$$C_4 \colon v^{|s|}(s/4)^{|s|}s \twoheadrightarrow v^{|s|} \times \{y < s^{|s|} \mid \exists w < s\, \forall i < |s|\, \psi(x, w \oplus y_i)\},$$

thus

$$\Pr_y(\varphi(x,y)) \preceq^1_0 \frac{s}{4^{|s|}} \le \frac{1}{2}. \qquad \square$$

Acknowledgement

The research was done while the author was visiting the Department of Computer Science of the University of Toronto. Supported by NSERC Discovery grant, grant IAA1019401 of GA AV ČR, and grant 1M0545 of MŠMT ČR.

Chapter II

Approximate counting by hashing in bounded arithmetic

Abstract

We show how to formalize approximate counting via hash functions in subsystems of bounded arithmetic, using variants of the weak pigeonhole principle. We discuss several applications, including a proof of the tournament principle, and an improvement on the known relationship of the collapse of the bounded arithmetic hierarchy to the collapse of the polynomial-time hierarchy.

1 Introduction

Counting the number of elements of a finite set is one of the most fundamental operations in discrete mathematics. However, exact counting is not available in weak systems of first-order arithmetic where exponentiation is not a total operation unless the polynomial hierarchy collapses, because of Toda's theorem [174]. This does not exclude the possibility of *approximate counting*, which is sufficient in many counting applications: we estimate the size of the set up to a negligible error (where the meaning of "negligible" depends on the context).

A popular way of simulating approximate counting arguments in bounded arithmetic is to apply variants of the weak pigeonhole principle, see e.g. [146, 152, 136]. A systematic approach was taken in Chapter I: we have proved in the theory $PV_1 + sWPHP(PV)$ (defined below in Section 2) that for any bounded set defined by a Boolean circuit, there exist suitable kind of surjective "counting functions" (also definable by circuits) which allow us to coherently define the approximate size of the set up to a polynomially small error. This framework admits smooth development of basic counting and probability arguments (including, e.g., the inclusion-exclusion principle, and the Chernoff bounds), and provides a suitable means to define and discuss randomized algorithms in bounded arithmetic. However, it also suffers from a significant defect: if X is a subset of $[0, a]$, we can only estimate the size of X up to an error polynomially small relative to a—the size of the ambient interval—rather than relative to the size of X itself. (One of the reasons being that the size of X is estimated by sampling it with a pseudorandom number generator.) Sufficiently "sparse" sets are thus indistinguishable from the empty set.

This precludes more sophisticated combinatorial counting arguments (in particular, inductive arguments such as in the proof of the Ramsey theorem), and it is at odds with what usually goes by the name "approximate counting" in theoretical computer science.

Sipser's Coding Lemma [166], which is an application of Carter–Wegman 2-universal families of hash functions [49], shows that the polynomial-time hierarchy is closed under a stronger form of approximate counting: if X is the finite set we want to count, and n is a parameter given in unary, we can find s such that $s \le |X| \le s(1+\varepsilon)$ for any $\varepsilon \le n^{-O(1)}$. Our aim is to show that Sipser's definition makes a well-behaved concept of approximate counting in bounded arithmetic. We work in the theory $T^1_2 + sWPHP(PV_2)$ (i.e., the one as before, but relativized with an NP-oracle; it is a subtheory of T^3_2), or in the slightly weaker theory $T^1_2 + rWPHP(PV_2)$ (see Section 2). The key technical result (which can be thought of as formalization of the Coding Lemma in bounded arithmetic) states that Sipser-style approximate counting in terms of hash functions is (more or less) equivalent to the existence of certain surjective functions (Corollary 3.5 and Theorem 3.8). Armed with this "implementation-independent" view of hashing, we are able to prove basic properties of counting (Section 3), such as the size of a disjoint union is (approximately) the sum of sizes of the summands.

In Section 4 we mention some applications, intended as examples demonstrating how the methods developed in Section 3 may be used to formalize counting arguments in bounded arithmetic. We solve an open problem of Krajíček, Pudlák, and Takeuti [55, 116] by showing that $T^1_2 + rWPHP(PV_2)$ (hence also T^3_2) proves Erdős's [77] tournament principle (Theorem 4.2). We also prove a generalization of the tournament principle (Theorem 4.3), which allows us to strengthen the results of [121, 39] showing that the collapse of the bounded arithmetic hierarchy implies collapse of the polynomial-time hierarchy (Theorem 4.6 and Corollary 4.7). We observe that approximate counting provides an approximate Euler characteristic (in the sense of Krajíček [118]) for models of $S_2(\alpha)$ (Theorem 4.10). We also include two applications from computational complexity: we formalize in bounded arithmetic Cai's [47] result $\mathrm{S}^P_2 \subseteq \mathrm{ZPP}^{\mathrm{NP}}$ (Theorem 4.11), and the existence of an AM-algorithm for graph nonisomorphism by Goldwasser and Sipser [83] (Theorem 4.12).

We remark that the "new" approximate counting method does not make the "old" counting of Chapter I superfluous: while the method in the present paper allows for better approximation (we can estimate the size of a set X up to an error which is a polynomially small fraction of $|X|$, rather than of the size of the ambient universe) which also agrees with the established usage of the term "approximate counting" in computer science, the price we pay is an increase in the complexity of the counting functions, which requires an increase of the strength of the base theory by one level of the bounded arithmetic hierarchy. To put it differently, $T^1_2 + sWPHP(PV_2)$ can count P^{NP}/poly-sets using the old method, but only NP/poly-sets using the new method. Moreover, some results from Chapter I are used in an essential way in Section 3.

2 Preliminaries

We assume some degree of familiarity with first-order bounded arithmetic, however the basic definitions are summarized below. More background can be found in [116, 40, 86].

Buss's theories are formulated in the language $L = \langle 0, S, +, \times, \leq, \#, |x|, \lfloor \frac{x}{2} \rfloor \rangle$. The intended meaning of the symbols are the usual arithmetical operations on non-negative integers, and $|x| = \lceil \log_2(x+1) \rceil$, $x \# y = 2^{|x|\cdot|y|}$. *Bounded quantifiers* are introduced by

$$\exists x \leq t\, \varphi \Leftrightarrow \exists x\, (x \leq t \wedge \varphi),$$
$$\forall x \leq t\, \varphi \Leftrightarrow \forall x\, (x \leq t \to \varphi),$$

where t is a term without an occurrence of the variable x. Such a quantifier is *sharply bounded*, if t has the form $|s|$ for some term s. A formula φ is bounded (sharply bounded) if all quantifiers in φ are bounded (sharply bounded). A formula is Σ^b_1 if it is constructed from sharply bounded formulas by means of $\wedge$, $\vee$, sharply bounded, and existential bounded quantifiers. In general, Σ^b_i-formulas consist of i alternating blocks of bounded quantifiers followed by a sharply bounded formula, where the first block is existential, and we ignore sharply bounded quantifiers which are allowed to appear anywhere in the quantifier prefix. Π^b_i-formulas are defined similarly, but the first block is universal; in other words, Π^b_i-formulas are negations of Σ^b_i-formulas. The class of Boolean combinations of Σ^b_i-formulas is denoted by $\mathcal{B}(\Sigma^b_i)$. Bounded formulas capture the polynomial-time hierarchy (PH). More precisely, for any $i \geq 1$ the class Σ^P_i coincides with sets of natural numbers definable by Σ^b_i-formulas in $\mathbb{N}$ (the standard model of arithmetic), and dually $\Pi^P_i = \Pi^b_i(\mathbb{N})$, in particular $\mathrm{NP} = \Sigma^b_1(\mathbb{N})$.

The theory T^i_2 is axiomatized by a finite set of open axioms denoted by *BASIC*, which state elementary properties of the symbols of L, and the schema of *induction*

$$(\mathit{IND}) \qquad \varphi(0) \wedge \forall x < a\, (\varphi(x) \to \varphi(x+1)) \to \varphi(a)$$

for Σ^b_i-formulas φ. The theory S^i_2 is axiomatized over *BASIC* by the *polynomial induction* schema

$$(\mathit{PIND}) \qquad \varphi(0) \wedge \forall x \leq a\, (\varphi(\lfloor \tfrac{x}{2} \rfloor) \to \varphi(x)) \to \varphi(a)$$

for Σ^b_i-formulas φ. Alternatively, S^i_2 can be axiomatized over *BASIC* by the *length induction* schema

$$(\mathit{LIND}) \qquad \varphi(0) \wedge \forall x < |a|\, (\varphi(x) \to \varphi(x+1)) \to \varphi(|a|),$$

or by the *length maximization* schema

$$(\mathit{LMAX}) \qquad \varphi(|a|) \to \exists b \leq |a|\, (\varphi(b) \wedge \forall c < b\, \neg\varphi(c))$$

for Σ^b_i-formulas φ. We have $S^i_2 \subseteq T^i_2 \subseteq S^{i+1}_2$, the full bounded arithmetic is thus $S_2 = \bigcup_i S^i_2 = \bigcup_i T^i_2$. The theory S^{i+1}_2 is $\forall\Sigma^b_{i+1}$-conservative over T^i_2 by Buss's witnessing theorem [37] (in the case of $i = 0$ we need a minor adjustment of the language of T^0_2, see [94]).

PV is an equational theory introduced by Cook [63]. Its language contains function symbols for all polynomial-time algorithms, introduced inductively using limited recursion on notation (cf. Cobham [57]). It is axiomatized by defining equations of its function symbols, and a derivation rule similar to *PIND*. PV_1, also known as QPV, $T^0_2(\Box^p_1)$, or $\forall\Sigma^b_1(S^1_2)$, is a first-order variant of PV. It can be axiomatized by equations provable in PV together with the axioms $0 \neq 1$ and $\lfloor \frac{x}{2} \rfloor = 0 \to x = 0 \lor x = 1$, and it proves the *PIND* and *IND* schemata for sharply bounded formulas. We will also use the symbol PV to denote the set of function symbols of PV.

The theories PV_{i+1} for $i > 0$, introduced in [121], are defined similarly to PV_1, except that the basic functions of their language include the characteristic functions of all Σ^b_i-predicates, thus PV_{i+1}-functions correspond to $\mathrm{FP}^{\Sigma^P_i}$ in the standard model. Again, we will also use PV_{i+1} to denote the class of PV_{i+1}-functions. The class of PV_{i+1}-predicates (which corresponds to Δ^P_{i+1} in the standard model) is denoted by Δ^b_{i+1}; it coincides with predicates provably $\Sigma^b_{i+1} \cap \Pi^b_{i+1}$ in either T^i_2 or S^{i+1}_2.

As PV_{i+1} is a conservative extension of T^i_2 by definitions, we will simply identify PV_{i+1} with T^i_2, and work freely with PV_{i+1}-functions in T^i_2. The theory $S^1_2(PV_{i+1})$, which is axiomatized by PV_{i+1} and $\Sigma^b_1(PV_{i+1})$-*PIND*, is a conservative extension of S^{i+1}_2 for the same reason, hence we will also identify $S^{i+1}_2 = S^1_2(PV_{i+1})$.

All these theories can be *relativized* in a straightforward way. We include a new predicate α in the language, and define $\Sigma^b_i(\alpha)$ as before, but allowing α to be used in atomic formulas. The theory $T^i_2(\alpha)$ consists of *BASIC* and $\Sigma^b_i(\alpha)$-*IND* (i.e., there are no axioms involving α apart from the induction axioms), and similarly $S^i_2(\alpha) = \mathit{BASIC} + \Sigma^b_i(\alpha)$-*PIND*. In the case of $PV(\alpha)$ and $PV_i(\alpha)$, we allow the characteristic function of α to appear in functions constructed by limited recursion on notation, so that function symbols of $PV(\alpha)$ correspond to polynomial-time oracle algorithms. More generally, if Γ is a set of formulas, we define $\Sigma^b_i(\Gamma)$, $T^i_2(\Gamma)$, $S^i_2(\Gamma)$, and $PV_i(\Gamma)$ by substituting Γ-formulas for α in $\Sigma^b_i(\alpha)$, $T^i_2(\alpha)$, $S^i_2(\alpha)$, and $PVi(\alpha)$. (Notice that $T^1_2(\Sigma^b_i) = T^{i+1}_2$, and so on.) The main point of relativization is that this kind of substitution preserves derivability. Hence, e.g., if we prove a statement about counting of $\Sigma^b_1(\alpha)$-sets in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$, it also applies to counting of Σ^b_i-sets in $T^i_2 + sWPHP(PV_{i+1})$ for every $i > 0$.

The *choice* schema (aka bounded collection, or replacement) $BB\Gamma$ for a set of formulas Γ is defined by

$$\forall x < |a|\, \exists y \leq b\, \varphi(x,y) \to \exists w\, \forall x < |a|\, \varphi(x,(w)_x),$$

where $\varphi \in \Gamma$, and $(w)_x$ denotes the xth member of the sequence encoded by w. $BB\Sigma^b_i(\alpha)$ is provable in $S^i_2(\alpha)$.

For any functions f, g, the surjective (also called dual), injective, and retraction pigeonhole principles are defined by

$$\begin{aligned} sPHP^a_b(f) &\Leftrightarrow \exists v < b\, \forall u < a\, f(u) \neq v, \\ iPHP^a_b(g) &\Leftrightarrow \exists v < b\, g(v) \geq a \lor \exists v < b\, \exists v' < v\, g(v) = g(v'), \\ rPHP^a_b(f,g) &\Leftrightarrow \exists v < b\, (g(v) \geq a \lor f(g(v)) \neq v). \end{aligned}$$

(Recall that a *retraction pair* is a pair of functions f, g such that $f \circ g = \mathrm{id}$; the function f is called a retraction, and g is its coretraction.) Note that the functions f, g may involve other parameters not explicitly shown. The *weak pigeonhole principles* are defined by

$$?\,\mathit{WPHP}(f) = \forall\bigl(x > 0 \to ?\mathit{PHP}^{x|y|}_{x(|y|+1)}(f)\bigr),$$
$$?\,\mathit{WPHP}(\Gamma) = \{?\,\mathit{WPHP}(f) : f \in \Gamma\},$$

where $\forall$ denotes universal closure, Γ is a set of functions, and $? \in \{s, i, r\}$. In the case of $r\mathit{WPHP}(PV(\Gamma))$ and $i\mathit{WPHP}(PV(\Gamma))$, the principles thus introduced are equivalent to the more usual variants with bounds $?\mathit{PHP}^x_{2x}$ or $?\mathit{PHP}^x_{x^2}$ over $PV_1(\Gamma)$. This however does not hold for $s\mathit{WPHP}$ (we need $S^1_2(\Gamma)$ to prove the equivalence, see [95] for details), we thus need to state the principle in the strong form as above.

As $T^i_2(\alpha)$ proves that every $PV_{i+1}(\alpha)$-function is on a bounded domain computable by a polynomial-size circuit with a $\Sigma^b_i(\alpha)$-oracle, the schema $?\,\mathit{WPHP}(PV_{i+1}(\alpha))$ is over $T^i_2(\alpha)$ equivalent to its single instance where f is the evaluation function for $\Sigma^b_i(\alpha)$-oracle circuits, for any ?.

Clearly $r\mathit{WPHP}(f, g)$ follows from either $s\mathit{WPHP}(f)$ or $i\mathit{WPHP}(g)$. The weak pigeonhole principles $s\mathit{WPHP}(f)$ and $i\mathit{WPHP}(f)$ are provable in $T^2_2(f)$ [146, 116, 125] (in particular, $s\mathit{WPHP}(PV_2(\alpha))$ is contained in $T^3_2(\alpha)$), but no variant of WPHP is provable in $S^2_2(f)$ [115, 160]. We will often use (for $i = 1$) the following connection between $r\mathit{WPHP}$ and $s\mathit{WPHP}$, which follows by relativization of [93, Cor. 1.15, 4.12].

Theorem 2.1 *The theory $S^{i+1}_2(\alpha) + BB\Sigma^b_{i+2}(\alpha) + s\mathit{WPHP}(PV_{i+1}(\alpha))$ is $\forall\Sigma^b_{i+1}(\alpha)$-conservative over $T^i_2(\alpha) + r\mathit{WPHP}(PV_{i+1}(\alpha))$ for any $i \geq 0$.* □

We will often work with *bounded definable sets*, which are collections of numbers of the form

$$X = \{x < a : \varphi(x)\},$$

where φ is a formula. Bounded sets are *not* genuine objects in our arithmetical theories, but a figure of speech: $x \in X$ is an abbreviation for $x < a \wedge \varphi(x)$. We will write $X \in \Sigma^b_1(\alpha)$ if X is a bounded set defined by a $\Sigma^b_1(\alpha)$-formula. When used in a context which asks for a set, a number a is assumed to represent the integer interval $[0, a)$; thus, for example, $X \subseteq a$ means that all elements of X are less than a. We will use simple set-theoretic operations, whose meaning should be generally clear from the context; for example, if $X \subseteq a$ and $Y \subseteq b$, we may define

$$X \times Y = \{bx + y : x \in X, y \in Y\} \subseteq ab,$$
$$X \mathbin{\dot\cup} Y = X \cup \{y + a : y \in Y\} \subseteq a + b.$$

On the other hand, we will occasionally (especially in the applications) need to refer to "small" sets directly encoded by a number. They should be distinguishable from definable sets by the context; in particular, by the absence of a complexity measure (as in "a $\Sigma^b_1(\alpha)$-set"). If X is such a small set, we denote by $|X|$ its cardinality, defined in a natural way (e.g., as in Corollary 3.10). This notation should not be confused with the length (or logarithm) function $|a|$ from the basic language of bounded arithmetic.

Due to general absence of BB in our base theory, we will often need to work with a strengthened notion of surjectivity. We will call a function $f\colon X \to Y$ a *smooth surjection*, written as

$$f\colon X \twoheadrightarrow Y,$$

if for every sequence w of elements of Y, there exists a sequence v of elements of X, such that $\mathrm{lh}(v) = \mathrm{lh}(w)$, and $f(v_i) = w_i$ for every $i < \mathrm{lh}(v)$, where $\mathrm{lh}(v)$ denotes the length of the sequence v. (Note that the length of w is implicitly polynomially bounded as $\mathrm{lh}(w) \le |w|$, but we do not impose other restrictions on it.) We also extend the definition so that the empty *partial* function is considered a smooth surjection from any set X on the empty set Y. In many situations, a surjection is automatically smooth (in particular, we will often use (i) without explicit mention):

Observation 2.2 *(in $T^1_2(\alpha)$) Let $X \in \Sigma^b_1(\alpha)$. A surjective $PV_2(\alpha)$-function $f\colon X \to Y$ is smooth whenever at least one of the following holds:*

(i) *f has a $PV_2(\alpha)$-coretraction,*

(ii) *f has a $\Sigma^b_1(\alpha)$-graph,*

(iii) *$BB\Sigma^b_2(\alpha)$,*

(iv) *f is a composition of two smooth surjections.* □

(Note in particular that all surjections are smooth in the standard model of arithmetic. Smoothness is only a technical condition needed to compensate for the lack of appropriate instances of BB.) We will write just $X \twoheadrightarrow Y$ if there exists a function $f\colon X \twoheadrightarrow Y$ (of suitable complexity, which should be clear from the context).

We will use the shorthand notation

$$x \in \mathrm{Log} \Leftrightarrow \exists y\, x = |y|.$$

We will also work with rational numbers, which are assumed to be represented by pairs of integers in a natural way. The expression $x^{-1} \in \mathrm{Log}$ is a shorthand notation meaning that x is a positive rational number, whose inverse is bounded above by an integer $n \in \mathrm{Log}$. The symbol $\mathbb{Q}_{\mathrm{Log}}$ denotes the set of rationals whose nominator and denominator belong to Log.

Many of our results take place *inside* formal theories like $T^1_2 + rWPHP(PV_2)$. If T is a theory, a parenthesized expression "in T" in the heading of a definition or theorem indicates that the definition is introduced in T, or that the theorem is formulated and proved inside T. However, we will slightly abuse this convention for reasons of compactness: when we write e.g. "for every PV_2-function f ..." in a formalized context, it is assumed that the quantification over PV_2-functions takes place in the metatheory, and only *parameters* of the function are quantified inside T. Formulas, definable sets, and other non-first-order objects are treated similarly.

In fact, in most cases the sets or functions thus quantified will only have a bounded domain. As already mentioned above, speaking of (say) $PV_2(\alpha)$-functions, or $\Sigma^b_1(\alpha)$-sets in such a context is equivalent to using circuits with a $\Sigma^b_1(\alpha)$-oracle, or non-deterministic Boolean circuits with an oracle α, respectively. We will, however, generally use the former expression, as we believe it is easier to read (even though the latter may be formally more correct). We point out that the reader should think about Σ^b_1-sets as corresponding to NP/poly, rather than just NP as is usual in bounded arithmetic.

We will also need some notation and results from Chapter I. For convenience, we state it in a relativized version (which is the one we will actually use); in particular, what we denote $\preceq_\varepsilon$ below is closer to what is denoted by $\preceq^1_\varepsilon$ in Chapter I.

Let $X, Y \subseteq a$ be definable sets, and $\varepsilon \le 1$. We say that the size of X is approximately less than the size of Y with error ε, written as $X \preceq_\varepsilon Y$, if there exists a $PV_2(\alpha)$-function C, and $v \neq 0$, such that

$$C\colon v \times (Y \mathbin{\dot\cup} \varepsilon a) \twoheadrightarrow v \times X.$$

The sets X and Y have approximately the same size with error ε, written as $X \approx_\varepsilon Y$, if $X \preceq_\varepsilon Y$ and $Y \preceq_\varepsilon X$. (We recall that we identify a number s with the interval $[0, s)$, thus as a special case, $X \approx_\varepsilon s$ means that the size of X is equal to s with error ε.)

If p is a rational, we also write

$$\Pr_{x<a}(\varphi(x)) \preceq_\varepsilon p \iff \{x < a : \varphi(x)\} \preceq_\varepsilon pa,$$

and similarly for $\succeq$, $\approx$. We will often omit the mention of a when it is clear from context. For example, a sequence $\vec{A} = \langle A_i : i < k\rangle$ of $t \times n$ binary matrices is encoded by a number $x < 2^{ktn}$, hence we write $\Pr_{\vec{A}}(\varphi(\vec{A}))$ instead of $\Pr_{x<2^{ktn}}(\varphi(\text{the sequence of matrices encoded by } x))$.

Theorem 2.3 (Thm. I.3.7) *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) Let $X \subseteq a$, $X \in \Delta^b_2(\alpha)$, and $\varepsilon^{-1} \in \mathrm{Log}$. There exists a number $s \le a$ such that $X \approx_\varepsilon s$, moreover the surjections required by the definition of $\approx$ have $PV_2(\alpha)$-coretractions, and the numbers v from the definition belong to* Log*.* □

The reader may find it helpful to familiarize her/himself with basic properties of $\preceq_\varepsilon$ from §I.3.

We will occasionally use some results from Chapter I on definable randomized algorithms, in particular, AM. Recall that a promise problem is a pair $L = \langle L^+, L^-\rangle$ of disjoint sets of strings (a language $L \subseteq \Sigma^*$ is identified with the promise problem $\langle L, \Sigma^* \smallsetminus L\rangle$). A promise problem L is in *promise* AM(α) (prAM(α) for short), if there exists a probabilistic polynomial-time algorithm $A(x, y)$ with oracle α such that

$$A(x, y) \Rightarrow |y| \le p(|x|)$$

for some polynomial p, and

$$x \in L^+ \Rightarrow \Pr(\exists y\, A(x, y)) \ge 3/4,$$
$$x \in L^- \Rightarrow \Pr(\exists y\, A(x, y)) \le 1/4.$$

A language is in AM(α) if the corresponding promise problem is in prAM(α).

We formalize this definition in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$ as follows. Let β be a PV-function with values in $(0, 1/2)$. A pair $\langle \varphi, r\rangle$, where $\varphi(x, w)$ is a $\Sigma^b_1(\alpha)$-formula, and r is a PV-function, *β-defines a* prAM(α) *problem* $L = \langle L^+, L^-\rangle$ if $L^+ \supseteq L^+_{\varphi,r,\beta}$ and $L^- \supseteq L^-_{\varphi,r,\beta}$, where

$$\begin{aligned} x \in L^+_{\varphi,r,\beta} &\iff \Pr_{w<r(x)}(\neg\varphi(x, w)) \preceq_0 \beta(x), \\ x \in L^-_{\varphi,r,\beta} &\iff \Pr_{w<r(x)}(\varphi(x, w)) \preceq_0 \beta(x). \end{aligned}$$

The pair $\langle \varphi, r\rangle$ *β-defines an* AM(α)*-language,* if $\forall x\,(x \in L^+_{\varphi,r,\beta} \vee x \in L^-_{\varphi,r,\beta})$. If unspecified, we take $\beta = 1/4$.

The definition is insensitive on the choice of β in the following sense: if t, s are PV-functions such that $t(x), s(x) > 0$ and $1/s(x) + 1/|t(x)| \le 1/2$, then $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$ proves that L is a definable prAM(α)-problem iff it is a $(1/2 - 1/|t|)$-definable prAM(α)-problem iff it is a $1/s$-definable prAM(α)-problem (Proposition I.5.3). We could also use an asymmetric definition with different bounds for L^+ and L^-, but we will not write it down explicitly.

We will need the following statement, formalizing the result that AM $\subseteq$ NP/poly.

Theorem 2.4 (Prop. I.5.5) *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) If L is a $1/4$-definable* prAM(α)-*problem, and $n \in$ Log, then there exists a polynomial-size nondeterministic oracle circuit $C\colon 2^n \to 2$ such that*

$$\begin{aligned} x \in L^+ &\Rightarrow C(x) = 1, \\ x \in L^- &\Rightarrow C(x) = 0 \end{aligned}$$

for every $x < 2^n$. □

If L_0, L_1 are definable prAM(α)-problems, it is easy to see that $L_0 \cap L_1 := \langle L^+_0 \cap L^+_1, L^-_0 \cup L^-_1\rangle$ and $L_0 \cup L_1 := \langle L^+_0 \cup L^+_1, L^-_0 \cap L^-_1\rangle$ are also definable prAM(α)-problems. More importantly, definable prAM(α)-problems are in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$ closed under bounded existential quantification (Theorem I.5.4): if q is a PV-function, and L is a definable prAM(α)-problem, then so is $L^\exists := \langle L^{+\exists}, L^{-\forall}\rangle$, where

$$\begin{aligned} x \in L^{+\exists} &\iff \exists y < q(x)\, \langle x, y\rangle \in L^+, \\ x \in L^{-\forall} &\iff \forall y < q(x)\, \langle x, y\rangle \in L^-. \end{aligned}$$

3 The toolbox

We begin with a definition of approximate counting based on Sipser [166]. Rather than defining what is a size of a set, we introduce a predicate $X \precsim_\varepsilon s$ which means that the size of X is bounded above by s (approximately, with relative error ε). (This $\precsim$ should not be confused with $\preceq$.)

The basis of the construction is to use linear hash functions. The idea is as follows. Let $X \subseteq 2^n$, $s = |X|$, and choose a parameter t. We consider a random linear function $A\colon 2^n \to 2^t$ (which is given by a matrix, thus a polynomial-size object). Ideally, we would like A to be injective on X, which would witness that $s \le 2^t$. This is rather unlikely to happen unless t is really huge, but it is possible that A is injective at least on a sizable part of X. Let thus X' be the set of all elements $x \in X$ such that $A(x) \neq A(y)$ for all $y \in X$ different from x, so that A is injective on X'. Elements of X' are called *separated* by A. The probability that $A(x) = A(y)$ for $x \neq y$ is 2^{-t}, hence any $x \in X$ is not separated by A with probability at most $s2^{-t} \leq 1/2$, as long as $2^t \geq 2s$. The expected size of X' is thus at least $s/2$. In order to cover all of X, we choose independently random linear functions $A_i \colon 2^n \to 2^t$ for $i < t$. The probability that $x \in X$ is not separated by A_i is at most $1/2$, hence the probability that it is not separated by any A_i, $i < t$, is at most 2^{-t}. The expected number of $x \in X$ not separated by any A_i is thus at most $s2^{-t} \leq 1/2$, hence there exist matrices $A_0, \dots, A_{t-1}$ such that *every* $x \in X$ is separated by some A_i. However, the existence of such $\vec{A}$ does not conversely guarantee that $|X| \leq s$. Each A_i injects a part of X to 2^t, hence we can inject X into $t2^t$. We may choose $2^t \leq 4s$, hence we obtain only an injection of X to $4s \log 4s$.

This form of hashing thus directly distinguishes sets of size s from roughly $4s \log s$. We want to distinguish size s from $s(1+\varepsilon)$ for polynomially small ε; we achieve this by considering Cartesian powers. Instead of our set X, we apply the hashing to X^c for some c. We can distinguish its size s^c from $4s^c \log s^c = 4cs^c \log s$, and the latter is less than $(s(1+\varepsilon))^c$ as long as $(1+\varepsilon)^c \geq 4c \log s$. We have $(1+\varepsilon)^{c_1} \geq 2$ for $c_1 \geq \varepsilon^{-1}$, and $2^{c_2} \geq 4c \log s$ for c_2 about $\log\log s + \log c$, hence it suffices to take roughly $c = \Omega(\varepsilon^{-1}(\log\log s + \log \varepsilon^{-1}))$. We will actually use somewhat larger (but still polynomial in ε^{-1} and $\log s$) c for convenience to simplify some computations below.

Definition 3.1 Let $X \subseteq 2^n$, and $x < 2^n$. A matrix $A \in 2^{t \times n}$ (i.e., a t-by-n matrix over $GF(2)$) *separates* x from X if $Ax \neq Ay$ for all $y \in X \smallsetminus \{x\}$ (where we view elements of 2^n as column vectors over $GF(2)$). A sequence $\vec{A} = \langle A_i : i < k \rangle$ of matrices *isolates* X, written as

$$\vec{A} \colon X \looparrowright 2^t,$$

if every $x \in X$ is separated from X by some A_i. Let $\varepsilon^{-1} \in \mathrm{Log}$. If $s > 0$, we write

$$X \precsim_\varepsilon s$$

if there exist $\langle A_i : i < t \rangle$ such that $\vec{A} \colon X^c \looparrowright 2^t$, where $c = 12|S|\lceil \varepsilon^{-1} \rceil^2$, and $t = |S^c| + 1$ for some $0 < S \leq s$. We also define $X \precsim_\varepsilon 0$ iff X is empty. We write $X \precsim s$ if $X \precsim_\varepsilon s$ for every $\varepsilon^{-1} \in \mathrm{Log}$.

Remark 3.2 If X is $\Sigma^b_1(\alpha)$, then the properties "A separates x from X", and "$\vec{A}$ isolates X" are $\Pi^b_1(\alpha)$, hence $X \precsim_\varepsilon s$ is $\Sigma^b_2(\alpha)$.

The definition makes $\precsim$ monotone: if $Y \subseteq X \precsim_\varepsilon s \leq t$, then $Y \precsim_\varepsilon t$.

If $X \subseteq 2^n$, $n < m$, $\vec{A} \in 2^{t \times m}$, and $\vec{B} \in 2^{t \times n}$ is the sequence of restrictions of A_i's to the first n columns, then $\vec{A}$ isolates X iff $\vec{B}$ does. The definition of $X \precsim_\varepsilon s$ thus does not depend on the choice of n.

A moment's reflection will persuade the reader that it is next to impossible to work directly with the hash functions. For example, if $\vec{A} \colon X \looparrowright 2^t$, and $\vec{B} \colon Y \looparrowright 2^t$, there is apparently no way of constructing $\vec{C}$ such that, say, $\vec{C} \colon X \cup Y \looparrowright 2^{t+1}$. In the real world, this is no problem as we have a well-behaved preexisting notion of cardinality, and we merely observe that the hashes agree with it. Obviously, this does not work in bounded arithmetic if we want to use the hashes to define (approximate) cardinality in the first place. We get around the problem by showing that $X \precsim_\varepsilon s$ is, up to ε, equivalent to the existence of suitable surjections from a power of s to a corresponding power of X; these surjections will be much easier to handle. The key result is Theorem 3.4 (the other direction will be much simpler), which is essentially a formalization of Sipser's Coding Lemma in bounded arithmetic.

Lemma 3.3 (*in* T^0_2) *If* $c \in \mathrm{Log}$*, there exists a* PV*-bijection*

$$f\colon \dot{\bigcup_{i\le c}} \binom{c}{i} \times X^i \times Y^{c-i} \simeq (X \mathbin{\dot\cup} Y)^c$$

with a PV*-inverse.*

Proof: Let $u < \binom{c}{i}$, $\langle x_j : j < i\rangle \in X^i$, and $\langle y_j : j < c-i\rangle \in Y^{c-i}$. We can enumerate subsets of c of size i by $\binom{c}{i}$, let thus $U \subseteq c$ be the uth set. Let $\langle \pi_j : j < i\rangle$ be an increasing enumeration of U, and $\langle \varrho_j : j < c-i\rangle$ an increasing enumeration of $c \smallsetminus U$. We define $f(u,\vec{x},\vec{y}) = \vec{z}$, where $z_{\pi_j} = x_j$, $z_{\varrho_j} = y_j$. It is easy to see that f is a bijection. □

Theorem 3.4 (*in* $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) *Let* $d, r > 0$*,* $d \in \mathrm{Log}$*, and* $f\colon rs^d \twoheadrightarrow r \times X^d$*, where* X *is* $\Sigma^b_1(\alpha)$*, and* f *is* $PV_2(\alpha)$*. Then there exists* $\langle A_i : i < t\rangle$ *such that* $\vec{A}\colon X \looparrowright 2^t$*, where* $t = |s| + 1$*, and moreover,*

$$\Pr_{\vec{A}}\bigl(\vec{A} \textit{ does not isolate } X\bigr) \preceq_0 2/3.$$

Proof: Let B be the set of sequences $\langle A_i : i < t\rangle$, $A_i \in 2^{t\times n}$, such that $\vec{A}$ does not isolate X. We define a PV-function

$$g_0\colon (2^n \smallsetminus \{0\}) \times 2^{n-1} \to 2^n$$

as follows. Let $i < n$ be the index of the first set bit of x. We decompose $w = w_0{}^\frown w_1$, where $w_0 < 2^i$ and $w_1 < 2^{n-i-1}$, and we put $g_0(x,w) = w_0{}^\frown b^\frown w_1$, where $b = x^{\mathrm{T}}(w_0{}^\frown 0^\frown w_1)$. Clearly, $g_0(x,\cdot)$ is a surjection of 2^{n-1} onto $\{a \in 2^n : a^{\mathrm{T}}x = 0\}$ whenever $x \neq 0$. Then we can define a PV-function

$$g\colon (2^n \smallsetminus \{0\}) \times 2^{(n-1)t} \to 2^{t\times n}$$

so that $g(x,\langle w_0,\dots,w_{t-1}\rangle)$ is the matrix A such that the jth row of A is $g_0(x,w_j)^{\mathrm{T}}$ for every $j < t$. It follows that

$$g(x,\cdot)\colon 2^{(n-1)t} \twoheadrightarrow \{A \in 2^{t\times n} : Ax = 0\}$$

for every $x \neq 0$.

We define a $PV_2(\alpha)$-function

$$h\colon r^{t+1}\bigl(s^{t+1}2^{(n-1)t^2}\bigr)^d \to r^{t+1}(2^{t\times n})^{td}$$

as follows. We interpret the input to h as sequence consisting of u, $\langle v_i : i < t\rangle$, and $\langle w_{i,j} : i < t, j < d\rangle$, where $u, v_i < rs^d$, $w_{i,j} < 2^{(n-1)t}$. We compute $f(u) = \langle p, x_j : j < d\rangle \in r \times X^d$, and in a similar way, $f(v_i) = \langle q_i, y_{i,j} : j < d\rangle$. For each $i < t$ and $j < d$, we define $A_{i,j} = g(x_j + y_{i,j}, w_{i,j})$ (where $+$ is vector addition) if $x_j \neq y_{i,j}$, and $A_{i,j} = 0$ otherwise. We let $\langle p, q_i, A_{i,j} : i < t, j < d\rangle$ be the output of h.

We claim that h is a surjection of $r^{t+1}\big(s^{t+1}2^{(n-1)t^2}\big)^d$ onto $r^{t+1} \times B^d$. Indeed, consider a sequence $\langle p, q_i, A_{i,j} : i < t, j < d\rangle \in r^{t+1} \times B^d$. For each $j < d$, there exists an $x_j \in X$ which is not separated from X by $\langle A_{i,j} : i < t\rangle$; we can collect them to a sequence $\langle x_j : j < d\rangle$ by $BB\Sigma^b_1(\alpha) \subseteq T^1_2(\alpha)$. Likewise, there exists a sequence $\langle y_{i,j} : i < t, j < d\rangle$ of witnesses to the non-separation of x_j by $A_{i,j}$, i.e., $y_{i,j} \neq x_j$, and $A_{i,j}x_j = A_{i,j}y_{i,j}$. The latter is equivalent to $A_{i,j}(x_j + y_{i,j}) = 0$, and as $x_j + y_{i,j} \neq 0$, we can use the properties of g to find a sequence $\langle w_{i,j} : i < t, j < d\rangle$ such that $g(x_j + y_{i,j}, w_{i,j}) = A_{i,j}$. As f is surjective, we can find a $u < rs^d$ such that $f(u) = \langle p, x_j : j < d\rangle$. We construct suitable v_i in a similar way, using smoothness of f. Then $h(u, \vec{v}, \vec{w}) = \langle p, \vec{q}, \vec{A}\rangle$.

As $s \leq 2^{t-1}$, we have

$$r^{t+1}\big(s^{t+1}2^{(n-1)t^2}\big)^d \leq 2^{-d}r^{t+1}2^{nt^2d},$$

thus $sWPHP(PV_2(\alpha))$ implies that h is not onto $r^{t+1} \times (2^{t\times n})^{td}$, hence $B \neq (2^{t\times n})^t$. Any $\vec{A} \in (2^{t\times n})^t \smallsetminus B$ isolates X.

As B is $\Sigma^b_1(\alpha)$, and we assume $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$, there exists a b such that $B \approx_{1/20} b$ by Theorem 2.3. By definition, there exists $0 < e \in \mathrm{Log}$, and a $PV_2(\alpha)$-surjection

$$e \times \left(B \mathbin{\dot\cup} \tfrac{1}{20}2^{nt^2}\right) \twoheadrightarrow eb.$$

For any $k \in \mathrm{Log}$, we can thus construct a chain of surjections

$$\begin{aligned} r^{(t+1)\lceil k/d\rceil} e^k \sum_{i=0}^{k} \binom{k}{i} 2^{(nt^2-1)d\lceil i/d\rceil} \left(\tfrac{1}{20}2^{nt^2}\right)^{k-i} \\ &= e^k \sum_{i=0}^{k} \binom{k}{i} \left(r^{t+1}2^{(nt^2-1)d}\right)^{\lceil i/d\rceil} \left(r^{t+1}\right)^{\lceil k/d\rceil - \lceil i/d\rceil} \left(\tfrac{1}{20}2^{nt^2}\right)^{k-i} \\ &\twoheadrightarrow e^k \dot{\bigcup_{i\leq k}} \binom{k}{i} r^{(t+1)\lceil k/d\rceil} B^i \left(\tfrac{1}{20}2^{nt^2}\right)^{k-i} \\ &\simeq r^{(t+1)\lceil k/d\rceil} \dot{\bigcup_{i\leq k}} \binom{k}{i} (eB)^i \left(\tfrac{1}{20}e2^{nt^2}\right)^{k-i} \\ &\twoheadrightarrow r^{(t+1)\lceil k/d\rceil} \left(e\left(B \mathbin{\dot\cup} \tfrac{1}{20}2^{nt^2}\right)\right)^k \twoheadrightarrow r^{(t+1)\lceil k/d\rceil} e^k b^k, \end{aligned}$$

where the surjection from the second to the third line is constructed using $h\colon r^{t+1}2^{(nt^2-1)d} \twoheadrightarrow r^{t+1}B^d$, and the last but one surjection follows from Lemma 3.3. We have

$$\begin{aligned} \sum_{i=0}^{k} \binom{k}{i} 2^{(nt^2-1)d\lceil i/d\rceil} \left(\tfrac{1}{20}2^{nt^2}\right)^{k-i} &\leq 2^{nt^2(k+d)} \sum_{i=0}^{k} \binom{k}{i} 2^{-i} 20^{-(k-i)} \\ &= 2^{nt^2(k+d)}(11/20)^k = \left(\tfrac{11}{20}2^{nt^2}\right)^k 2^{nt^2 d} \leq \tfrac{1}{2}\left(\tfrac{12}{20}2^{nt^2}\right)^k \end{aligned}$$

as long as $k \geq 8(nt^2d + 1)$, hence $b \leq (12/20)2^{nt^2}$ by $sWPHP(PV_2(\alpha))$. As $B \preceq_{1/20} b$, we have $B \preceq_0 b + (1/20)2^{nt^2} < (2/3)2^{nt^2}$. □

Corollary 3.5 *(in $T^1_2(\alpha)+sWPHP(PV_2(\alpha))$) Let $d, r > 0$, $d, \varepsilon^{-1} \in \mathrm{Log}$, and $f\colon rs^d \twoheadrightarrow r \times X^d$, where X is $\Sigma^b_1(\alpha)$, and f is $PV_2(\alpha)$. Then $X \precsim s$, and moreover,*

$$\Pr_{\vec{A}}(\vec{A} \textit{ does not isolate } X^c) \preceq_0 2/3,$$

where $c = 12|s|\lceil\varepsilon^{-1}\rceil^2$, $t = |s^c| + 1$, $X \subseteq 2^n$, and $\vec{A} = \langle A_i : i < t\rangle$ is a sequence of matrices $A_i \in 2^{t\times n}$ as in Definition 3.1. □

Remark 3.6 If we assume that f has a $PV_2(\alpha)$-coretraction, the existence of $\vec{A}$ in Theorem 3.4 and Corollary 3.5 is provable even in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$, as the statement becomes $\forall\Sigma^b_2(\alpha)$. This is quite typical behaviour. To avoid unnecessary cluttering of the text, we will only indicate provability in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$ below if it applies to an unmodified statement of a theorem, or if it does not directly follow from Theorem 2.1.

Lemma 3.7 *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) Let $s, \varepsilon^{-1}, c \in \mathrm{Log}$, $c > 0$, $X \in \Sigma^b_1(\alpha)$. If there exists a $PV_2(\alpha)$-surjection $f\colon \lfloor(s+1-\varepsilon)^c\rfloor \twoheadrightarrow X^c$, then there exists a surjection $s \twoheadrightarrow X$ (encoded by a sequence, hence PV-definable).*

Proof: Let $k \le s + 1$ be maximal such that there exists a sequence of length k of pairwise distinct elements of X (by $\Sigma^b_1(\alpha)$-*LMAX* $\subseteq T^1_2(\alpha)$). If $k \le s$, we have $s \twoheadrightarrow X$. Otherwise $X \twoheadrightarrow s + 1$, which implies

$$(s+1)^c\left(1 - \frac{\varepsilon}{s+1}\right) \ge (s+1-\varepsilon)^c \twoheadrightarrow X^c \twoheadrightarrow (s+1)^c,$$

contradicting *sWPHP*. □

Theorem 3.8 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If X is $\Sigma^b_1(\alpha)$, and $X \precsim_\varepsilon s$, there exists a $PV_2(\alpha)$-function f such that $f\colon \lfloor s(1+\varepsilon)\rfloor^c \twoheadrightarrow X^c$ (with a $PV_2(\alpha)$-coretraction), where $0 < c \le 12|s|\lceil\varepsilon^{-1}\rceil^2$.*

Proof: W.l.o.g. $s = S$ in the notation of Definition 3.1. The case $s \le 1$ is left to the reader, we assume $s \ge 2$. Let $a = 4|s|\lceil\varepsilon^{-1}\rceil$, $c = 3\lceil\varepsilon^{-1}\rceil a$, and fix $x_0 \in X^c$, and $\vec{A}\colon X^c \looparrowright 2^t$, where $t = |s^c| + 1$. We define a mapping $f\colon t \times 2^t \to X^c$ by

$$f(i,u) = \begin{cases} x & \text{if } x \in X^c,\ A_i x = u \text{ and } A_i \text{ separates } x \text{ from } X^c, \\ x_0 & \text{otherwise.}\end{cases}$$

The definition of $\looparrowright$ ensures that f is onto; it has a $PV_2(\alpha)$-coretraction defined by

$$g(x) = \langle i, A_i x\rangle, \qquad i = \min\{i < t : A_i \text{ separates } x \text{ from } X^c\}.$$

The function f is itself $PV_2(\alpha)$, as it is computable by the following algorithm: if

$$\neg\exists x \in X^c\, A_i x = u \vee \exists x, x' \in X^c\,(A_i x = A_i x' = u \wedge x \ne x')$$

(these are $\Sigma^b_1(\alpha)$ oracle calls), output x_0. Otherwise there exists a unique x satisfying the $\Sigma^b_1(\alpha)$-condition $x \in X^c \wedge A_i x = u$, and we can find it by binary search.

As $a \geq 4|s| \geq 8$, we have $2^a \geq 3a^2 + 4$. Moreover $(1 + \varepsilon/3)^{\lceil 3/\varepsilon \rceil} \geq 2$, hence

$$\begin{aligned} t2^t &\leq 4s^c(|s^c| + 1) \leq 4s^c(c|s| + 1) = s^c\big(12|s|\lceil\varepsilon^{-1}\rceil a + 4\big) \\ &\leq s^c(3a^2 + 4) \leq s^c 2^a \leq s^c(1 + \varepsilon/3)^{\lceil 3\varepsilon^{-1}\rceil a} \leq (s(1 + \varepsilon/3))^c. \end{aligned}$$

If $s \geq 3/(2\varepsilon)$, we obtain

$$(s(1 + \varepsilon/3))^c \leq (s(1 + \varepsilon) - 1)^c \leq \lfloor s(1 + \varepsilon)\rfloor^c.$$

If $s \leq 3/(2\varepsilon)$, we have $s(1 + \varepsilon/3) \leq s + 1/2$, and $s \in \mathrm{Log}$, hence $s \twoheadrightarrow X$ by Lemma 3.7 (which works in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$, as our surjection has a $PV_2(\alpha)$-coretraction). □

The proof of Theorem 3.8 actually shows the following:

Corollary 3.9 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If $X \in \Sigma^b_1(\alpha)$ and $X \looparrowright 2^t$, there exists a $PV_2(\alpha)$-surjection $f\colon t2^t \twoheadrightarrow X$ with a $PV_2(\alpha)$-coretraction.* □

The corollary below states the important principle that approximate counting with small error reduces to exact counting whenever the latter is possible.

Corollary 3.10 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) Let $X \in \Sigma^b_1(\alpha)$, and $s \leq \varepsilon^{-1} \in \mathrm{Log}$. We have $X \precsim_\varepsilon s$ iff there exists a sequence of length at most s which includes all elements of X.*

Proof: If w is such a sequence, then $w\colon s \twoheadrightarrow X$, hence $X \precsim s$ by Corollary 3.5. (We can use *sWPHP* by Theorem 2.1.) On the other hand, $X \precsim_\varepsilon s$ implies the existence of w by the proof of Lemma 3.7 and Theorem 3.8. □

The following corollary serves several purposes. First, it shows the basic counting principle that upper bounds on cardinality are preserved by surjections. Second, it shows that the present approximate counting generalizes the method of Chapter I in the following sense: if $Y \subseteq 2^n$ and $Y \preceq_\varepsilon s$, then $Y \precsim s + 2\varepsilon 2^n$. Finally, we will often use the special case when f is the identity function to reduce the error of approximation in favor of worse bounds: $X \precsim_\varepsilon s$ implies $X \precsim_\delta \lfloor s(1 + \varepsilon)\rfloor$ for every $\delta^{-1} \in \mathrm{Log}$.

Corollary 3.11 *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) Let $X, Y \in \Sigma^b_1(\alpha)$, $f \in PV_2(\alpha)$, $d, \varepsilon^{-1} \in \mathrm{Log}$, $d, r > 0$. If $X \precsim_\varepsilon s$, and $f\colon r \times X^d \twoheadrightarrow r \times Y^d$, then $Y \precsim \lfloor s(1 + \varepsilon)\rfloor$.*

Proof: We have $\lfloor s(1+\varepsilon)\rfloor^c \twoheadrightarrow X^c$ for some c by Theorem 3.8, hence $r^c\lfloor s(1+\varepsilon)\rfloor^{cd} \twoheadrightarrow r^c X^{cd} \twoheadrightarrow r^c Y^{cd}$, thus $Y \precsim \lfloor s(1 + \varepsilon)\rfloor$ by Corollary 3.5. □

The next two results state fundamental counting principles for computing upper bounds on the size of Cartesian products and unions.

Corollary 3.12 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If $X, Y \in \Sigma^b_1(\alpha)$, $X \precsim_\varepsilon s$, and $Y \precsim_\varepsilon t$, then $X \times Y \precsim \lfloor st(1 + \varepsilon)^2\rfloor$.*

Proof: Use Corollary 3.5, and Theorems 3.8 and 2.1. □

Theorem 3.13 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If $X, Y \in \Sigma^b_1(\alpha)$, $X \precsim_\varepsilon s$, and $Y \precsim_\varepsilon t$, then $X \cup Y \precsim \lfloor (s+t)(1+2\varepsilon)\rfloor$.*

Proof: We can use $sWPHP$ by Theorem 2.1. Take $PV_2(\alpha)$-functions $f\colon S^c \twoheadrightarrow X^c$, and $g\colon T^d \twoheadrightarrow Y^d$ by Theorem 3.8, where $S = \lfloor s(1+\varepsilon)\rfloor$, $T = \lfloor t(1+\varepsilon)\rfloor$. Put $k = 3(c|s| + d|t|)\lceil \varepsilon^{-1}\rceil$. Using Lemma 3.3, we can construct smooth $PV_2(\alpha)$-surjections

$$(X\cup Y)^k \twoheadleftarrow \dot{\bigcup_{i\le k}} \binom{k}{i} X^i Y^{k-i} \twoheadleftarrow \sum_{i\le k}\binom{k}{i} S^{c\lceil i/c\rceil} T^{d\lceil (k-i)/d\rceil} \le S^c T^d \sum_{i\le k}\binom{k}{i} S^i T^{k-i}$$
$$= S^c T^d (S+T)^k \le 2^{c|S|+d|T|}(S+T)^k \le \bigl((1+\varepsilon/3)(S+T)\bigr)^k$$

as $(1+\varepsilon/3)^{\lceil 3\varepsilon^{-1}\rceil} \ge 2$. If

$$(1+\varepsilon/3)(1+\varepsilon)(s+t) \le (s+t)(1+2\varepsilon) - 1 \le \lfloor (s+t)(1+2\varepsilon)\rfloor,$$

we are done by Corollary 3.5. Otherwise $s, t \in \mathrm{Log}$, in which case $\lfloor (s+t)(1+2\varepsilon)\rfloor \twoheadrightarrow X\cup Y$ by Lemma 3.7. □

Theorem 3.13 is one of the most important elementary counting principles. Its dual, which says that the size of a disjoint union $X \mathbin{\dot\cup} Y$ is (approximately) bounded below by the sum of the sizes of X and Y (it has to be formulated contrapositively, see Theorem 3.17), is just as fundamental, but it is considerably harder to prove in our setting. To see why, consider the case where $X \mathbin{\dot\cup} Y = [0, a)$: the obvious fact that $X \cup Y \precsim_\varepsilon a$ does not give us any useful information, hence we must be ready to produce out of thin air a function witnessing that the size of X or Y is (approximately) at most $a/2$. Theorem 2.3 comes to our rescue, as production of magic surjections is exactly what it is good for.

But first we state another consequence of Chapter I. It allows us to reduce the complexity of $\precsim_\varepsilon$ from $\Sigma^b_2(\alpha)$ to $\Pi^b_1(\alpha)$ in many situations, which is indispensable in proofs by induction (notice that our favourite theory has induction only for $\mathcal{B}(\Sigma^b_1(\alpha))$-formulas). We recall that this does not imply any fancy derandomization of AM, as $\Pi^b_1(\alpha)$ here has the meaning of coNP/poly, not coNP (see Section 2).

Recall Definition 3.1.

Lemma 3.14 *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) Let $c \in \mathrm{Log}$, $X \subseteq a \times 2^n$, $X \in \Sigma^b_1(\alpha)$, and put $X_x = \{y < 2^n : \langle x, y\rangle \in X\}$ for each $x < a$. There exists a $\Pi^b_1(\alpha)$-predicate C such that*

$$C(x,t) \Rightarrow X_x \looparrowright 2^t,$$
$$\neg C(x,t) \Rightarrow \Pr_{A_0,\dots,A_{t-1}\in 2^{t\times n}}\bigl(\vec A \textit{ isolates } X_x\bigr) \preceq_0 1/4$$

for every $x < a$, $t < c$.

Proof: The promise problem $L = \langle L^+, L^-\rangle$, where

$$\langle x,t\rangle \in L^+ \Leftrightarrow \Pr_{\vec A}\bigl(\vec A \text{ isolates } X_x\bigr) \preceq_0 1/8,$$
$$\langle x,t\rangle \in L^- \Leftrightarrow \Pr_{\vec A}\bigl(\vec A \text{ isolates } X_x\bigr) \succeq_0 1/4,$$

is a definable $\mathrm{prAM}(\alpha)$-problem, hence the existence of C follows from Theorem 2.4. □

Lemma 3.15 *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) Let $X \in \Sigma^b_1(\alpha)$, $X \neq \varnothing$. There exists a $t \in \mathrm{Log}$ such that $X \looparrowright 2^{t+2}$, and for every $\varepsilon^{-1} \in \mathrm{Log}$ there exists a positive $r \in \mathrm{Log}$, and a smooth $PV_2(\alpha)$-surjection $f\colon r \times (X \mathbin{\dot\cup} \varepsilon 2^t) \twoheadrightarrow r2^t$ with a $PV_2(\alpha)$-coretraction.*

Proof: Assume $X \subseteq 2^n$. Let C be as in Lemma 3.14, find the minimal $k \leq n$ such that $C(k) = 1$ by $PV_2(\alpha)$-induction, and put $t = k - 2$. We have $X \looparrowright 2^k$. Take $g\colon k2^k \twoheadrightarrow X$ and its coretraction $h\colon X \to k2^k$ from Corollary 3.9, and define

$$A = \{u < k2^k : h(g(u)) = u\} = \mathrm{rng}(h).$$

Let $\eta = \varepsilon/4n$, and $A \approx_\eta a$ by Theorem 2.3. We have $g\colon A \simeq X$ with inverse $h\colon X \simeq A$, and there exists a $PV_2(\alpha)$-surjection $r(a + \eta k2^k) \twoheadrightarrow rA$ with a $PV_2(\alpha)$-coretraction for some $r > 0$, $r \in \mathrm{Log}$, hence

$$r(a + \eta k2^k) \twoheadrightarrow r \times X.$$

By minimality of k, and Theorem 3.4 we have

$$a + \eta k2^k \geq 2^t.$$

There exists a $PV_2(\alpha)$-surjection $r(A \mathbin{\dot\cup} \eta k2^k) \twoheadrightarrow ra$ with a $PV_2(\alpha)$-coretraction by Theorem 2.3. We compose it with h to obtain $r(X \mathbin{\dot\cup} \eta k2^k) \twoheadrightarrow ra$, hence

$$r(X \mathbin{\dot\cup} \varepsilon 2^t) \supseteq r(X \mathbin{\dot\cup} 2\eta k2^k) \twoheadrightarrow r(a + \eta k2^k) \geq r2^t. \qquad \square$$

Theorem 3.16 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If $X, Y \in \Sigma^b_1(\alpha)$, and $X \times Y \precsim_\varepsilon st$, then $X \precsim \lfloor s(1+\varepsilon)\rfloor$, or $Y \precsim \lfloor t(1+\varepsilon)\rfloor$.*

Proof: We can use $sWPHP$ by Theorem 2.1. Assume that $X \neq \varnothing \neq Y$, and take a $PV_2(\alpha)$-function $f\colon \lfloor st(1+\varepsilon)\rfloor^d \twoheadrightarrow (X \times Y)^d$ by Theorem 3.8. Let $0 < c \in \mathrm{Log}$, and take $k, \ell, r \in \mathrm{Log}$ such that $X^{cd} \looparrowright 2^{k+2}$, $r(X^{cd} \mathbin{\dot\cup} \eta 2^k) \twoheadrightarrow r2^k$, $Y^{cd} \looparrowright 2^{\ell+2}$, $r(Y^{cd} \mathbin{\dot\cup} \eta 2^\ell) \twoheadrightarrow r2^\ell$ by Lemma 3.15, where $\eta = (8(k+\ell+4))^{-1}$. By Corollary 3.9, there are smooth $PV_2(\alpha)$-surjections $(k+2)2^{k+2} \twoheadrightarrow X^{cd}$, $(\ell+2)2^{\ell+2} \twoheadrightarrow Y^{cd}$. As

$$\eta(k+2)2^{k+\ell+2} + \eta(\ell+2)2^{k+\ell+2} + \eta^2 2^{k+\ell} \leq \tfrac{5}{8}2^{k+\ell},$$

we can construct smooth $PV_2(\alpha)$-surjections

$$\begin{aligned} r^2(X^{cd} \times Y^{cd} \mathbin{\dot\cup} \tfrac{5}{8}2^{k+\ell}) &\twoheadrightarrow r^2(X^{cd} \times Y^{cd} \mathbin{\dot\cup} \eta 2^k Y^{cd} \mathbin{\dot\cup} \eta 2^\ell X^{cd} \mathbin{\dot\cup} \eta^2 2^{k+\ell}) \\ &\simeq r(X^{cd} \mathbin{\dot\cup} \eta 2^k) \times r(Y^{cd} \mathbin{\dot\cup} \eta 2^\ell) \twoheadrightarrow r^2 2^{k+\ell}. \end{aligned}$$

On the other hand, $f^{(c)}\colon \lfloor st(1+\varepsilon)\rfloor^{cd} \twoheadrightarrow X^{cd} \times Y^{cd}$, hence

$$r^2\big(\lfloor st(1+\varepsilon)\rfloor^{cd} + \tfrac{5}{8}2^{k+\ell}\big) \twoheadrightarrow r^2 2^{k+\ell},$$

which implies

$$(st(1+\varepsilon))^{cd} \geq 2^{k+\ell}(1 - \tfrac{6}{8}) = 2^{k-1}2^{\ell-1}$$

by $sWPHP(PV_2(\alpha))$. Therefore $(s(1+\varepsilon/2))^{cd} \geq 2^{k-1}$ or $(t(1+\varepsilon/2))^{cd} \geq 2^{\ell-1}$, hence

$$8(s(1+\varepsilon/2))^{cd}\big(cd|s(1+\varepsilon/2)|+3\big) \geq 8(s(1+\varepsilon/2))^{cd}|4(s(1+\varepsilon/2))^{cd}| \geq 2^{k+2}(k+2)$$

or

$$8(t(1+\varepsilon/2))^{cd}\big(cd|t(1+\varepsilon/2)|+3\big) \geq 8(t(1+\varepsilon/2))^{cd}|4(t(1+\varepsilon/2))^{cd}| \geq 2^{\ell+2}(\ell+2),$$

which implies

$$8(s(1+\varepsilon/2))^{cd}\big(cd|s(1+\varepsilon/2)|+3\big) \twoheadrightarrow X^{cd} \quad \text{or} \quad 8(t(1+\varepsilon/2))^{cd}\big(cd|t(1+\varepsilon/2)|+3\big) \twoheadrightarrow Y^{cd}.$$

We may fix $c \in \mathrm{Log}$ so that

$$8\big(cd|\max\{s,t\}(1+\varepsilon/2)|+3\big) \leq (1+\varepsilon/4)^{cd},$$

hence there exists a $PV_2(\alpha)$-function

$$\Big(s\left(1+\tfrac{7}{8}\varepsilon\right)\Big)^{cd} \geq \big(s(1+\varepsilon/2)(1+\varepsilon/4)\big)^{cd} \twoheadrightarrow X^{cd}$$

or

$$\Big(t\left(1+\tfrac{7}{8}\varepsilon\right)\Big)^{cd} \geq \big(t(1+\varepsilon/2)(1+\varepsilon/4)\big)^{cd} \twoheadrightarrow Y^{cd}.$$

Then

$$X \precsim \lfloor s(1+\varepsilon)\rfloor \quad \text{or} \quad Y \precsim \lfloor t(1+\varepsilon)\rfloor$$

by Corollary 3.5 and Lemma 3.7. □

Theorem 3.17 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If $X, Y \in \Sigma^b_1(\alpha)$, and $X \,\dot\cup\, Y \precsim_\varepsilon s+t+1$, then $X \precsim \lfloor s(1+2\varepsilon)\rfloor$, or $Y \precsim \lfloor t(1+2\varepsilon)\rfloor$.*

Proof: We can use $sWPHP(PV_2(\alpha))$ and $BB\Sigma^b_3(\alpha)$ by Theorem 2.1. W.l.o.g. assume $s \leq t$. Put $S = s(1+\varepsilon)$, $T = (t+1)(1+\varepsilon)$. We fix a surjection $\lfloor S+T\rfloor^c \twoheadrightarrow (X \,\dot\cup\, Y)^c$ by Theorem 3.8. Let $d \in \mathrm{Log}$ be such that $8cd|6(S+T)| \leq (1+\varepsilon/4)^{cd}$, and put $\eta = 1/4$. For each $i \leq cd$, we fix $k_i \leq cd|S+T|$ such that $X^i \times Y^{cd-i} \looparrowright 2^{k_i+2}$, and $PV_2(\alpha)$-functions $f_i\colon r(X^i \times Y^{cd-i} \,\dot\cup\, \eta 2^{k_i}) \twoheadrightarrow r2^{k_i}$, by Lemma 3.15 and $BB\Sigma^b_3(\alpha)$. Then we can construct surjections

$$r\bigg((S+T)^{cd} + \eta\sum_i \binom{cd}{i} 2^{k_i}\bigg) \twoheadrightarrow r\dot{\bigcup_i}\binom{cd}{i}(X^i \times Y^{cd-i} \,\dot\cup\, \eta 2^{k_i}) \twoheadrightarrow r\sum_i \binom{cd}{i}2^{k_i}$$

using Lemma 3.3. By $sWPHP(PV_2(\alpha))$,

$$(1-\eta/4)r\sum_i\binom{cd}{i}2^{k_i} \leq r\bigg((S+T)^{cd} + \eta\sum_i\binom{cd}{i}2^{k_i}\bigg),$$

hence

$$\sum_i\binom{cd}{i}2^{k_i} \leq (S+T)^{cd}(1+2\eta) = \sum_i\binom{cd}{i}S^iT^{cd-i}(1+2\eta)$$

by $sWPHP(PV_2(\alpha))$, which implies

$$2^{k_i} \le S^i T^{cd-i}(1+2\eta) \le \tfrac{3}{2} S^i T^{cd-i}$$

for some $i \le cd$. It follows that

$$\tfrac{3}{4} S^i T^{cd-i}(1+\varepsilon/4)^{cd} \ge 6 S^i T^{cd-i} |6 S^i T^{cd-i}| \ge (k_i+2) 2^{k_i+2} \twoheadrightarrow X^i \times Y^{cd-i}$$

using Corollary 3.9, hence $X^i \times Y^{cd-i} \precsim \frac{3}{4} S^i T^{cd-i}(1+\varepsilon/4)^{cd}$ by Corollary 3.5, which implies

$$X^i \precsim (S(1+\varepsilon/4))^i \quad \text{or} \quad Y^{cd-i} \precsim (T(1+\varepsilon/4))^{cd-i}$$

by Theorem 3.16. We obtain a $PV_2(\alpha)$-function

$$\bigl(s(1+\varepsilon)(1+\varepsilon/3)\bigr)^{ie} \twoheadrightarrow X^{ie} \quad \text{or} \quad \bigl((t+1)(1+\varepsilon)(1+\varepsilon/3)\bigr)^{(cd-i)e} \twoheadrightarrow Y^{(cd-i)e}$$

for some $e \in \mathrm{Log}$ by Theorem 3.8, hence

$$X \precsim \lfloor s(1+2\varepsilon) \rfloor \quad \text{or} \quad Y \precsim \lfloor (t+1)(1+\tfrac{9}{5}\varepsilon) \rfloor$$

by Corollary 3.5 and Lemma 3.7. If

$$(t+1)(1+\tfrac{9}{5}\varepsilon) \le t(1+2\varepsilon),$$

we are done. Otherwise $s \le t < 5\varepsilon^{-1} + 9 \in \mathrm{Log}$, hence the result follows by exact counting, using Corollary 3.10. □

We formulated the key theorems 3.12, 3.13, 3.16, and 3.17 for binary sums and products. It is straightforward to generalize them to sums and products of logarithmically many sets, using simple induction.

Corollary 3.18 *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) Let $n, \varepsilon^{-1}, \delta^{-1} \in \mathrm{Log}$, and let $\{X_i : i < n\}$ be a $\Sigma^b_1(\alpha)$ parametric family of subsets of some 2^m.*

(i) *If $X_i \precsim_\varepsilon s_i$ for every $i < n$, then $\bigcup_{i<n} X_i \precsim \bigl\lfloor (1+2\varepsilon) \sum_{i<n} s_i \bigr\rfloor$.*

(ii) *If $X_i \precsim_\varepsilon s_i$ for every $i < n$, then $\prod_{i<n} X_i \precsim \bigl\lfloor (1+\varepsilon)^{n+1} \prod_{i<n} s_i \bigr\rfloor$.*

(iii) *If $\dot{\bigcup}_{i<n} X_i \succsim_\varepsilon \sum_{i<n} s_i - 1$, there exists $i < n$ such that $X_i \succsim \lfloor s_i(1+2\varepsilon) \rfloor - 1$.*

(iv) *If $n > 0$, and $\prod_{i<n} X_i \succsim_\varepsilon \prod_{i<n} s_i$, there exists $i < n$ such that $X_i \succsim \lfloor s_i(1+\varepsilon) \rfloor$.*

Proof: (i): Let $X = \bigcup_i X_i$, and $s = \sum_i s_i$. We have $X_i \precsim_\eta \lfloor s_i(1+\varepsilon) \rfloor$ by Corollary 3.11, hence

$$X \precsim_\eta \bigl\lfloor s(1+\varepsilon)(1+2\eta)^{n-1} \bigr\rfloor$$

by induction on n from Theorem 3.13 (we can make the induction hypothesis $\Pi^b_1(\alpha)$ using Lemma 3.14). We choose $\eta = \varepsilon/(12n)$ so that $(1+2\eta)^n \le (1+\varepsilon/3)$. We have

$$\bigl(s(1+\varepsilon)(1+\varepsilon/3)\bigr)^c \twoheadrightarrow X^c$$

by Theorem 3.8, hence $X \precsim \lfloor s(1+2\varepsilon) \rfloor$ by Corollary 3.5 and Lemma 3.7.

The other items are proved in a similar way, using Corollary 3.12, and Theorems 3.16 and 3.17. □

We also prove versions of Theorems 3.13 and 3.17 which apply to a "large" number of summands with a uniform description. They can be thought of as averaging arguments: if there are more than st objects in a rectangle of length s, some column must hold at least the average, which is more than $st/s = t$.

Theorem 3.19 *(in $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$) If $X, Y, Z \in \Sigma^b_1(\alpha)$, $Z \subseteq X \times Y$, $X \precsim_\varepsilon s$, and $\{y \in Y : \langle x, y\rangle \in Z\} \precsim_\varepsilon t$ for every $x \in X$, then $Z \precsim \lfloor st(1 + 4\varepsilon)\rfloor$.*

Proof: Assume $X \subseteq 2^n$, and fix $\eta^{-1} \in \mathrm{Log}$ such that $(1 + \eta)^{6n+2} \le 1 + \varepsilon/4$. We denote

$$Z_{a..b} = \{\langle x, y\rangle \in Z : a \le x < b\}$$

for every $a < b \le 2^n$, and $Z_a = Z_{a..a+1}$. By Lemma 3.14, Corollary 3.5 and Theorem 3.8, there exist $\Pi^b_1(\alpha)$-predicates $C(u, v, a), D(u, v, a)$ such that

$$\begin{aligned} C(u, v, a) \to \qquad Z_{u..v} &\precsim_\eta a \to C(u, v, \lfloor a(1 + \eta)\rfloor), \\ D(u, v, a) \to X \cap [u, v) &\precsim_\eta a \to D(u, v, \lfloor a(1 + \eta)\rfloor). \end{aligned}$$

We prove

$$(1) \qquad \forall u < v \le 2^n\, \forall a \le 2^n \left(v - u = 2^k \wedge D(u, v, a) \to C\bigl(u, v, \lfloor at(1 + \varepsilon)(1 + \eta)^{6k+1}\rfloor\bigr)\right)$$

by induction on $k \le n$. The case $k = 0$ is clear, let thus $k > 0$. Assume $D(u, v, a)$. Put $w = (u+v)/2$, and find $b, c \le 2^n$ such that $D(u, w, b)$, $\neg D(u, w, b-1)$, $D(w, v, c)$, $\neg D(w, v, c-1)$ using induction (where "$D(\dots, -1)$" counts as false). By (1) for $k - 1$, we have

$$\begin{aligned} Z_{u..w} &\precsim_\eta \lfloor bt(1 + \varepsilon)(1 + \eta)^{6k-5}\rfloor, \\ Z_{w..v} &\precsim_\eta \lfloor ct(1 + \varepsilon)(1 + \eta)^{6k-5}\rfloor, \end{aligned}$$

hence

$$Z_{u..v} \precsim_\eta \lfloor (b + c)t(1 + \varepsilon)(1 + \eta)^{6k-3}\rfloor$$

by Theorem 3.13. On the other hand, we have $X \cap [u, w) \not\precsim_\eta \lceil b(1 + \eta)^{-1}\rceil - 1$, $X \cap [w, v) \not\precsim_\eta \lceil c(1 + \eta)^{-1}\rceil - 1$, thus

$$X \cap [u, v) \not\precsim_\eta \left\lceil \frac{b}{(1 + \eta)^3}\right\rceil + \left\lceil \frac{c}{(1 + \eta)^3}\right\rceil - 1$$

by Theorem 3.17. As $X \cap [u, v) \precsim_\eta a$, we obtain $a \ge (b + c)(1 + \eta)^{-3}$, i.e., $b + c \le \lfloor a(1 + \eta)^3\rfloor$. Hence

$$Z_{u..v} \precsim_\eta \lfloor at(1 + \varepsilon)(1 + \eta)^{6k}\rfloor,$$

which implies $C\bigl(u, v, \lfloor a(1 + \varepsilon)(1 + \eta)^{6k+1}\rfloor\bigr)$.

Take $k = n$. We have $D(0, 2^n, \lfloor s(1 + \varepsilon)(1 + \eta)\rfloor)$, hence $C\bigl(0, 2^n, \lfloor st(1 + \varepsilon)^2(1 + \eta)^{6n+2}\rfloor\bigr)$ by (1), which gives

$$Z = Z_{0..2^n} \precsim_\eta \lfloor st(1 + \varepsilon)^2(1 + \eta)^{6n+2}\rfloor \le \lfloor st(1 + 4\varepsilon)\rfloor.$$

As it stands, the proof needs $\Pi^b_2(\alpha)$-*LIND*. The theorem is unfortunately not $\forall\Sigma^b_2(\alpha)$, we thus cannot directly use Theorem 2.1. Nevertheless, we can decrease the complexity of the induction as follows. Let $a(u,v)$ be a $PV_2(\alpha)$-function which computes $a \le 2^n$ such that $D(u,v,a) \wedge \neg D(u,v,a-1)$ by binary search. We define a $PV_2(\alpha)$-function $f(k)$ (where $k \le n$ is given in unary) by

$$f(0) = 0,$$

$$f(k+1) = \begin{cases} f(k) + 2^{n-k-1}, & C\big(u, u+2^{n-k-1}, \lfloor a(u,u+2^{n-k-1})t(1+\varepsilon)(1+\eta)^{6(n-k)-5}\rfloor\big), \\ f(k) & \text{otherwise.} \end{cases}$$

If we assume for contradiction $Z \not\precsim_\eta \lfloor st(1+\varepsilon)^2(1+\eta)^{6n+2}\rfloor$, we can prove

$$\neg C\big(f(k), f(k)+2^{n-k}, \lfloor a(f(k), f(k)+2^{n-k})t(1+\varepsilon)(1+\eta)^{6(n-k)+1}\rfloor\big)$$

by $PV_2(\alpha)$-*LIND* on $k \le n$, using the same argument as above. Taking $k = n$, we have $a(f(k), f(k)+1) \le 1$, and we obtain a contradiction with the assumptions. □

Theorem 3.20 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) If $X, Y, Z \in \Sigma^b_1(\alpha)$, $Z \subseteq X \times Y$, and $Z \precsim_\varepsilon st$, then $X \precsim s-1$, or there exists $x \in X$ such that $\{y \in Y : \langle x,y\rangle \in Z\} \precsim \lfloor t(1+2\varepsilon)\rfloor$.*

Proof: Let $n \in \mathrm{Log}$ be such that $X \subseteq 2^n$. Fix $\eta \in \mathrm{Log}$ such that $(1+\eta)^{6n} \le (1+\varepsilon/2)$, and assume $X \not\precsim_\eta s-1$. By induction on $k \le n$, we will show

$$(2) \qquad \exists u < v \le 2^n\, \exists a \le 2^n \Big(v-u \le 2^{n-k} \wedge a \ne 0 \wedge (X \cap [u,v)) \not\precsim_\eta a-1 \wedge Z_{u..v} \precsim_\eta \lfloor at(1+\varepsilon)(1+\eta)^{6k}\rfloor\Big),$$

where $Z_{u..v}$ is as in the proof of Theorem 3.19. If $k=0$, we may take $u=0$, $v=2^n$, $a=s$. Assume that (2) holds for $k<n$. Put $w = \lceil (u+v)/2\rceil$, and find b,c such that $X\cap[u,w) \not\precsim_\eta b-1$, $X\cap[u,w) \precsim_\eta \lfloor b(1+\eta)\rfloor$, $X\cap[w,v) \not\precsim_\eta c-1$, $X\cap[w,v) \precsim_\eta \lfloor c(1+\eta)\rfloor$. Assume $b \ne 0 \ne c$, the other cases are easy. We have $X \cap [u,v) \precsim_\eta \lfloor (b+c)(1+\eta)^3\rfloor$ by Theorem 3.13, hence $a \le \lfloor (b+c)(1+\eta)^3\rfloor$, which implies $Z_{u..v} \precsim_\eta \lfloor bt(1+\varepsilon)(1+\eta)^{6k+4}\rfloor + \lfloor ct(1+\varepsilon)(1+\eta)^{6k+4}\rfloor + 1$. By Theorem 3.17, we obtain $Z_{u..w} \precsim_\eta \lfloor bt(1+\eta)^{6(k+1)}\rfloor$ or $Z_{w..v} \precsim_\eta \lfloor ct(1+\eta)^{6(k+1)}\rfloor$, which gives (2) for $k+1$.

Take u,v,a which witness (2) for $k=n$. Then $v-u \le 1$, and $X\cap[u,v) \ne \varnothing$, hence $v=u+1$, $u \in X$, $a=1$, and $Z_u \precsim_\eta \lfloor t(1+\varepsilon)(1+\eta)^{6n}\rfloor$, which implies $Z_u \precsim \lfloor t(1+2\varepsilon)\rfloor$.

As in the proof of Theorem 3.19, we can replace $\precsim_\eta$ by a $\Pi^b_1(\alpha)$-formula in (2), thus the argument formalizes in $S^2_2(\alpha) + sWPHP(PV_2(\alpha))$. The result is $\forall\Sigma^b_2(\alpha)$, hence it is provable in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$ by Theorem 2.1. (We can also eliminate the instance of $\Sigma^b_2(\alpha)$-*LIND* explicitly as in Theorem 3.19.) □

In the special case $Z = X \times Y$, Theorem 3.20 implies a variant of Theorem 3.16 with slightly different parameters, which may be favourable in some applications (e.g., see the proof of Theorem 4.3): if $X \times Y \precsim_\varepsilon st$, then $X \precsim s-1$ or $Y \precsim \lfloor t(1+2\varepsilon)\rfloor$.

The next theorem shows that we can construct almost counting functions for any set X. Moreover, the conditions imposed on f and g make them very well-behaved: the "local defects" by which the functions differ from true counting functions (i.e., monotone bijections) are small, and evenly distributed across the domain. A possible use of the theorem is that we can apply various results of Chapter I to relatively dense subsets of a sparse set X, as we can lift the whole situation to an interval. (Lifting a $\Sigma^b_1(\alpha)$-set by a $PV_2(\alpha)$-function increases its complexity to $\Delta^b_2(\alpha)$, which is fine as $T^1_2(\alpha) + sWPHP(PV_2(\alpha))$ can count $\Delta^b_2(\alpha)$-sets in the framework of Chapter I.)

The main idea of the construction was suggested by Neil Thapen.

Theorem 3.21 *(in $T^1_2(\alpha) + rWPHP(PV_2(\alpha))$) Let $X \in \Sigma^b_1(\alpha)$, and $\varepsilon^{-1} \in \mathrm{Log}$. There exist numbers t, s such that $s \le t \le \lfloor s(1+\varepsilon)\rfloor$, and non-decreasing $PV_2(\alpha)$-functions*

$$\begin{tikzcd} t \arrow[r, shift left, "f"] & X \arrow[l, shift left, "f'"] \arrow[r, shift left, "g"] & s \arrow[l, shift left, "g'"] \end{tikzcd}$$

such that $f \circ f' = \mathrm{id}_X$, $g \circ g' = \mathrm{id}_s$ (hence f, g are onto, and f', g' are injective), f, g are ≤ 2-to-1, and

$$\left\lfloor \frac{s}{t} u \right\rfloor \le g(f(u)) \le \left\lceil \frac{s}{t} u \right\rceil$$

for every $u < t$.

Proof: Fix $n \in \mathrm{Log}$ such that $X \subseteq 2^n$, and $\eta^{-1} \in \mathrm{Log}$ such that $(1+\eta)^{8n} \le 1+\varepsilon$. Let C be a $\Pi^b_1(\alpha)$-predicate such that

$$C(u,v,w) \to X \cap [u,v) \precsim_\eta w \to C(u,v,\lfloor w(1+\eta)\rfloor)$$

for all $u, v, w \le 2^n$. Using binary search, we can define a $PV_2(\alpha)$-function S such that

$$C(u,v,S(u,v)) \wedge \neg C(u,v,S(u,v)-1)$$

for all $u, v \le 2^n$. Put $a = S(0, 2^n)$. If $a < \eta^{-1}$, then $a \in \mathrm{Log}$, hence the required functions exist by Corollary 3.10. We thus assume $a \ge \eta^{-1}$. Consider the following algorithm (where u, u_k, v_k, w_k are rationals):

```
**input:** either $u \in [0,1)$, or $x \in X$
$x_0 \leftarrow 0$, $y_0 \leftarrow 2^n$, $u_0 \leftarrow 0$, $v_0 \leftarrow 1$, $r_0 \leftarrow a$
**for** $k = 0, \dots, n-1$ **do:**
    $z_k \leftarrow (x_k + y_k)/2$, $p_k \leftarrow S(x_k, z_k)$, $q_k \leftarrow S(z_k, y_k)$
    $w_k \leftarrow (q_k u_k + p_k v_k)/(p_k + q_k)$
    **if** $u < w_k$ or $x < z_k$ **then** $\langle x_{k+1}, y_{k+1}, u_{k+1}, v_{k+1}, r_{k+1}\rangle \leftarrow \langle x_k, z_k, u_k, w_k, p_k\rangle$
        **else** $\langle x_{k+1}, y_{k+1}, u_{k+1}, v_{k+1}, r_{k+1}\rangle \leftarrow \langle z_k, y_k, w_k, v_k, q_k\rangle$
```

If it is necessary to indicate the input, we will write $p_k(u)$ for the value of p_k assigned by the algorithm on input u, and so on.

Claim 3.21.1 *Let $u \le u' < 1$, $x \le x' \in X$, and $\ell \le k \le n$.*

(i) $y_k = x_k + 2^{n-k}$.

(ii) $u_k(u) \le u < v_k(u)$, $x_k(x) \le x < y_k(x)$.

(iii) $r_k = S(x_k, y_k) \ne 0$, *and* $p_k + q_k \ne 0$ (*hence the division step makes sense*).

(iv) $x_\ell \le x_k$, $y_\ell \ge y_k$, $u_\ell \le u_k$, $v_\ell \ge v_k$.

(v) *Either* $\langle x_k(u), y_k(u), u_k(u), v_k(u)\rangle = \langle x_k(u'), y_k(u'), u_k(u'), v_k(u')\rangle$, *or* $y_k(u) \le x_k(u')$, $v_k(u) \le u_k(u')$.

(vi) *Either* $\langle x_k(x), y_k(x), u_k(x), v_k(x)\rangle = \langle x_k(x'), y_k(x'), u_k(x'), v_k(x')\rangle$, *or* $y_k(x) \le x_k(x')$, $v_k(x) \le u_k(x')$.

(vii) *If* $x_k(u) \le x < y_k(u)$ *or* $u_k(x) \le u < v_k(x)$, *then*

$$\langle x_k(u), y_k(u), u_k(u), v_k(u)\rangle = \langle x_k(x), y_k(x), u_k(x), v_k(x)\rangle.$$

(viii) $(1+\eta)^{-3} r_k \le p_k + q_k \le (1+\eta)^3 r_k$.

(ix) $(1+\eta)^{-3k} r_k \le a(v_k - u_k) \le (1+\eta)^{3k} r_k$.

Proof: (i)–(vii): Straightforward induction on k.

(viii): As $p_k = S(x_k, z_k)$, we have $\neg C(x_k, z_k, p_k - 1)$, thus $X \cap [x_k, z_k) \not\gtrsim_\eta \lceil p_k(1+\eta)^{-1}\rceil - 1$. Similarly $X \cap [z_k, y_k) \not\gtrsim_\eta \lceil q_k(1+\eta)^{-1}\rceil - 1$, hence

$$\begin{aligned} X \cap [x_k, y_k) &\not\gtrsim_\eta \left\lceil \lceil p_k(1+\eta)^{-1}\rceil (1+\eta)^{-2}\right\rceil + \left\lceil \lceil q_k(1+\eta)^{-1}\rceil (1+\eta)^{-2}\right\rceil - 1 \\ &\ge \lceil (p_k+q_k)(1+\eta)^{-3}\rceil - 1 \end{aligned}$$

by Theorem 3.17. On the other hand, $r_k = S(x_k, y_k)$, hence $C(x_k, y_k, r_k)$, and $X \cap [x_k, y_k) \gtrsim_\eta r_k$. This implies $r_k \ge \lceil (p_k + q_k)(1+\eta)^{-3}\rceil$, hence $r_k(1+\eta)^3 \ge p_k + q_k$. In a similar way we have $X \cap [x_k, z_k) \lesssim_\eta p_k$, $X \cap [z_k, y_k) \lesssim_\eta q_k$, and $X \cap [x_k, y_k) \not\gtrsim_\eta \lceil r_k(1+\eta)^{-1}\rceil - 1$, hence $\lceil r_k(1+\eta)^{-1}\rceil \le \lfloor (p_k+q_k)(1+\eta)^2\rfloor$ by Theorem 3.13, thus $(1+\eta)^{-3} r_k \le (p_k + q_k)$.

(ix): By induction on k, using (viii), and the identities

$$w_k - u_k = \frac{p_k}{p_k+q_k}(v_k - u_k), \qquad v_k - w_k = \frac{q_k}{p_k+q_k}(v_k - u_k). \qquad \square \text{ (Claim 3.21.1)}$$

Let $t = \lceil a(1+\eta)^{3n}\rceil$, $s = \lfloor a(1+\eta)^{-3n}\rfloor$, $f(u) = x_n(u/t)$, $f'(x) = \lceil t v_n(x)\rceil - 1$, $g(x) = \lceil s v_n(x)\rceil - 1$, $g'(v) = x_n(v/s)$ for any integers $u < t$, $v < s$, $x \in X$. We have $t \le a(1+\eta)^{4n} \le \lfloor a(1+\eta)^{-3n}\rfloor (1+\eta)^{8n} \le s(1+\varepsilon)$. Notice that $r_n = 1$ (hence $x_n \in X$) by (i) and (iii), thus

$$\frac{1}{2s} \le \frac{1}{t} \le v_n - u_n \le \frac{1}{s} \le \frac{2}{t}$$

by (ix) (in particular, $f'(x), g(x) \ge 0$). Clearly $f\colon t \to X$, $f'\colon X \to t$, $g\colon X \to s$, $g'\colon s \to X$, and all the functions are monotone by (v), (vi).

If $u = f'(x)$, we have $u_n(x) \le v_n(x) - 1/t \le u/t < v_n(x)$, hence $f(u) = x_n(u/t) = x_n(x) = x$ by (vii), thus $f \circ f' = \mathrm{id}$. If $f(u) = x$, then $tu_n(x) = tu_n(u/t) \le u < tv_n(u/t) = tv_n(x)$ by (vii). As there are at most two integers in $[tu_n(x), tv_n(x))$, we have $|f^{-1}(x)| \le 2$.

If $x = g'(v)$, then $sv_n(x) - 1 \le su_n(x) = su_n(v/s) \le v < sv_n(v/s) = sv_n(x)$ by (vii), hence $v = g(x)$, thus $g \circ g' = \mathrm{id}$. Assume that $x, x', x'' \in X$, $x < x' < x''$. We have $y_n(x) = x + 1 \le x_n(x') = x'$ by (i) and (ii), hence $sv_n(x) \le su_n(x') \le sv_n(x') - 1/2$ by (vi). Similarly $sv_n(x') \le sv_n(x'') - 1/2$, hence $g(x'') = \lceil sv_n(x'')\rceil - 1 \ge \lceil sv_n(x)\rceil > g(x)$. It follows that $|g^{-1}(v)| \le 2$ for any $v < s$.

Let $u < t$, and put $x = f(u)$, $v = g(x)$. We have $v_n(x) - 1/s \le u_n(x) = u_n(u/t) \le u/t < v_n(u/t) = v_n(x)$ by (vii) and (ii), and $v_n(x) - 1/s \le v/s < v_n(x)$ by definition, hence $-1/s < u/t - v/s < 1/s$. □

4 Applications

We begin with a classical theorem which cannot be avoided by any self-respecting theory of counting.

Theorem 4.1 (Ramsey theorem) (*in* $T^1_2(G) + rWPHP(PV_2(G))$) *An undirected graph G on N vertices contains a clique or independent set of size at least $|N|/2$.*

Proof: We formalize the following well-known proof. We pick a node a_0, and let c_0 be the majority colour among edges incident with a_0. We continue with nodes connected to a_0 by a c_0-coloured edge, and repeat the process. In this way, we construct a sequence $a_0, \dots, a_{k-1}$ of nodes and a sequence $c_0, \dots, c_{k-1}$ of colours such that the edge from a_i to a_j is c_i-coloured for $i < j$, and there are at least (roughly) $N/2^k$ nodes connected to every a_i by a c_i-coloured edge. We can carry on as long as $k < \log_2 N$, obtaining a prehomogeneous set of size $\log_2 N$, from which we select a homogeneous set of size $\log_2 N/2$ by taking the majority colour among $\vec{c}$. We proceed with the formal details.

We can use *sWPHP* by Theorem 2.1. For every $a \ne b$, we define $C(a,b) < 2$ so that $C(a,b) = 1$ iff there is an edge between a and b in G. Let $\varepsilon^{-1} \in \mathrm{Log}$ be such that $(1-2\varepsilon)^{|N|} \ge 1/2$. By induction on $k \le |N| - 2$, we prove that there exists a sequence $\langle c_i : i < k\rangle$ of $c_i < 2$, and a sequence $\langle a_i : i < k\rangle$ of pairwise distinct $a_i < N$ such that $C(a_i, a_j) = c_i$ whenever $i < j < k$, and

$$S(\vec{a};\vec{c}) := \{x : \forall i < k\, C(a_i, x) = c_i\} \not\precsim_\varepsilon \left\lfloor \frac{N}{2^k}(1-2\varepsilon)^{k+1} \right\rfloor - 1.$$

(We can make the induction hypothesis $\Sigma^b_1(G)$ by Lemma 3.14, as in the proof of Theorem 3.19.) The base step $k = 0$ amounts to $N \not\precsim_\varepsilon \lfloor N(1-2\varepsilon)\rfloor - 1$, which follows from Theorem 3.8 and *rWPHP*. Assume the statement holds for k, we will show it for $k+1$. We have $S(\vec{a};\vec{c}) \ne \varnothing$, we may thus pick any $a_k \in S(\vec{a};\vec{c})$. The set $S(\vec{a};\vec{c})$ can be divided into nodes x such that $C(a_k, x) = 0$, nodes such that $C(a_k, x) = 1$, and node a_k itself, hence

$$S(\vec{a};\vec{c}) = \{a_k\} \cup S(\vec{a}, a_k; \vec{c}, 0) \cup S(\vec{a}, a_k; \vec{c}, 1).$$

We have $1 \precsim_\varepsilon 1$, and

$$\Big\lfloor (1+2\varepsilon)\Big(1+2\big(\lfloor N2^{-(k+1)}(1-2\varepsilon)^{k+2}\rfloor - 1\big)\Big)\Big\rfloor \\ \le \Big\lfloor 2(1+2\varepsilon)N2^{-(k+1)}(1-2\varepsilon)^{k+2} - (1+2\varepsilon)\Big\rfloor \le \lfloor N2^{-k}(1-2\varepsilon)^{k+1} - 1\rfloor,$$

hence

$$S(\vec{a}, a_k; \vec{c}, c_k) \not\precsim_\varepsilon \left\lfloor \frac{N}{2^{k+1}}(1-2\varepsilon)^{k+2} \right\rfloor - 1$$

for some $c_k < 2$ by Theorem 3.13.

Let $\vec{a}$, $\vec{c}$ be the sequences given by the statement above for $k = |N| - 2$. We have $S(\vec{a}; \vec{c}) \not\precsim_\varepsilon 0$, hence there exists $a_k \in S(\vec{a}; \vec{c})$. There exists $c < 2$ such that $|\{i < k : c_i = c\}| \ge \lceil k/2 \rceil$, then $\{a_i : c_i = c\} \cup \{a_k\}$ is a homogeneous set of size $\lceil |N|/2 \rceil$. □

The Ramsey theorem was, of course, proved in bounded arithmetic by Pudlák [152]. The point of Theorem 4.1 is that (apart from a few ε sprinkled here and there) the argument follows almost literally the usual combinatorial proof of the theorem, without the need to resort to *ad hoc* functions for simulation of counting by *WPHP*.

Our first real result will be the tournament principle (originally discovered by Erdős [77]), whose provability in bounded arithmetic was posed as a problem by Krajíček [55, 116]. Recall that a *tournament* is a directed graph G in which there exists exactly one directed edge between any pair of distinct vertices ("players"); if there is an edge going from a to b, we write $a \to b$, and say that a beats b. A *dominating set* is a set D of vertices such that every player outside of D is beaten by some player in D.

Theorem 4.2 (Tournament principle) (*in* $T^1_2(G) + rWPHP(PV_2(G))$) *A tournament* G *with* N *players has a dominating set of size at most* $|N|$.

Proof: Informally, the argument is as follows. There are $N(N-1)/2$ edges in the tournament, hence we may choose a player a_0 who beats at least $(N-1)/2$ other players. We repeat the process with the subtournament consisting of the unbeaten players, halving the size at each step. After at most $|N|$ steps, we reach the empty set, hence we obtaining a dominating set of size $|N|$. We now give the formal proof.

We can work in $S^2_2(G) + sWPHP(PV_2(G))$ by Theorem 2.1. Choose $\varepsilon^{-1} \in \mathrm{Log}$ such that $(1+\varepsilon)^{8(|N|+1)} < 2$. If $\langle a_i : i < k \rangle$ is a sequence of vertices, we denote

$$G(\vec{a}) = \{x < N : \forall i < k\, x \to a_i\}.$$

By $\Sigma^b_2(G)$-*LIND* on $k \le |N| + 1$, we will prove that there exists a sequence $\langle a_i : i < k \rangle$ such that

$$G(\vec{a}) \precsim_\varepsilon \left\lfloor \frac{N}{2^k}(1+\varepsilon)^{8k} \right\rfloor. \tag{3}$$

The case $k = |N| + 1$ then gives $G(\vec{a}) = \varnothing$, i.e., $\vec{a}$ is a dominating set of size $|N| + 1$. (How to get rid of the $+1$ is left as an exercise. Hint: in real world, the bound $|N|$ is not tight.)

The base case $k = 0$ is obvious. Assume that (3) holds for k, we will show it for $k+1$. Find s such that $G(\vec{a}) \precsim_\varepsilon \lfloor s(1+\varepsilon)\rfloor$, $G(\vec{a}) \not\precsim_\varepsilon s-1$. Notice that $s \le N2^{-k}(1+\varepsilon)^{8k}$. We have

$$\{\langle x,y\rangle \in G(\vec{a}) : x \ne y\} \subseteq G(\vec{a})^2 \precsim_\varepsilon \lfloor s^2(1+\varepsilon)^4\rfloor \le 2\left\lfloor \frac{s^2}{2}(1+\varepsilon)^4\right\rfloor + 1$$

by Corollary 3.12, hence

$$\{\langle x,y\rangle \in G(\vec{a})^2 : y \to x\} \precsim_\varepsilon \left\lfloor \frac{s^2}{2}(1+\varepsilon)^6\right\rfloor \quad \text{or} \quad \{\langle x,y\rangle \in G(\vec{a})^2 : x \to y\} \precsim_\varepsilon \left\lfloor \frac{s^2}{2}(1+\varepsilon)^6\right\rfloor$$

by Theorem 3.17, and properties of the tournament. In the former case, there exists an $x \in G(\vec{a})$ such that

$$G(\vec{a},x) = \{y \in G(\vec{a}) : y \to x\} \precsim_\varepsilon \left\lfloor \frac{s}{2}(1+\varepsilon)^8\right\rfloor \le \left\lfloor \frac{N}{2^{k+1}}(1+\varepsilon)^{8(k+1)}\right\rfloor$$

by Theorem 3.20. The latter case is symmetric. □

As proved by E. and G. Szekeres [168], every tournament has a dominating set of size $|N| - ||N|| + O(1)$. We could formalize this stronger result with no additional difficulty; we skip the proof as it involves lengthy quotes from [168] with no particular benefit for our purpose (which is to illustrate the machinery developed in Section 3).

For the sake of completeness, we mention that Erdős [77] proved a lower bound of $|N| - 2||N|| + O(1)$ on the minimal size of a dominating set in random tournaments, and Razborov [155] provided tournaments computable by $AC^0[2]$-circuits with the same property. We do not know how to prove these lower bounds in bounded arithmetic. (Ojakian [136] formalizes Erdős's proof in a different setting, where $N \in \mathrm{Log}$.) An explicit construction of tournaments without small dominating sets was given in [85]: if $p \equiv -1 \pmod 4$ is a prime, the tournament with p players defined by

$$a \to b \iff \left(\frac{a-b}{p}\right) = 1$$

has no dominating set of size $\frac{1}{2}|p| - ||p||$. However, their proof depends on Weil's Riemann hypothesis for curves over finite fields, which we cannot expect to prove in bounded arithmetic by any stretch of imagination.

It turns out that generalizations of the tournament principle are more useful in applications than the principle itself. We provide such a generalization next. The statement seems to be new even outside the context of bounded arithmetic; it was inspired by a variant of the tournament principle introduced in [82] (our Corollary 4.4), and a combinatorial principle implicit in [121] (Corollary 4.5).

In order to explain it, let us consider first Corollary 4.4, which is a symmetric generalization of the tournament principle to arbitrary binary relations that may not be tournaments. We can reformulate it as follows: given a colouring of ordered pairs of points of a by two colours, there is a colour $i < 2$, and a set D of size $\log a$ with the following property: for any point x, there is an i-coloured pair whose ith coordinate is x, and the other coordinate belongs to D. Now we can generalize the statement to higher dimensions as follows (this is the special case of

Theorem 4.3 with $a_i = a$, $p_i = 1/d$, $m_i = 1$): given a colouring of d-tuples of points of a by d colours, there exists a colour $i < d$, and a set D of $(d-1)$-tuples of size $(d-1)\log a$ with the following property: for any point x, there exists an i-coloured d-tuple whose ith coordinate is x, and the tuple consisting of the remaining coordinates belongs to D.

In order to accommodate Corollary 4.5, we introduce as an extra complication the possibility that the colouring is not total. We only require that it is "dense", in the sense that every hypercube with sufficiently large sides (sets of size m) contains a tuple whose colour is defined. The conclusion is modified so that the d-tuple only needs to be i-coloured if its colour is defined, and there will be an exceptional small (of size less than m) set M whose points $x \in M$ are exempt from the existence condition. To guard against trivializing the conclusion, we also require that any tuple from D can be extended to a d-tuple with defined colour ($D \subseteq S_i$ in the notation below). Finally, we allow each coordinate to use a different value of a and m for extra generality, as it does not change the proof, and indeed it simplifies the notation used in the proof in that it allows us to conveniently specify which coordinate in the product a^d are we referring to.

Theorem 4.3 *(in $T^1_2(C)+rWPHP(PV_2(C))$) Let $0 < d \in \mathrm{Log}$. Let $\langle a_i : i < d\rangle$ and $\langle m_i : i < d\rangle$ be sequences of positive integers such that $m_i \in \mathrm{Log}$, $\langle p_i : i < d\rangle$ a sequence of rationals $p_i \in \mathbb{Q}_{\mathrm{Log}}$ such that $0 < p_i < 1$, and C a partial function from $\prod_{i<d} a_i$ to d.*

Assume that $\sum_{i<d} p_i \le 1$, and $\mathrm{dom}(C) \cap \prod_{i<d} M_i \ne \varnothing$ for every sequence $\langle M_i : i < d\rangle$ of subsets $M_i \subseteq a_i$ such that $|M_i| = m_i$. Put

$$S_i = \Big\{\langle x_j : j \ne i\rangle \in \prod_{j\ne i} a_j : \exists x_i \in a_i\, \vec{x} \in \mathrm{dom}(C)\Big\}.$$

Then there exists an $i < d$, a set $D \subseteq S_i$ of size at most

$$2 + \lfloor \log_{(1-p_i)^{-1}}(a_i/m_i)\rfloor \le 1 + (p_i^{-1} - 1)|\lfloor a_i/m_i\rfloor|,$$

and a set $M \subseteq a_i$ of size $|M| < m_i$ with the following property: for every $x_i \in a_i \smallsetminus M$ there exists $\langle x_j : j \ne i\rangle \in D$ such that $C(\vec{x}) = i$ or $\vec{x} \notin \mathrm{dom}(C)$.

Proof: The statement is $\forall\Sigma^b_2(C)$, we can thus work in $S^2_2(C) + sWPHP(PV_2(C))$. We write $C(\vec{x})\!\uparrow$ for $\vec{x} \notin \mathrm{dom}(C)$. If $\vec{x} \in \prod_{j\ne i} a_j$ and $x \in a_i$, and if i is clear from the context, we will write $C(\vec{x}, x)$ instead of $C(x_0, \dots, x_{i-1}, x, x_{i+1}, \dots, x_{d-1})$.

For each $i < d$, put $c_i = 2 + \lfloor \log_{(1-p_i)^{-1}}(a_i/m_i)\rfloor$. As $c_i \in \mathrm{Log}$, and $a_i(1-p_i)^{c_i-1} < m_i$, we can construct $\delta_i \in \mathbb{Q}_{\mathrm{Log}}$ such that $0 < \delta_i < p_i$, and $a_i(1-\delta_i)^{c_i} < m_i$. Then there exists an $0 < \varepsilon \in \mathbb{Q}_{\mathrm{Log}}$ such that $\big(1 - p_i(1+\varepsilon)^{-19}\big)(1+\varepsilon)^3 \le 1 - \delta_i$ for every $i < d$.

By $\Sigma^b_2(C)$-*LMAX*, we can find the maximal k such that there exist sequences $\langle k_i : i < d\rangle$, $\langle \vec{x}^{i,j} : i < d, j < k_i\rangle$ satisfying $k = \sum_i k_i$, $k_i \le c_i$, $\vec{x}^{i,j} \in S_i$, and

$$M_i := \{x < a_i : \forall j < k_i\, \exists \ell \ne i\, C(\vec{x}^{i,j}, x) = \ell\} \precsim_\varepsilon \lfloor a_i(1-\delta_i)^{k_i}\rfloor$$

for every $i < d$. If $|M_i| < m_i$ for some $i < d$, the conclusion of the theorem holds with $M = M_i$, $D = \{\vec{x}^{i,j} : j < k_i\}$. We thus assume $|M_i| \ge m_i$ (which implies $k_i < c_i$ by the choice of

$\delta_i)$ for every $i < d$, and we intend to reach a contradiction. We put $X = \prod_j M_j \cap \operatorname{dom}(C)$, $X_i = \{\vec{x} \in X : C(\vec{x}) = i\}$, $N_i = S_i \cap \prod_{j \neq i} M_j$, and $O_i = (N_i \times M_i) \smallsetminus X$.

The intuition is as follows. For any i and $\vec{x} \in S_i$, we have

$$|\{x \in M_i : \exists \ell \neq i\, C(\vec{x}, x) = \ell\}| \geq a_i(1-\delta_i)^{k_i+1} \geq |M_i|(1-\delta_i)$$

by maximality of k_i, hence $\Pr_{x \in M_i}(C(\vec{x}, x){\uparrow} \vee C(\vec{x}, x) = i) \leq \delta_i$. Consequently,

$$1 = \sum_i \Pr_{\vec{x} \in X}(C(\vec{x}) = i) \leq \sum_i \Pr_{\vec{x} \in N_i \times M_i}(C(\vec{x}){\uparrow} \vee C(\vec{x}) = i) \leq \sum_i \delta_i < 1$$

using nonemptiness of X, which is a contradiction. Now we formalize this argument using approximate counting.

By assumption, $X \neq \varnothing$, hence we can find a $t > 0$ such that $X \precsim_\varepsilon \lfloor t(1+\varepsilon)\rfloor$, $X \not\precsim_\varepsilon t-1$ by $\Sigma^b_2(C)$-*LIND*. Fix $i < d$, and let w_i be such that $M_i \precsim_\varepsilon \lfloor w_i(1+\varepsilon)\rfloor$, and $M_i \not\precsim_\varepsilon w_i - 1$. Consequently, $w_i \leq \lfloor a_i(1-\delta_i)^{k_i}\rfloor$.

Take any $\vec{x} \in S_i$. By Theorem 3.17, we have

$$\{x \in M_i : C(\vec{x}, x){\uparrow} \vee C(\vec{x}, x) = i\} \precsim_\varepsilon \big\lfloor \lfloor w_i p_i (1+\varepsilon)^{-18}\rfloor (1+\varepsilon)^2 \big\rfloor \leq \lfloor w_i p_i (1+\varepsilon)^{-16}\rfloor \tag{4}$$

or

$$\begin{aligned}\{x \in M_i : \exists \ell \neq i\, C(\vec{x}, x) = \ell\} &\precsim_\varepsilon \big\lfloor \big(\lfloor w_i(1+\varepsilon)\rfloor - \lfloor w_i p_i (1+\varepsilon)^{-18}\rfloor - 1\big)(1+\varepsilon)^2 \big\rfloor \\ &\leq \lfloor w_i \big(1 - p_i(1+\varepsilon)^{-19}\big)(1+\varepsilon)^3 \rfloor \leq \lfloor a_i(1-\delta_i)^{k_i+1}\rfloor.\end{aligned}$$

The latter however contradicts the maximality of k_i, hence (4) holds for every $\vec{x} \in S_i$. Find v_i such that $N_i \precsim_\varepsilon \lfloor v_i(1+\varepsilon)\rfloor$, $N_i \not\precsim_\varepsilon v_i - 1$. We have

$$P_i := \{\vec{x} \in N_i \times M_i : C(\vec{x}){\uparrow} \vee C(\vec{x}) = i\} \precsim_\varepsilon \lfloor v_i w_i p_i (1+\varepsilon)^{-11}\rfloor \tag{5}$$

by Theorem 3.19, and (4). Let $O_i \precsim_\varepsilon \lfloor u_i(1+\varepsilon)\rfloor$, $O_i \not\precsim_\varepsilon u_i - 1$. We claim

$$v_i w_i \leq \lfloor (t + u_i)(1+\varepsilon)^6 \rfloor. \tag{6}$$

Note that $N_i \times M_i \subseteq X \cup O_i \precsim_\varepsilon \lfloor (t+u_i)(1+\varepsilon)^3\rfloor$ by Theorem 3.13. Assume first $v_i \geq 2/\varepsilon$. We have $N_i \not\precsim_\varepsilon v_i - 1 \geq \big\lfloor \lfloor v_i(1+\varepsilon)^{-2}\rfloor (1+2\varepsilon)\big\rfloor$, hence

$$\lfloor (t+u_i)(1+\varepsilon)^3 \rfloor > w_i \lfloor v_i (1+\varepsilon)^{-2}\rfloor \geq v_i w_i (1+\varepsilon)^{-3}$$

by Theorem 3.20. The case $w_i \geq 2/\varepsilon$ is symmetric. If $v_i, w_i \leq 2/\varepsilon$, then in particular $v_i, w_i \in \mathrm{Log}$, and we can derive $N_i \times M_i \not\precsim_\varepsilon \lceil v_i w_i (1+\varepsilon)^{-1}\rceil - 1$ easily by exact counting, which implies (6) as above.

The definition of S_i implies $X \subseteq N_i \times M_i$, hence $P_i = O_i \mathbin{\dot\cup} X_i$. We thus obtain

$$O_i \mathbin{\dot\cup} X_i \precsim_\varepsilon \lfloor v_i w_i p_i (1+\varepsilon)^{-11}\rfloor \leq \lfloor (t+u_i) p_i (1+\varepsilon)^{-5}\rfloor \leq \lfloor t p_i (1+\varepsilon)^{-5}\rfloor + \lceil u_i p_i (1+\varepsilon)^{-5}\rceil$$

from (5) and (6), which implies

$$O_i \precsim_\varepsilon \big\lfloor \big(\lceil u_i p_i (1+\varepsilon)^{-5}\rceil - 1\big)(1+\varepsilon)^2 \big\rfloor \leq \lceil u_i (1+\varepsilon)^{-3}\rceil - 1 \quad \text{or} \quad X_i \precsim_\varepsilon \lfloor t p_i (1+\varepsilon)^{-3}\rfloor$$

by Theorem 3.17. The former contradicts the choice of u_i, hence the latter holds for every $i < d$. As $t > 0$, we obtain

$$X = \bigcup_i X_i \precsim_\varepsilon \Big\lfloor (1+\varepsilon)^2 \sum_i tp_i(1+\varepsilon)^{-3} \Big\rfloor \le \lfloor t(1+\varepsilon)^{-1} \rfloor \le t-1$$

from Corollary 3.18, which contradicts the definition of t. □

Corollary 4.4 *(in $T^1_2(R) + rWPHP(PV_2(R))$) Let R be a binary relation on a. There exists a set $D \subseteq a$ of size at most $|a| + 1$ such that*

$$\forall x < a\, \exists y \in D\, R(x,y) \vee \forall y < a\, \exists x \in D\, \neg R(x,y).$$

Proof: Use Theorem 4.3 with $d = 2$, $a_i = a$, $p_i = 1/2$, $m_i = 1$, and C the (total) characteristic function of R. □

Corollary 4.5 *(in $T^1_2(R) + rWPHP(PV_2(R))$) Let $c \in \mathrm{Log}$, and let $a^{[i]}$ denote the set of i-element subsets of a. Assume that $R \subseteq a^{[c]} \times a$ is a relation satisfying*

$$\forall X \in a^{[c+1]}\, \exists x \in X\, R((X \smallsetminus \{x\}), x).$$

Then there exists a set $D \subseteq a^{[c]}$ of size $|D| \le c|a|$, and a set $M \subseteq a$ of size at most c, such that

$$\forall x \in a \smallsetminus \big(M \cup \bigcup D\big)\, \exists X \in D\, R(X,x).$$

Proof: Apply Theorem 4.3 with $d = m_i = c+1$, $a_i = a$, $p_i = 1/d$, and

$$C(x_0,\dots,x_c) = \begin{cases} \min\{i \le c : R(\{x_j : j \ne i\}, x_i)\} & \text{if } x_i \text{ are pairwise distinct,} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Observe that $\vec{x} \in S_i$ iff the elements of $\vec{x}$ are pairwise distinct. □

The collapse of the bounded arithmetic hierarchy implies the collapse of the polynomial-time hierarchy. The original result is by Krajíček et al. [121], who prove that $T^i_2 = S_2$ implies $\Sigma^P_{i+1} \subseteq \Delta^P_{i+1}/\mathrm{poly}$ (hence also $\mathrm{PH} = \Sigma^P_{i+2} = \Pi^P_{i+2}$). Buss [39] formalized a weaker conclusion inside the bounded arithmetic: if $T^i_2 = S^{i+1}_2$, then T^i_2 proves $\Sigma^b_{i+1} \subseteq \Pi^b_{i+1}/\mathrm{poly}$, and $\Sigma^b_\infty = \mathcal{B}(\Sigma^b_{i+2})$ (cf. also Zambella [182]). We will show that the stronger collapse from [121] can be formalized in bounded arithmetic as well. Surprisingly, this also allows us to strengthen the collapse to $\mathrm{PH} = \mathcal{B}(\Sigma^P_{i+1})$, using a result from [67].

Theorem 4.6 *If $T^i_2 = S^{i+1}_2$, then T^i_2 proves $\Sigma^b_{i+1} \subseteq \Delta^b_{i+1}/\mathrm{poly}$.*

Proof: It suffices to formalize in T^i_2 the proofs of [121, Thm. B, L. 2.2], also repeated (with a slightly different notation) in [116, Thm. 10.2.4, L. 10.2.2]. We assume the reader has one of these two proofs at hand, but we sketch an outline of the proof here for convenience. We consider a Σ^b_{i+1}-predicate $\exists w \le v\, B(v,w)$, where $B \in \Pi^b_i$, we need to show that there is an $\mathrm{FP}^{\Sigma^b_i}$-function $g(u,v)$ and a polynomially bounded advice function $h(n)$ such that

$$\exists w \le v\, B(v,w) \to B(v, g(h(|v|), v)).$$

We consider the Δ^b_{i+1}-relation

$$R(\langle v_1, \dots, v_r\rangle, \langle w_1, \dots, w_s\rangle) \Leftrightarrow s \leq r \wedge \forall \ell \leq s\,(w_\ell \leq v_\ell \wedge B(v_\ell, w_\ell)).$$

By an application of the KPT witnessing theorem to an instance of Σ^b_{i+1}-*LMAX*, provable in T^i_2 by assumption, we obtain a combinatorial principle called Ω_i which states that we can compute a length-maximal b such that $R(a, b)$ from a by a certain counterexample computation in constantly many rounds. Using Ω_i, we define a certain algorithm for computing a pair $\langle \ell, w\rangle$ from $a = \langle v_1, \dots, v_k\rangle$. Let $V_1 = \{|v| = n : \exists w \leq v\, B(v, w)\}$. If Q is a $(k-1)$-element subset of V_1, and $v \in V_1 \smallsetminus Q$, we say that Q *helps* v [116] or $\langle Q, v\rangle$ *is good* [121], if there is an ordering $\{v_1, \dots, v_{\ell-1}, v_{\ell+1}, \dots, v_k\}$ of Q such that the algorithm assigns $\langle \ell, w\rangle$ to $\langle v_1, \dots, v_{\ell-1}, v, v_{\ell+1}, \dots, v_k\rangle$, where w is a witness for v (i.e., $w \leq v \wedge B(v, w)$). (Here, k is a constant parameter we obtain along with the principle Ω_i.) Using a counting argument, we constructs sets $V_1 \supseteq V_2 \supseteq \cdots \supseteq V_t$ and $Q_j \subseteq V_j$ for some $t = O(n)$ so that Q_j have $k-1$ elements, Q_j helps all elements of $V_j \smallsetminus V_{j+1}$, and $|V_t| \leq k$. Then we can compute a witness for v by a $\mathrm{FP}^{\Sigma^b_i}$-function g given the sets $Q_1, \dots, Q_t, V_t$ as well as witnesses for all their elements, which will be encoded in the advice $h(n)$.

Now we turn to the formalization. Notice that the assumption $T^i_2 = S^{i+1}_2$ implies $T^i_2 = S_2$ by [39], hence we actually work in full bounded arithmetic; in particular, we can apply our results above to approximately count sets defined by arbitrary bounded formulas.

By inspection of the proof as given in [116] or [121], we see that T^i_2 proves the principle Ω_i (as the conclusion of the KPT witnessing theorem is provable, not just true), the analysis of the algorithm constructing $\langle \ell, w\rangle$ (straightforward, as the number of steps is a standard constant), as well as the final definition of the function g (obvious). The missing part is the construction the sets $Q_1, \dots, Q_{t-1}, V_t$ and the advice string $h(n)$. We close this gap by an application of Corollary 4.5, where $c = k-1$, and $R(Q, v)$ is the "Q helps v" relation. □

Corollary 4.7 *If $T^i_2 = S^{i+1}_2$, then T^i_2 proves $\Sigma^b_\infty = \mathcal{B}(\Sigma^b_{i+1})$, and $\Sigma^b_{i+1} \subseteq \Pi^b_{i+1}/O(1)$.*

Proof: Cook and Krajíček [67] show (in a two-sorted setting) that Theorem 4.6 implies Corollary 4.7 when $i = 0$. Their results relativize in a straightforward way. □

After showing that $PV_1 \vdash \mathrm{NP} \subseteq \mathrm{P/poly}$ implies $PV_1 \vdash \mathrm{PH} = \mathrm{BH}$ (where $\mathrm{BH} = \mathcal{B}(\mathrm{NP})$ is the Boolean hierarchy), Cook and Krajíček [67] also asked whether the converse holds. We can answer their question affirmatively:

Corollary 4.8 *If T^i_2 proves $\Sigma^b_\infty = \mathcal{B}(\Sigma^b_{i+1})$, then T^i_2 proves $\Sigma^b_{i+1} \subseteq \Delta^b_{i+1}/\mathrm{poly}$.*

Proof: The assumption implies $T^i_2 = S_2$ by Zambella [182], which gives the conclusion by Theorem 4.6. □

The base case $i = 0$ of Corollary 4.8 was meanwhile independently shown by Beyersdorff and Müller [30] using a direct proof.

Krajíček [117, 118] has studied connections between validity of variants of *PHP* in first-order structures M, and existence of certain types of abstract counting functions which map definable sets of M to elements of a ring (or semiring), and behave reasonably wrt embeddings, disjoint unions, and Cartesian products. In particular, a structure which admits a so-called nontrivial *approximate Euler characteristic* (see below) satisfies $iWPHP^{2n}_n$, and conversely, any structure which satisfies $iWPHP^{2n}_n$ and an additional principle (any two definable sets are comparable wrt definable embedding) admits a nontrivial approximate Euler characteristic.

Definition 4.9 If R is a partially ordered commutative ring, we write $a \mathrel{\dot\le} b$ if for every rational $q > 1$ there exist $k, l \in \mathbb{N}$ such that $l/k < q$ and $ka \le lb$. We also put $a \doteq b$ iff $a \mathrel{\dot\le} b \wedge b \mathrel{\dot\le} a$.

Let M be a first-order structure, and $\mathrm{Def}(M)$ the set of all subsets of M^k, $k \in \mathbb{N}$, definable with parameters from M. An *approximate Euler characteristic* is a function $\xi\colon \mathrm{Def}(M) \to R$, where R is a partially ordered commutative ring, such that

(i) $\xi(A) = |A|$ for finite A,

(ii) $\xi(A \mathbin{\dot\cup} B) \doteq \xi(A) + \xi(B)$,

(iii) $\xi(A \times B) \doteq \xi(A) \cdot \xi(B)$,

(iv) $\xi(A) \mathrel{\dot\le} \xi(B)$ if A is definably embeddable into B,

for all $A, B \in \mathrm{Def}(M)$. ξ is *trivial* if $R = 0$.

We also consider extra conditions

(v) $\xi(A) \mathrel{\dot\le} c\xi(B)$ if $\xi(f^{-1}[b]) \mathrel{\dot\le} c$ for all $b \in B$,

(vi) $c\xi(B) \mathrel{\dot\le} \xi(A)$ if $c \mathrel{\dot\le} \xi(f^{-1}[b])$ for all $b \in B$,

where $c \in R$, and $f\colon A \to B$ is a definable injection.

Let M be a model of bounded arithmetic formulated in a purely relational language (i.e., we replace functions with their graphs), and consider an interval $[0, a]_M$ as its substructure. Then definable sets in $[0, a]$ are definable in M by a bounded formula, hence $[0, a]$ satisfies $iWPHP^{2n}_n$ if $M \vDash iWPHP^{2n}_n(\Sigma^b_\infty)$. On the other hand, it is not known to satisfy the principle of comparing cardinalities (and it seems rather unlikely to hold in general). Nevertheless, we can show the following.

Theorem 4.10 *Let M be a model of $S_2(\alpha)$, and $a \in M$. Then $[0, a]_M$ with the induced structure admits a* (*totally ordered*) *nontrivial approximate Euler characteristic satisfying the extra conditions* (v), (vi).

Proof: W.l.o.g. assume that a is nonstandard. Let R be the totally ordered ring whose nonnegative part is M. Notice that $x \mathrel{\dot\le} y$ iff $x \le (1 + c^{-1})y$ for some $c > \omega$. Fix $\varepsilon = 1/n$, where $n \in \mathrm{Log}(M) \smallsetminus \omega$ (say, $n = |a|$). If A is a definable set in $[0, a]$, then A is definable in M by a $\Sigma^b_\infty(\alpha)$-formula, hence there exists an $s \in M$ such that $M \vDash (A \precsim_\varepsilon s \wedge A \not\precsim_\varepsilon s - 1)$; we define $\xi(A) = s$. Then ξ is an approximate Euler characteristic by 3.10, 3.13, 3.17, 3.12, 3.16, and 3.11 (as any injection defined by a bounded formula has a retraction definable by a bounded formula). The extra conditions hold for ξ because of Theorems 3.19 and 3.20. □

The class S_2^P, defined independently by Russell and Sundaram [161], and Canetti [48], consists of languages L for which there exists a poly-time predicate R such that

$$x \in L \Rightarrow \exists y \, \forall z \, R(x, y, z),$$
$$x \notin L \Rightarrow \exists z \, \forall y \, \neg R(x, y, z),$$

where $|y|, |z|$ are implicitly bounded by a polynomial in $|x|$. The class S_2^P occupies an interesting position inside the second level of PH: obviously $\mathrm{S}_2^P \subseteq \Sigma_2^P \cap \Pi_2^P$, we also know that $\mathrm{MA} \subseteq \mathrm{S}_2^P$ (hence $\mathrm{BPP} \subseteq \mathrm{S}_2^P$), and $\mathrm{P}^{\mathrm{S}_2^P} = \mathrm{S}_2^P$ (hence $\Delta_2^P \subseteq \mathrm{S}_2^P$) [161], and the standard proof of the Karp–Lipton theorem shows that $\mathrm{NP} \subseteq \mathrm{P/poly}$ implies $\mathrm{PH} = \mathrm{S}_2^P$. The definition of S_2^P does in no way guarantee abundance of witnesses for the existential quantifiers; surprisingly, Cai [47] has shown that nevertheless $\mathrm{S}_2^P \subseteq \mathrm{ZPP}^{\mathrm{NP}}$. We will formalize this result in bounded arithmetic. (The other results mentioned above are also easy to prove in bounded arithmetic, we leave the details to the reader.)

Theorem 4.11 (*in* $T^1_2 + rWPHP(PV_2)$) *The complexity class* S_2^P *is contained in* $\mathrm{ZPP}^{\mathrm{NP}}$*.*

Proof: Let $L \in \mathrm{S}_2^P$. Fix a constant c, and a poly-time relation R such that

$$x \in L \Rightarrow \exists y < 2^{|x|^c} \, \forall z < 2^{|x|^c} \, R(x, y, z),$$
$$x \notin L \Rightarrow \exists z < 2^{|x|^c} \, \forall y < 2^{|x|^c} \, \neg R(x, y, z).$$

By the relativization of the formalized Wilkie's witnessing theorem [93, P. 1.16] applied to Corollary 4.4, there exists a $\mathrm{ZPP}^{\mathrm{NP}}$-predicate P definable in $T^1_2 + rWPHP(PV_2)$ such that the same theory proves

$$P(x) \Rightarrow \exists D \subseteq 2^{|x|^c} \, \forall z < 2^{|x|^c} \, \exists y \in D \, R(x, y, z),$$
$$\neg P(x) \Rightarrow \exists D \subseteq 2^{|x|^c} \, \forall y < 2^{|x|^c} \, \exists z \in D \, \neg R(x, y, z).$$

Clearly, the conditions implied by $x \in L$ and $\neg P(x)$ are contradictory, and vice versa, hence $x \in L$ iff $P(x)$. □

Another application of approximate counting in computational complexity is the equivalence of public-coin and private-coin interactive protocols [83]. We illustrate it on the example of the isomorphism problem: given two structures G_0 and G_1 (as tables) of the same signature, determine whether $G_0 \simeq G_1$. (The most prominent, and indeed universal, special case is when the structures are graphs.) The problem is obviously in NP, and its complement admits a simple private-coin interactive proof system: the verifier picks randomly an $i < 2$, and a permutation π, and sends $\pi(G_i)$ to the prover, who has to determine i. If $G_0 \not\simeq G_1$, a (computationally unlimited) prover can succeed with probability 1, whereas if $G_0 \simeq G_1$, no prover can do any better (or worse, for that matter) than 1/2. It is much harder to construct a public-coin proof system (i.e., an AM-algorithm) for the same problem, and it requires approximate counting.

Theorem 4.12 (*in* $T^1_2 + sWPHP(PV_2)$) *The isomorphism problem is in* coAM.

Proof: For simplicity, we will ignore floor and ceiling signs. Put $\varepsilon = 1/42$. As in the proof of Lemma 3.14, there exists a definable prAM-problem $L = \langle L^+, L^- \rangle$ such that

$$\begin{aligned} X_b &\not\precsim_\varepsilon a \Rightarrow \langle a, b\rangle \in L^+, \\ X_b &\precsim_\varepsilon \tfrac{42}{43}a \Rightarrow \langle a, b\rangle \in L^- \end{aligned}$$

for any parametric family of NP-sets X_b. As prAM is closed under bounded existential quantification, conjunction, and disjunction, we can define a prAM problem $L = \langle L^+, L^- \rangle$ such that

$$\begin{aligned} \exists a \left(\big(A_0 \not\precsim_\varepsilon a \vee A_1 \not\precsim_\varepsilon a\big) \wedge W_0 \cup W_1 \not\precsim_\varepsilon \frac{3n!}{2a} \right) &\Rightarrow \langle G_0, G_1\rangle \in L^+, \\ \forall a \left(\big(A_0 \precsim_\varepsilon \tfrac{42}{43}a \wedge A_1 \precsim_\varepsilon \tfrac{42}{43}a\big) \vee W_0 \cup W_1 \precsim_\varepsilon \frac{4n!}{3a} \right) &\Rightarrow \langle G_0, G_1\rangle \in L^-, \end{aligned}$$

where G_0, G_1 are structures with domain n, and A_i and W_i are the Σ^b_1-sets

$$\begin{aligned} A_i &= \mathrm{Aut}(G_i) = \{\pi \in S_n : \pi(G_i) = G_i\}, \\ W_i &= \{\pi(G_i) : \pi \in S_n\}, \end{aligned}$$

where S_n is the set of all permutations of n. It suffices to show

$$\begin{aligned} G_0 \not\simeq G_1 &\Rightarrow \langle G_0, G_1\rangle \in L^+, \\ G_0 \simeq G_1 &\Rightarrow \langle G_0, G_1\rangle \in L^-. \end{aligned}$$

Claim 4.12.1

(i) *If $A_i \precsim_\varepsilon a$, and $W_i \precsim_\varepsilon b$, then $ab \geq \frac{5}{6}n!$.*

(ii) *If $A_i \not\precsim_\varepsilon a$, and $W_i \not\precsim_\varepsilon b$, then $ab \leq \frac{10}{9}n!$.*

Proof: (i): If $H \in W_i$, and π_0 is any permutation such that $H = \pi_0(G_i)$, then the mapping $\pi \mapsto \pi_0 \circ \pi$ is a poly-time bijection of A_i onto $M(H) := \{\pi : \pi(G_i) = H\}$, with $\pi \mapsto \pi_0^{-1} \circ \pi$ being its inverse. It follows that $M(H) \precsim_\varepsilon \frac{43}{42}a$ by Corollary 3.11, thus

$$M := \{\langle \pi, \pi(G_i)\rangle : \pi \in S_n\} = \dot{\bigcup_{H \in W_i}} M(H) \precsim_\varepsilon \tfrac{9}{8}ab$$

by Theorem 3.19. Clearly $\pi \mapsto \langle \pi, \pi(G_i)\rangle$ is a bijection of S_n onto M. Moreover, there exists a poly-time enumeration of S_n by $n!$, hence $\frac{7}{6}ab \twoheadrightarrow n!$, which implies $n! \leq \frac{6}{5}ab$ by *sWPHP*.

(ii): Similar. □ (Claim 4.12.1)

Assume $G_0 \not\simeq G_1$, and find a_0, a_1 such that $A_i \not\precsim_\varepsilon a_i$, $A_i \precsim_\varepsilon \frac{43}{42}a_i$. We have $W_i \not\precsim_\varepsilon 13n!/16a_i$ by (i). The sets W_i are disjoint, hence

$$W_0 \cup W_1 \not\precsim_\varepsilon \frac{3n!}{4}\left(\frac{1}{a_0} + \frac{1}{a_1}\right) \geq \frac{3n!}{2a_i}$$

for some i by Theorem 3.17, thus $\langle G_0, G_1\rangle \in L^+$.

On the other hand, assume $G_0 \simeq G_1$, and let a_i be such that $A_i \precsim_\varepsilon \frac{42}{43}a_i$, $A_i \not\precsim_\varepsilon \frac{20}{21}$. Then $W_0 \cup W_1 = W_i \precsim_\varepsilon 7n!/6a_i$ by (ii), as $W_0 = W_1$. Consequently $\langle G_0, G_1\rangle \in L^-$. □

Notice that if we change the definition of AM formalized in bounded arithmetic to use $\precsim$ instead of $\preceq$ (which might be a good idea anyway), the statement of Theorem 4.12 becomes $\forall\Sigma^b_2$, hence we can prove it already in $T^1_2 + rWPHP(PV_2)$.

Acknowledgements

I am indebted to Neil Thapen and Jan Krajíček for insightful discussions, and many useful suggestions, and to Pavel Pudlák for clarification of [121].

The research was supported by grant IAA1019401 of GA AV ČR, and grant 1M0545 of MŠMT ČR.

Chapter III

Abelian groups and quadratic residues in weak arithmetic

Abstract

We investigate the provability of some properties of abelian groups and quadratic residues in variants of bounded arithmetic. Specifically, we show that the structure theorem for finite abelian groups is provable in $S^2_2 + iWPHP(\Sigma^b_1)$, and use it to derive Fermat's little theorem and Euler's criterion for the Legendre symbol in $S^2_2 + iWPHP(PV)$ extended by the pigeonhole principle $PHP(PV)$. We prove the quadratic reciprocity theorem (including the supplementary laws) in the arithmetic theories $T^0_2 + Count_2(PV)$ and $I\Delta_0 + Count_2(\Delta_0)$ with modulo-2 counting principles.

1 Introduction

Bounded arithmetic is primarily studied because of its connections to complexity theory, see e.g. Buss [37], Krajíček [116], Cook and Nguyen [68]. However, as with other systems of formal arithmetic, it is also interesting to note which mathematical (typically, number-theoretic or combinatorial) theorems are provable in weak arithmetical theories, or to put it differently, to find as weak a natural theory as possible which proves a given statement. (This approach is called "bounded reverse mathematics" by Nguyen [132], in analogy with "reverse mathematics" [78, 165]. However, note that unlike standard reverse mathematics, in bounded arithmetic one usually does not obtain the *equivalence* of the statement to the theory.) Examples include the proof of infinitude of prime numbers in $I\Delta_0 + WPHP(\Delta_0)$ by Paris, Wilkie, and Woods [146], the proof of Lagrange's four-square theorem in $I\Delta_0 + WPHP(\Delta_0)$ by Berarducci and Intrigila [29], the proof of the prime number theorem in $I\Delta_0 + \exp$ by Cornaros and Dimitracopoulos [71], or the proof of a discrete version of the Jordan curve theorem in $V^0[2]$ by Nguyen [132].

The first contribution of the present paper is a proof of the *structure theorem for finite abelian groups*—stating that every finite abelian group is isomorphic to a direct sum of cyclic groups (see e.g. Mac Lane and Birkhoff [124])—in the theory $S^2_2 + iWPHP(\Sigma^b_1)$ (a subtheory of Buss's T^2_2), where we represent a finite group by a Σ^b_1-definable binary operation on a bounded set of numbers. The easy part of the structure theorem, viz. representation of any finite abelian group as a direct sum of p-groups, was formalized in $I\Delta_0 + \Omega_1$ by D'Aquino and Macintyre [72]. Here we prove the full structure theorem, and we find a refined proof with the aim of bringing the complexity of the theory needed to formalize the argument down as low as possible.

Our motivating example, and main application, for the structure theorem is *Fermat's little theorem* (*FLT*), stating

$$a^p \equiv a \pmod p$$

for a prime p. FLT was considered in the context of bounded arithmetic by Krajíček and Pudlák [120], who have shown that S^1_2 does not prove FLT if the RSA cryptosystem is secure. (Actually, their argument applies to a weak corollary of FLT stating that multiplication modulo a prime is a torsion group, which is provable using the weak pigeonhole principle.) Jeřábek [93] proved that FLT is in S^1_2 equivalent to the correctness of the Rabin–Miller probabilistic primality testing algorithm. It remains an open problem whether FLT is provable in the bounded arithmetic S_2. Here we derive FLT using the structure theorem for finite abelian groups in the theory $S^2_2 + \mathit{iWPHP}(\mathit{PV}) + \mathit{PHP}(\mathit{PV})$, which includes the strong pigeonhole principle for polynomial-time functions.

Next to Fermat's little theorem, we consider *Euler's criterion* for quadratic residues stating

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod p$$

for an odd prime p, where $(a|p)$ is the Legendre symbol. We will show in S^1_2 that Euler's criterion is equivalent to FLT together with a statement ensuring that the quadratic character $a \mapsto a^{(p-1)/2} \bmod p$ is nontrivial. In particular, we obtain a proof of Euler's criterion in $S^2_2 + \mathit{iWPHP}(\mathit{PV}) + \mathit{PHP}(\mathit{PV})$.

Finally, we will discuss another result on quadratic residues: the *quadratic reciprocity* theorem. Quadratic reciprocity, originally proved by Carl Friedrich Gauss, is one of the most famous theorems of elementary number theory. Apart from Gauss (who gave no less than eight different proofs of the theorem), over 200 proofs of quadratic reciprocity have been published by various authors. As far as bounded arithmetic is concerned, the work of D'Aquino and Macintyre [73] on quadratic forms aims towards proving quadratic reciprocity or at least some of its special cases in S_2, and Cornaros [70] formalized a standard textbook proof of quadratic reciprocity in $I\mathcal{E}^2_*$ (a rather strong theory corresponding to the Grzegorczyk class $\mathcal{E}^2 = \mathrm{LinSpace}$). The supplementary laws were proved by Berarducci and Intrigila [29] in $I\Delta_0$ extended with modular counting principles.

Observe that many elementary proofs of quadratic reciprocity (e.g., proofs based on Gauss's lemma or Zolotarev's lemma, and Eisenstein's proof) directly or indirectly involve counting the parity of sets. We will show, using a proof which appears to be new even outside the context of bounded arithmetic, that rudimentary counting modulo 2 indeed suffices to prove the theorem. More precisely, we do not even require the existence of modulo-2 counting functions, as we can witness the parity of all sets we need by explicit functions. We only need to assume the *modulo-*2 *counting principle* Count_2; in detail, we can prove the law of quadratic reciprocity as well as the supplementary laws and multiplicativity of the Legendre symbol in the theories $T^0_2 + \mathit{Count}_2(\mathit{PV})$ and $I\Delta_0 + \mathit{Count}_2(\Delta_0)$. We also generalize these statements to the Jacobi symbol, and prove the soundness of the standard polynomial-time algorithm for the Jacobi symbol in $S^1_2 + \mathit{Count}_2(PV)$.

2 Preliminaries

We review below basic facts about bounded arithmetic, we refer the reader to Krajíček [116] for more details.

We will work with two kinds of arithmetical systems: theories based on $I\Delta_0$ (introduced by Parikh [142]), and Buss's theories [37]. $I\Delta_0$ is a theory in the basic language of arithmetic $L_{PA} = \langle 0, S, +, \cdot, \leq \rangle$. A formula φ is bounded (or Δ_0) if every quantifier in φ is bounded, i.e., it has one of the forms

$$\begin{aligned}\exists x \leq t\, \psi(x) &:= \exists x\, (x \leq t \wedge \psi(x)),\\ \forall x \leq t\, \psi(x) &:= \forall x\, (x \leq t \to \psi(x)),\end{aligned}$$

where t is a term not containing the variable x. The axioms of $I\Delta_0$ include the axioms of Robinson's arithmetic Q (which state basic inductive properties of addition, multiplication, and ordering), and the induction schema Δ_0-*IND*:

$$(\varphi\text{-}\mathit{IND}) \qquad \varphi(0) \wedge \forall x\, (\varphi(x) \to \varphi(x+1)) \to \varphi(a).$$

We formulate Buss's theories in the language $L = \langle 0, S, +, \cdot, \leq, \#, |x|, \lfloor x/2^y \rfloor \rangle$, where the intended meaning of the symbols is $|x| = \lceil \log_2(x+1) \rceil$, $x \# y = 2^{|x|\cdot|y|}$. A bounded quantifier is called sharply bounded if its bounding term is of the form $|t|$. A formula is $\Sigma^b_0 = \Pi^b_0$ if all its quantifiers are sharply bounded. A formula is Σ^b_{i+1} (Π^b_{i+1}) if it is constructed from $\Sigma^b_i \cup \Pi^b_i$-formulas by means of conjunctions, disjunctions, sharply bounded quantifiers, and existential (universal, respectively) bounded quantifiers. The set of Boolean combinations of Σ^b_i-formulas is denoted by $\mathcal{B}(\Sigma^b_i)$.

The theory T^i_2 is axiomatized by a finite set *BASIC* of open axioms stating basic properties of the symbols of L, and the schema Σ^b_i-*IND*. If $i > 0$, the theory S^i_2 consists of *BASIC* and the polynomial induction schema Σ^b_i-*PIND*

$$(\varphi\text{-}\mathit{PIND}) \qquad \varphi(0) \wedge \forall x\, (\varphi(\lfloor x/2 \rfloor) \to \varphi(x)) \to \varphi(a).$$

Alternatively, S^i_2 can be axiomatized over *BASIC* by the length induction schema Σ^b_i-*LIND*

$$(\varphi\text{-}\mathit{LIND}) \qquad \varphi(0) \wedge \forall x\, (\varphi(x) \to \varphi(x+1)) \to \varphi(|a|),$$

the length minimization schema Σ^b_i-*LMIN*

$$(\varphi\text{-}\mathit{LMIN}) \qquad \varphi(a) \to \exists b \leq a\, (\varphi(b) \wedge \forall x\, (|x| < |b| \to \neg\varphi(x))),$$

or the analogous length maximization schema Σ^b_i-*LMAX*.

The theory S^i_2 also proves the Σ^b_i-comprehension schema

$$(\varphi\text{-}\mathit{COMP}) \qquad \exists b < a \# 1\, \forall i < |a|\, (i \in b \leftrightarrow \varphi(i)),$$

where we define $i \in x$ iff $\lfloor x/2^i \rfloor$ is odd.

The basic relationship of these theories is given by $T^i_2 \subseteq S^{i+1}_2 \subseteq T^{i+1}_2$. Buss's witnessing theorem implies that S^{i+1}_2 is a $\forall\Sigma^b_{i+1}$-conservative extension of T^i_2. The theory $S_2 = \bigcup_i S^i_2 = \bigcup_i T^i_2$ is an extension of $I\Delta_0 + \Omega_1$ by definitions.

All these theories can be relativized by introducing an extra unary predicate α in the language. $\Sigma^b_i(\alpha)$ and $\Pi^b_i(\alpha)$ formulas in the expanded language $L(\alpha)$ are defined as above. The theories $S^i_2(\alpha)$ and $T^i_2(\alpha)$ include the (polynomial) induction schema for $\Sigma^b_i(\alpha)$-formulas, but no other axioms about the predicate α.

PV is an equational theory introduced by Cook [63]. Its language contains function symbols for all polynomial-time algorithms, introduced inductively using limited recursion on notation (cf. Cobham [57]). It is axiomatized by defining equations of its function symbols, and a derivation rule similar to *PIND*. We will denote the set of *PV*-function symbols also by *PV*. All *PV*-function have provably total Σ^b_1-definitions in T^0_2 such that T^0_2 proves their defining equations and $\Sigma^b_0(PV)$-*IND* (Jeřábek [94]), furthermore every $\Sigma^b_i(PV)$-formula is equivalent to a Σ^b_i-formula for $i > 0$, hence we will use *PV*-functions freely in T^0_2 and its extensions.

In particular, sequence encoding is available in T^0_2. We will denote by $(s)_i$ and $\mathrm{lh}(s)$ *PV*-functions which give the ith element of the sequence s, and the length of s, respectively. We will often write just s_i instead of $(s)_i$. Conversely, we will write sequences using angle brackets, so that $s = \langle s_i \mid i < \mathrm{lh}(s)\rangle$ if s encodes a sequence. Notice that always $\mathrm{lh}(s) \in \mathrm{Log}$ as $\mathrm{lh}(s) \le |s|$, where we write $x \in \mathrm{Log}$ as a shorthand for the formula $\exists y\, x = |y|$. If $s(i)$ is given by a *PV*-function, then $f(x) = \langle s(i) \mid i < |x|\rangle$ is also definable by a *PV*-function.

If f is a definable function (possibly with parameters), the injective weak pigeonhole principle $iWPHP(f)$ is the axiom

$$a > 0 \to \exists x < 2a\, f(x) \ge a \vee \exists x < x' < 2a\, f(x) = f(x').$$

If Γ is a set of functions (or formulas, meaning the functions with Γ-definable graph), then we put $iWPHP(\Gamma) = \{iWPHP(f) \mid f \in \Gamma\}$. The multifunction weak pigeonhole principle $mWPHP(R)$ is the axiom

$$a > 0 \to \exists x < 2a\, \forall y < a\, \neg R(x,y) \vee \exists x < x' < 2a\, \exists y < a\, (R(x,y) \wedge R(x',y)),$$

where R is a definable binary relation. Again, we put $mWPHP(\Gamma) = \{mWPHP(R) \mid R \in \Gamma\}$ for a set Γ of formulas. Note that $mWPHP(\Gamma)$ implies $iWPHP(\Gamma)$. The schema $mWPHP(\Sigma^b_i(\alpha))$ for $i > 0$ is provable in $T^{i+1}_2(\alpha)$ by Maciel, Pitassi, and Woods [125].

3 Finite abelian groups

Definition 3.1 (in $S^1_2(\alpha)$) A *definable finite abelian group* is a structure $\langle G, +\rangle$, where G is a nonempty subset of an interval $[0, t)$ (which we will denote simply as t), and $+$ is a definable binary operation on G satisfying the usual axioms of abelian groups:

$$x + (y + z) = (x + y) + z,$$
$$x + y = y + x,$$
$$\exists v\, (x + v = y).$$

We will denote the group $\langle G, +\rangle$ by just G, if there is no danger of confusion. If Γ is a set of formulas, and G and $+$ are definable by Γ-formulas (with parameters), we say that $\langle G, +\rangle$ is a Γ-definable finite abelian group, or simply Γ *finite abelian group.* Observe that a group is $\Sigma^b_1(\alpha)$ iff it is definable by a nondeterministic circuit with oracle α; we may identify the group with (the number representing) the circuit, hence it makes sense to speak of, e.g., sequences of groups. Notice that G is automatically $\Sigma^b_1(\alpha)$-definable by the formula $\exists y < t\, x + x = y$ if $+$ is $\Sigma^b_1(\alpha)$.

Definition 3.2 (in $S^1_2(\alpha)$) For any positive integer n, $C(n)$ denotes the cyclic group of addition modulo n. The trivial abelian group $C(1)$ is also denoted 0.

Let $\langle G_i \mid i < k\rangle$ be a sequence of $\Sigma^b_1(\alpha)$ abelian groups such that $G_i \subseteq t_i$ for each $i < k$. (Notice that $k \in \mathrm{Log}$, as it is the length of a sequence.) The *direct sum* $\bigoplus_{i<k} G_i$ is the $\Sigma^b_1(\alpha)$ group

$$G = \{\langle a_i \mid i < k\rangle \mid \forall i < k\, a_i \in G_i\} \subseteq \prod_{i<k} t_i$$

with addition defined by

$$\langle a_i \mid i < k\rangle + \langle b_i \mid i < k\rangle = \langle a_i + b_i \mid i < k\rangle.$$

Here, $a = \langle a_i \mid i < k\rangle$ for definiteness refers to the specific sequence encoding function $a_i = \lfloor a/\prod_{j<i} t_j\rfloor \bmod t_i$, whence the bound on the domain of G.

Lemma 3.3 *(in $S^1_2(\alpha)$) If G is a $\Sigma^b_1(\alpha)$ finite abelian group, there exists a $\Sigma^b_1(\alpha)$-definable function nx such that $0x = 0$, $1x = x$, $(n + m)x = nx + mx$, $n(x + y) = nx + ny$, $(nm)x = n(mx)$, and $(-n)x = -nx$ for every $x, y \in G$, and integers n, m.*

If $+$ is defined by a $PV(\alpha)$-function, then so is nx for nonnegative n.

Proof: We can define nx for nonnegative n by limited recursion on notation:

$$0x = 0,$$
$$(2n)x = nx + nx,$$
$$(2n + 1)x = (2n)x + x.$$

We put $(-n)x = -nx$. Verification of the properties is then straightforward. □

Note that the extended Euclidean algorithm can be formalized by a PV-function, and analyzed in a straightforward way in T^0_2. In particular, coprimeness is Δ^b_1 in T^0_2, and T^0_2 proves Bézout's lemma. For the sake of completeness, we include a simpler proof in S^1_2:

Lemma 3.4 (Bézout's lemma) *(in S^1_2) For every a, b, there exist integers u, v such that $d = ua + vb$ divides both a and b (and therefore $d = \gcd(a, b)$).*

Proof: If $b = 0$, we have $\gcd(a, b) = a = 1a + 0b$, and similarly if $a = 0$, hence we may assume that $a, b > 0$. Notice that for any $0 \le d \le a$, the property that there exist integers u, v such that $d = ua + vb$ is Σ^b_1, as it is equivalent to

$$\exists u \le b\, \exists v \le a\, d = ua - vb.$$

Indeed, assume that $d = u'a + v'b$, and write $u' = xb + u$, where $0 < u \leq b$. Then $d = ua - vb$ for $v = -(xa + v')$, and $0 = a - a \leq vb = ua - d \leq ab$, thus $0 \leq v \leq a$.

We have $a = 1a + 0b$, hence by Σ^b_1-*LMIN*, there exists $0 < d \leq a$ of minimal length such that $d = ua + vb$ for some integers u, v. We can write $a = xd \pm d'$, where $0 \leq d' \leq \lfloor d/2 \rfloor$. Then $|d'| < |d|$, and $d' = \pm(1 - xu)a \mp (xv)b$, which contradicts minimality of d unless $d' = 0$, i.e., d divides a. By a symmetric argument d divides b, too. □

Recall that a *torsion element* of a group G is an $x \in G$ such that $nx = 0$ for some $n > 0$. A *torsion group* is an abelian group consisting of torsion elements.

Lemma 3.5 (*in* $S^1_2(\alpha)$) *If x is a torsion element of a $\Sigma^b_1(\alpha)$ finite abelian group, there exists a unique positive integer $o(x)$ (the* order *of x) such that*

$$ax = 0 \iff o(x) \mid a$$

for every a.

Proof: By $\Sigma^b_1(\alpha)$-*LMIN*, there exists $o(x) > 0$ such that $o(x)x = 0$ of minimal length. Assume that $ax = 0$, and let $d = \gcd(a, o(x))$. By Bézout's lemma, there exist integers u, v such that $d = ua + vo(x)$, hence $dx = 0$. If d is a proper divisor of $o(x)$, then $|d| < |o(x)|$, which contradicts the choice of $o(x)$. Therefore $d = o(x)$, and $o(x) \mid a$. Uniqueness of $o(x)$ is obvious. □

Lemma 3.6 (*in* $S^1_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$) *Any $\Sigma^b_1(\alpha)$ finite abelian group is a torsion group.*

Proof: Let $x \in G \subseteq t$. By *iWPHP*, there exist $a < b < 2t$ such that $ax = bx$, hence $(b-a)x = 0$. □

Remark 3.7 Similarly to Lemma 3.6, $S^1_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$ also proves that any finite structure with a $\Sigma^b_1(\alpha)$-definable associative, commutative, and cancellative binary operation is an abelian group.

Before we turn to the main structure theorem, we prove the simpler decomposition to p-primary components below. It is a consequence of the structure theorem, but we prove it separately because we can formalize the proof in a weaker theory than the full structure theorem. The decomposition to p-primary components was proved in $I\Delta_0 + \Omega_1$ by D'Aquino and Macintyre [72] (formulated for multiplicative groups of the prime fields $\mathbb{F}_p$, but the argument works for general abelian groups).

Definition 3.8 (in $S^1_2(\alpha)$) If $\langle G, + \rangle$ is a $\Sigma^b_1(\alpha)$ finite abelian group, and p is a prime, then the *p-primary component of G* is defined by

$$G_p = \{x \in G \mid \exists e\, p^{|e|} x = 0\}.$$

Notice that G_p is a subgroup of G. If G is a torsion group with $o(x) \leq t$ for every x, then G_p is $\Sigma^b_1(\alpha)$, as we can bound e by t.

A *p-group* is a $\Sigma^b_1(\alpha)$ finite abelian group $\langle G, + \rangle$ such that $G = G_p$.

Theorem 3.9 *(in $S^1_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$) Let $\langle G, +\rangle$ be a $\Sigma^b_1(\alpha)$ finite abelian group. There exists a sequence $\langle p_i \mid i < k\rangle$ of pairwise distinct primes, such that the mapping*

$$\varphi\colon \bigoplus_{i<k} G_{p_i} \to G$$

defined by $\varphi(\langle x_i \mid i < k\rangle) = \sum_{i<k} x_i$ is an isomorphism. If a prime p does not appear in $\{p_i \mid i < k\}$, then $G_p = 0$.

Proof: Let $G \subseteq t$. By $\Sigma^b_1(\alpha)$-*LMAX*, there exists the maximal k with the property that $k \le |t| + 1$ and there exists a sequence $\langle z_i \mid i < k\rangle$ of nonzero elements of G, and a sequence $\langle p_i \mid i < k\rangle$ of pairwise coprime integers $p_i \le 2t$ such that $p_i z_i = 0$. (Notice that coprimeness is Σ^b_1 by Bézout's lemma, and all other universal quantifiers in this definition are sharply bounded, hence the property is indeed $\Sigma^b_1(\alpha)$.) By $\Sigma^b_1(\alpha)$-*LMIN*, there exists the smallest l such that there exist $\vec{z}$ and $\vec{p}$ as above with $\sum_i |p_i| \le l$. Fix the witnesses $\vec{z}$ and $\vec{p}$.

By the choice of l, all p_i are primes: if $p_i = mn$ is a nontrivial factorization, then either $mz_i = 0$, or $y = mz_i$ is a nonzero element such that $ny = 0$. We can thus replace p_i with m or n, which contradicts the minimality of l. In particular, $p_i = o(z_i)$ for all $i < k$.

Define $f\colon \prod_i p_i \to G$ by $f(\langle a_i \mid i < k\rangle) = \sum_i a_i z_i$. We claim that f is injective. Indeed, let $\sum_i a_i z_i = \sum_i a'_i z_i$, and fix $i < k$. Put $q = \prod_{j\ne i} p_j$. We have $0 = q\sum_j (a'_j - a_j) z_j = q(a'_i - a_i) z_i$, hence $o(z_i) = p_i \mid q(a'_i - a_i)$. However, q is coprime to p_i, thus $p_i \mid a'_i - a_i$. As $0 \le a'_i, a_i < p_i$, this implies $a_i = a'_i$. By $iWPHP(\Sigma^b_1(\alpha))$, we obtain $2^k \le \prod_i p_i < 2t$, hence $k \le |t|$.

Consequently, if $p \ne p_i$ for all i, then $G_p = 0$. Indeed, if $x \ne 0$, and $p^e x = 0$, we can extend $\vec{p}$ and $\vec{z}$ by p^e and x (respectively), which contradicts the maximality of k.

Clearly, φ is a group homomorphism. We claim that φ is injective, i.e., $\ker(\varphi) = 0$. Let thus $\sum_i x_i = 0$, $x_i \in G_{p_i}$. Consider $i < k$, and put $q = \prod_{j\ne i} p_j^{|t|}$. We have $0 = q\sum_j x_j = qx_i$, and $p_i^{|t|} x_i = 0$, hence $x_i = 0$, as q and $p_i^{|t|}$ are coprime.

It remains to show that φ is onto. Let thus $x \in G$, and using $\Sigma^b_1(\alpha)$-*LMIN* find a of minimal length such that $ax \in \operatorname{rng}(\varphi)$. We have $bx \in \operatorname{rng}(\varphi)$ iff $a \mid b$, as in the proof of Lemma 3.5. If $a = 1$, the proof is finished. Let us assume for contradiction $a > 1$, and choose a prime $p \mid a$. Put $q = \prod_{p_i \ne p} p_i^{|t|}$, and $y = (qa/p)x$. We have $ax = \sum_i x_i$ for some $x_i \in G_{p_i}$, hence

$$py = qax = q\sum_i x_i = \begin{cases} qx_i & p = p_i \text{ for some } i,\\ 0 & \text{otherwise.}\end{cases}$$

If $p = p_i$, we have $py = qx_i \in G_p$, hence also $y \in G_p \subseteq \operatorname{rng}(\varphi)$. If $p \ne p_i$ for all i, then $y \in G_p = 0 \subseteq \operatorname{rng}(\varphi)$. In both cases, we obtain $(qa/p)x \in \operatorname{rng}(\varphi)$, hence $a \mid (qa/p)$. This implies $p \mid q$, which contradicts the definition of q. □

Corollary 3.10 *(in $S^1_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$) If G is a $\Sigma^b_1(\alpha)$ finite abelian group, then there exists $n > 0$ such that $nG = 0$.*

Proof: Let $n = \prod_i p_i^{|t|}$, where $\vec{p}$ and t is as in Theorem 3.9. Then $nx = 0$ for all $x \in G$. □

Corollary 3.11 *(in $S^1_2(\alpha) + iWPHP(PV(\alpha))$) If $+$ is definable by a $PV(\alpha)$-function, then so is $-$.*

Proof: Under the assumption all instances of $iWPHP(\Sigma^b_1(\alpha))$ used above were actually instances of $iWPHP(PV(\alpha))$. If n is as in Corollary 3.10, then $-x = (n-1)x$ is $PV(\alpha)$ by Lemma 3.3. □

The main result of this section is the structure theorem below.

Theorem 3.12 *(in $S^2_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$) Let G be a $\Sigma^b_1(\alpha)$ finite abelian group. There exists a sequence of prime powers $P = \langle p_i^{e_i} \mid i < k\rangle$ with $e_i > 0$, and a sequence $\langle x_i \mid i < k\rangle$ of elements of G, such that the $\Sigma^b_1(\alpha)$-function*

$$\varphi\colon \bigoplus_{i<k} C(p_i^{e_i}) \to G$$

defined by

$$\varphi(\langle \alpha_i \mid i < k\rangle) = \sum_{i<k} \alpha_i x_i$$

is a group isomorphism. Moreover, P is unique up to permutation of indices.

Remark 3.13 No claim is being made on uniformity of the $\Sigma^b_1(\alpha)$-isomorphism, as the proof will give no nontrivial estimate on the complexity of finding the sequence $\vec{x}$.

Proof: Existence: let us say that $\langle x_i \mid i < k\rangle$ is an *independent sequence* with exponents $\langle m_i \mid i < k\rangle$ if

$$(1) \qquad \forall i < k\,(x_i \in G \wedge m_i > 1 \wedge m_i x_i = 0) \wedge \forall \vec{\alpha} \in \prod_{i<k} m_i \Big(\sum_{i<k} \alpha_i x_i = 0 \to \vec{\alpha} = \vec{0}\Big).$$

Notice that (1) is a $\mathcal{B}(\Sigma^b_1(\alpha)) \subseteq \Sigma^b_2(\alpha)$-formula, as the quantifier $\forall i < k$ is sharply bounded. If $\vec{x}$ is an independent sequence with exponents $\vec{m}$, then the mapping $\varphi\colon \bigoplus_{i<k} C(m_i) \to G$ defined by

$$\varphi(\langle \alpha_i \mid i < k\rangle) = \sum_{i<k} \alpha_i x_i$$

is easily seen to be a homomorphism, and $\ker(\varphi) = 0$, hence φ is injective. As φ is $\Sigma^b_1(\alpha)$, we can apply $iWPHP(\varphi)$, which implies that $\prod_i m_i \le 2t$, where $G \subseteq t$. In particular,

$$k \le \Big|\prod_{i<k} m_i\Big| \le |t| + 1.$$

We apply the $\Sigma^b_2(\alpha)$-$LMAX$ principle to fix the maximal k such that there exists an independent sequence of length k. Then we apply $\Sigma^b_2(\alpha)$-$LMAX$ once more to find an independent sequence $\langle x_i \mid i < k\rangle$ with exponents $\langle m_i \mid i < k\rangle$ such that $|\prod_i m_i|$ is maximal.

Claim 3.13.1 *Each m_i is a prime power.*

Proof: Assume for contradiction that m_i is not a prime power. By Claim 2 in [93, Ex. 1.13], we can write $m_i = ab$, where $a, b > 1$ are coprime. By Bézout's lemma, we can choose integers u, v such that $ua + vb = 1$. Put $y = uax_i$, $z = vbx_i$. Clearly, $by = 0 = az$. We will show that $\langle y, z, x_j \mid j \neq i\rangle$ is an independent sequence with exponents $\langle b, a, m_j \mid j \neq i\rangle$, contradicting the definition of k.

Let thus $\alpha < b$, $\beta < a$, $\alpha_j < m_j$ be such that $\alpha y + \beta z + \sum_{j\neq i} \alpha_j x_j = 0$. By the definition of y, z, we have

$$(\alpha ua + \beta vb)x_i + \sum_{j\neq i} \alpha_j x_j = 0,$$

thus the independence of $\vec{x}$ implies that $\alpha_j = 0$ for $j \neq i$, and $m_i \mid \alpha ua + \beta vb$. In particular, $a \mid \beta vb$, and as a is coprime to vb, $a \mid \beta$, hence $\beta = 0$. We can show $\alpha = 0$ by a symmetric argument. □ (Claim 3.13.1)

We write $m_i = p_i^{e_i}$, where p_i is prime, and define the mapping φ as above.

Claim 3.13.2 *φ is surjective.*

Proof: Assume for contradiction that there exists an element $x \in G$ such that $x \notin \operatorname{rng}(\varphi)$. By Lemma 3.6 and a generalization of Lemma 3.5, there exists an $a > 0$ such that $bx \in \operatorname{rng}(\varphi)$ iff $a \mid b$ for any integer b. As $a > 1$, there is a prime $p \mid a$. If $x' = (a/p)x$, then $bx' \in \operatorname{rng}(\varphi)$ iff $p \mid b$, hence we may simply assume that $a = p$ is prime. Write

$$px = \sum_i \beta_i x_i.$$

If i is such that $p \neq p_i$, then m_i is coprime to p, hence there exists u such that $um_i \equiv -\beta_i \pmod{p}$. Putting $\beta_i' = \beta_i + um_i$, we have $\beta_i' x_i = \beta_i x_i$, and $p \mid \beta_i'$. We may thus replace β_i with β_i', and assume that

$$p \neq p_i \to p \mid \beta_i$$

for every i. We have

$$px' := p\Bigl(x - \sum_{p\mid\beta_i} \frac{\beta_i}{p} x_i\Bigr) = \sum_{p\nmid\beta_i} \beta_i x_i,$$

and $x' \notin \operatorname{rng}(\varphi)$, hence we may replace x with x'. This means that we can assume that $\beta_i = 0$ whenever $p \mid \beta_i$; putting our constraints together, we have

$$\beta_i \neq 0 \to p = p_i \wedge p \nmid \beta_i. \tag{2}$$

We need to distinguish two cases.

Case 1: $px = 0$. We will show that $\langle x, x_i \mid i < k\rangle$ is an independent sequence with exponents $\langle p, m_i \mid i < k\rangle$, contradicting the choice of k. Take $\alpha < p$, $\alpha_i < m_i$ such that $\alpha x + \sum_i \alpha_i x_i = 0$. If $\alpha = 0$, then $\vec{\alpha} = \vec{0}$ by the independence of $\vec{x}$. On the other hand, if $\alpha \neq 0$, then there exists u such that $u\alpha \equiv 1 \pmod{p}$. Then $x = u\alpha x = -\sum_i u\alpha_i x_i$, which contradicts $x \notin \operatorname{rng}(\varphi)$.

Case 2: $px \neq 0$. We can find i_0 such that $\beta_{i_0} \neq 0$, and $e_{i_0} \geq e_i$ for all i such that $\beta_i \neq 0$. In order to simplify the notation, we assume that $i_0 = 0$. As all i that $\beta_i \neq 0$ have $p = p_i$ by (2), we obtain $p^{e_0+1}x = \sum_i p^{e_0}\beta_i x_i = 0$. We claim that $\langle x, x_i \mid i > 0\rangle$ is an independent sequence with exponents $\langle p^{e_0+1}, m_i \mid i > 0\rangle$. The sequence has length k; as $p^{e_0+1} = pm_0$, the length of $\prod_i m_i$ strictly increases, hence we obtain a contradiction with the choice of $\vec{x}$ and $\vec{m}$. So, take $\alpha < p^{e_0+1}$ and $\alpha_i < m_i$ such that $\alpha x + \sum_{i\neq 0}\alpha_i x_i = 0$. Multiplying the equation by p and expanding px we get

$$\alpha\beta_0 x_0 + \sum_{i\neq 0}(\alpha\beta_i + p\alpha_i)x_i = 0.$$

By the independence of $\vec{x}$, we have $p^{e_0} \mid \alpha\beta_0$. As β_0 is coprime to p by (2), we obtain $p^{e_0} \mid \alpha$, and in particular, $p \mid \alpha$. Using the expression of px in term of $\vec{x}$ once again, we have

$$\frac{\alpha}{p}\beta_0 x_0 + \sum_{i\neq 0}\left(\frac{\alpha}{p}\beta_i + \alpha_i\right)x_i = 0.$$

By the independence of $\vec{x}$, we have $p^{e_0} \mid (\alpha/p)\beta_0$, hence $p^{e_0+1} \mid \alpha$, which implies $\alpha = 0$. Then $\vec{\alpha} = \vec{0}$ by the independence of $\vec{x}$. □ (Claim 3.13.2)

We recall that φ is an injective homomorphism, hence the two claims imply that φ is an isomorphism of the form required in the theorem.

Uniqueness: assume that

$$\varphi' \colon \bigoplus_i C({p'_i}^{e'_i}) \simeq G$$

is another $\Sigma^b_1(\alpha)$-isomorphism. Let p^e be any prime power. We have

$$\{x \in C(p_i^{e_i}) \mid p^e x = 0\} = \begin{cases} 0 & p_i \neq p, \\ C(p_i^{e_i}) & p_i = p, e_i \leq e, \\ p_i^{e_i - e}C(p_i^e) \simeq C(p_i^e) & p_i = p, e_i > e. \end{cases}$$

It follows that φ induces a $\Sigma^b_1(\alpha)$-bijection

$$\{x \in G \mid p^e x = 0\} \simeq \bigoplus_{p_i = p} C(p^{\min(e,e_i)}) \approx p^{\lambda(p^e)},$$

where $\lambda(p^e) = \sum_{p_i=p}\min(e, e_i)$. Similarly, φ' induces a bijection of the same set and $p^{\lambda'(p^e)}$, thus $\lambda(p^e) = \lambda'(p^e)$ by $iWPHP(\Sigma^b_1(\alpha))$. However, we have

$$\lambda(p^{e+1}) - \lambda(p^e) = |\{i \mid p_i = p, e_i > e\}|,$$

and similarly for λ', hence

$$|\{i \mid p_i = p, e_i = e\}| = 2\lambda(p^e) - \lambda(p^{e+1}) - \lambda(p^{e-1}) = |\{i \mid p'_i = p, e'_i = e\}|.$$

It follows that $p'_i = p_{\pi(i)}$, $e'_i = e_{\pi(i)}$ for some permutation π. □

We can vary the strength of the weak pigeonhole principle needed to prove the theorem depending on the complexity of the representation of the group. We give two examples.

Corollary 3.14 *The structure theorem 3.12 for $\Sigma^b_1(\alpha)$ finite abelian groups $\langle G,+\rangle$ such that $+$ is given by a $PV(\alpha)$-function is provable in $S^2_2(\alpha) + iWPHP(PV(\alpha))$.*

Proof: If $+$ is $PV(\alpha)$, then all instances of $iWPHP$ used in the proofs of Lemma 3.6 and Theorem 3.12 are instances of $iWPHP(PV(\alpha))$. □

Definition 3.15 (in $S^1_2(\alpha)$) A Γ-*definable finite abelian group with nonabsolute equality* is a structure $\langle G,+,\approx\rangle$, where G is a nonempty Γ-definable subset of some t, $+$ is a Γ-definable ternary relation on G, and $\approx$ is a Γ-definable equivalence relation on G, such that

$$\exists w \in G\, +(x,y,w),$$
$$x \approx x' \wedge y \approx y' \wedge +(x,y,z) \wedge +(x',y',z') \to z \approx z'$$

for all $x,x',y,y',z,z' \in G$, and appropriate versions of the axioms of abelian groups hold, e.g., commutativity is expressed as

$$+(x,y,z) \wedge +(y,x,w) \to z \approx w.$$

Example 3.16 Let $\langle G,+\rangle$ be a $\Sigma^b_1(\alpha)$ finite abelian group, and H its $\Sigma^b_1(\alpha)$ subgroup. We can represent the quotient group G/H as a $\Sigma^b_1(\alpha)$ finite abelian group with nonabsolute equality $\langle G,+,\approx\rangle$, where $x \approx y$ iff $x - y \in H$.

Corollary 3.17 *The structure theorem 3.12 for $\Sigma^b_1(\alpha)$-definable finite abelian groups with nonabsolute equality is provable in $S^2_2(\alpha) + mWPHP(\Sigma^b_1(\alpha))$.*

Proof: The proof of Theorem 3.12 works without change, except that now we need to apply the weak pigeonhole principle to multivalued functions. □

We remind the reader that $S^2_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$ and $S^2_2(\alpha) + mWPHP(\Sigma^b_1(\alpha))$ are contained in $T^2_2(\alpha)$. On the other hand, the structure theorem implies that a finite vector space over $\mathbb{F}_2$ encoded by α has a basis, and this statement is not provable in $S^2_2(\alpha)$ [116, Cor. 11.3.5]. Thus, some version of the weak pigeonhole principle is indispensable to prove the structure theorem.

The unique representation of finite abelian groups in Theorem 3.12 in terms of cyclic p-groups is known as the *primary decomposition.* There is also another unique representation of finite abelian groups as sums of cyclic groups, known as *invariant factor decomposition.* We will describe it next, we can prove it easily from Theorem 3.12.

Lemma 3.18 *(in T^0_2) Let $n = \prod_{i<k} n_i$, where n_i are pairwise coprime. Then the mapping*

$$\varphi\colon C(n) \to \bigoplus_{i<k} C(n_i)$$

defined by

$$\varphi(\alpha) = \langle \alpha \bmod n_i \mid i < k\rangle$$

is an isomorphism, and its inverse is poly-time computable.

Proof: Easy, cf. Claim 1 in the proof of [93, Ex. 1.13]. □

Theorem 3.19 *(in $S^2_2(\alpha) + iWPHP(\Sigma^b_1(\alpha))$) Let G be a $\Sigma^b_1(\alpha)$ finite abelian group. There exists a unique sequence $\langle n_i \mid i < k\rangle$ of natural numbers $n_i > 1$ satisfying $n_{i+1} \mid n_i$ for every $i < k$, such that there exists a $\Sigma^b_1(\alpha)$-definable isomorphism*

$$\varphi\colon \bigoplus_{i<k} C(n_i) \simeq G$$

of the same form as in Theorem 3.12.

Proof: Existence: consider the isomorphism

$$\bigoplus_i C(p_i^{e_i}) \simeq G$$

from Theorem 3.12. We can collect powers of the same prime together, put each collection in nonincreasing order, and pad it with trivial factors $p_i^0 = 1$ so that all collections have the same length. We obtain a representation

$$\bigoplus_{\substack{i<k\\ j<l}} C(p_j^{e_{i,j}}) \simeq G,$$

where p_j are distinct primes, and $e_{i,j} \geq e_{i+1,j}$. Put $n_i = \prod_j p_j^{e_{i,j}}$. Clearly $n_{i+1} \mid n_i$, and we have

$$\bigoplus_i C(n_i) \simeq G$$

using Lemma 3.18.

Uniqueness: let

$$\varphi'\colon \bigoplus_i C(n'_i) \simeq G$$

be another such isomorphism. We may arrange the sequences $\vec{n}$, $\vec{n'}$ to have the same length by padding the shorter one with $\vec{1}$. We denote by $o_p(n)$ the maximal $e \leq |n|$ such that $p^e \mid n$. Let p^e be any prime power. As in the proof of Theorem 3.12, we can establish

$$\{x \in G \mid p^e x = 0\} \simeq \bigoplus_i C(\gcd(p^e, n_i)) \approx p^{\sum_i \min(e, o_p(n_i))},$$

and conclude

$$|\{i \mid o_p(n_i) = e\}| = |\{i \mid o_p(n'_i) = e\}|. \tag{3}$$

We observe that the sequence $o_p(n_i)$ is nonincreasing in i, as $n_{i+1} \mid n_i$. We can thus prove $o_p(n_i) = o_p(n'_i)$ by induction on i, using (3). This implies $n_i = n'_i$. □

Remark 3.20 Theorem 3.19 has also variants for $PV(\alpha)$-groups or groups with nonabsolute equality similar to Corollaries 3.14 and 3.17.

4 Fermat's little theorem and Euler's criterion

Definition 4.1 If f is a definable function (possibly with parameters), then $PHP(f)$ states that f is not a bijection of a onto b for any $a \neq b$, i.e.,

$$a \neq b \to \exists x < a\, f(x) \geq b \vee \exists x < x' < a\, f(x) = f(x') \vee \exists y < b\, \forall x < a\, f(a) \neq b.$$

If Γ is a set of definable functions, $PHP(\Gamma)$ denotes the schema $\{PHP(f) \mid f \in \Gamma\}$.

Notice that $iWPHP(f)$ is based on a somewhat different variant of the pigeonhole principle than our $PHP(f)$, hence $PHP(PV)$ does not seem to imply $iWPHP(PV)$ over, say, S^2_2.

Theorem 4.2 *$S^2_2 + iWPHP(PV) + PHP(PV)$ proves Fermat's little theorem:*

$$x^p \equiv x \pmod{p}$$

for every prime p and integer x.

Proof: Let $G = \mathbb{F}_p^\times$ (i.e., the multiplicative group of units of the finite field $\mathbb{F}_p$ of residues modulo p). By Corollary 3.14, there exists an isomorphism

$$\varphi\colon \bigoplus_{i<k} C(p_i^{e_i}) \to G$$

defined by a PV-function (as "+" of the group, i.e., modular multiplication, is poly-time). Let $n = \prod_i p_i^{e_i}$. Clearly $nx = 0$ (i.e., $x^n = 1$ in multiplicative notation) for every $x \in G$. As φ induces a bijection of n and $p - 1$, we must have $n = p - 1$ by PHP. □

PHP is a rather strong axiom, but in this case it seems unavoidable. If $\mathbb{F}_p^\times$ is cyclic, it is easy to see that Fermat's little theorem is in $S^2_2 + iWPHP(PV)$ *equivalent* to the instance of $PHP(PV)$ used in the proof of Theorem 4.2. In the absence of PHP, we see no reason why $\mathbb{F}_p^\times$ could not be isomorphic to, say, $C(p + 1)$, in which case Fermat's little theorem fails spectacularly. In view of this discussion, we conjecture that the answer to the following problem is negative.

Question 4.3 *Is Fermat's little theorem provable in S_2?*

Fermat's little theorem can be strengthened to Euler's criterion. Recall that the *Legendre symbol* is defined by

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } p \nmid a \text{ and } a \text{ is a quadratic residue modulo } p, \\ -1 & \text{if } p \nmid a \text{ and } a \text{ is a quadratic nonresidue modulo } p, \\ 0 & \text{if } p \mid a, \end{cases}$$

for any integer a, and an odd prime p. Euler's criterion states that

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}$$

for every such a, p. We are going to characterize the relationship of Euler's criterion to Fermat's little theorem in S^1_2, and in particular, we will show that Euler's criterion is provable in $S^2_2 + iWPHP(PV) + PHP(PV)$.

Berarducci and Intrigila [29] have shown multiplicativity of the Legendre symbol

$$\left(\frac{a}{p}\right)\left(\frac{b}{p}\right) = \left(\frac{ab}{p}\right)$$

in $I\Delta_0 + iWPHP(\Delta_0)$ (their proof also works in $T^0_2 + iWPHP(PV)$, cf. [95]). We will use a different proof to get multiplicativity under a weaker assumption (cf. Lemma 3.6).

Lemma 4.4 *(in T^0_2) If p is an odd prime such that $\mathbb{F}_p^\times$ is a torsion group, then $(\cdot|p)$ is multiplicative.*

Proof: We make a few observations about multiplication in $\mathbb{F}_p^\times$:

Claim 4.4.1

(i) *For any x, there are y, z such that $x = yz$ and $y^{2^k} = 1 = z^m$ for some k and odd m. We have $(z|p) = 1$ and $(x|p) = (y|p)$.*

(ii) *If $y^{2^k} = 1$, then either $y = 1$, or $y^{2^l} = -1$ for some $l < k$.*

(iii) *If $z^{2^k} = -1$ and $y^{2^{k+1}} = 1$, there exists a such that $z^a = y$.*

(iv) *If $y^{2^k} = -1$, then $(y|p) = 1$ if and only if $\exists z\, z^{2^{k+1}} = -1$.*

Proof:

(i): We have $x^n = 1$ for some $n > 0$, and we can write $n = 2^k m$ where m is odd. Pick u, v such that $um + v2^k = 1$, and put $y = x^{um}$ and $z = x^{v2^k}$. Then $x = yz$, and $y^{2^k} = 1$, $z^m = 1$. If $m = 2r + 1$, we have $(z^{r+1})^2 = z$, thus $(z|p) = 1$. It follows that $y = w^2$ iff $x = (wz^{r+1})^2$, and symmetrically $x = w^2$ iff $y = (wz^{-(r+1)})^2$, hence $(y|p) = (x|p)$.

(ii) follows immediately from the fact that the only square roots of 1 are ± 1.

(iii): We show by reverse induction on $l \le k+1$ that $\exists a < 2^{k+1}\, y^{2^l} = z^a$. The induction step: we assume $y^{2^{l+1}} = z^a$ by the induction hypothesis. We have $(-1)^a = z^{a2^k} = y^{2^{k+l+1}} = 1$, hence a is even. Thus $y^{2^l} = \pm z^{a/2}$, which equals either $z^{a/2}$ or $z^{a/2+2^k}$. As stated, the proof used Σ^b_1-*LIND*; however, we can clearly construct a explicitly by a PV-function, hence T^0_2 suffices.

(iv): If there exists such a z, then $y = z^a$ for some a by (iii). We have $1 = y^{2^{k+1}} = z^{2^{k+1}a} = (-1)^a$, hence a is even, and $y = (z^{a/2})^2$. On the other hand, if $y = z^2$, then $z^{2^{k+1}} = -1$. □ (Claim 4.4.1)

We have $(xx'|p) = (x|p)(x'|p)$ whenever $(x|p) = 1$ or $(x'|p) = 1$, as in the proof of (i). Let thus $(x|p) = (x'|p) = -1$, we want to show $(xx'|p) = 1$. We may assume $x^{2^r} = x'^{2^r} = 1$ for some r by (i). We can fix k, k' such that $x^{2^k} = x'^{2^{k'}} = -1$ by (ii). We must have $k = k'$ by (iv). We obtain $(xx')^{2^k} = 1$, hence $xx' = 1$ or $(xx')^{2^l} = -1$ for some $l < k$ by (ii), which implies $(xx'|p) = 1$ by (iv). □

Lemma 4.5 (*in* S^1_2) *Let G be a $\Sigma^b_1(\alpha)$ finite abelian group such that $nG = 0$ for some $n > 0$, and p be a prime. If $pG = G$, then $G_p = 0$.*

Proof: Write $n = p^e m$, where $p \nmid m$. Let $x \in G$ be such that $p^k x = 0$ for some k. Using $pG = G$ and $\Sigma^b_1(\alpha)$-*LIND*, there exists $y \in G$ such that $p^e y = x$, thus $mx = ny = 0$. As $\gcd(m, p^k) = 1$, we obtain $x = 0$. □

Theorem 4.6 (*in* S^1_2) *For any odd prime p, Euler's criterion*

$$\forall a \ \left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod p$$

is equivalent to the conjunction of Fermat's little theorem

$$\forall a \ a^p \equiv a \pmod p$$

and the statement

$$\exists a \ a^{(p-1)/2} \equiv -1 \pmod p.$$

Proof: Right-to-left: if $(a|p) = 1$, there exists a b such that $b^2 = a$, thus $a^{(p-1)/2} = b^{p-1} = 1$. If $(a|p) = -1$, we choose a b such that $b^{(p-1)/2} = -1$; then $(b|p) = -1$, thus $(ab|p) = 1$ by Lemma 4.4, hence $(ab)^{(p-1)/2} = 1$, which implies $a^{(p-1)/2} = -1$.

Left-to-right: FLT is clear. If $G = \mathbb{F}_p^\times$, then $-1 \in G_2 \neq 0$, hence $G^2 \neq G$ by Lemma 4.5, i.e., there exists a square nonresidue a. By Euler's criterion, $a^{(p-1)/2} = -1$. □

Theorem 4.7 $S^2_2 + iWPHP(PV) + PHP(PV)$ *proves Euler's criterion.*

Proof: We have Fermat's little theorem by Theorem 4.2. Fix an isomorphism of $\mathbb{F}_p^\times$ and $\bigoplus_{i<k} C(p_i^{e_i})$ by Corollary 3.14, where p_i are primes. We have $p - 1 = \prod_i p_i^{e_i}$ by *PHP*. As $\mathbb{F}_p^\times$ contains only two square roots of 1, only one of the p_i is 2; assume $p_0 = 2$, and put $e = e_0$. Then $(p-1)/2^e$ is an odd integer, and as $C(2^e)$ is cyclic, there exists a $b \in \mathbb{F}_p^\times$ such that $b^{2^{e-1}} = -1$. We have $b^{(p-1)/2} = (-1)^{(p-1)/2^e} = -1$, hence we obtain Euler's criterion by Theorem 4.6. □

In connection to Fermat's little theorem, it is natural to ask

Question 4.8 *Does $S_2 + PHP(PV)$ (or a similar theory) prove that the multiplicative group of $\mathbb{F}_p$ is cyclic for every prime p?*

Consider an isomorphism

$$\varphi \colon \bigoplus_{i<k} C(p_i^{e_i}) \simeq G = \mathbb{F}_p^\times$$

as in Theorem 4.2. If $p_i \neq p_j$ for every $i \neq j$, then $G \simeq C(n)$ is cyclic by Lemma 3.18, where $n = \prod_i p_i^{e_i}$. If q is a prime, then elements of $C(n)$ satisfying $qx = 0$ form a cyclic subgroup H of order 1 (if $q \nmid n$) or q (if $q \mid n$, in which case $H = (n/q)C(q)$).

On the other hand, if $p_i = p_j$ for some $i \neq j$, we can put $q = p_i$. The element $a = q^{e_i - 1}$ generates a subgroup isomorphic to $C(q)$ in $C(q^{e_i})$, and similarly there is an element $b \in C(q^{e_j})$ generating a subgroup isomorphic to $C(q)$. Then $\{a, b\}$ generates a subgroup isomorphic to $C(q) \oplus C(q)$, all elements of which satisfy $qx = 0$. Lifting the situation to G using φ, we obtain the following dichotomy:

Lemma 4.9 *(in $S^2_2 + iWPHP(PV)$) Let p be a prime, and let G be the multiplicative group of units in $\mathbb{F}_p$.*

(i) *If G is cyclic, then for every prime q, there exists a PV-surjection of q onto the set $\{x \in G \mid x^q = 1\}$.*

(ii) *If G is not cyclic, there exists a prime q, and a PV-injection of q^2 to $\{x \in G \mid x^q = 1\}$.*

□

The usual proof of cyclicity of $\mathbb{F}_p^\times$ relies on the fact that the degree q polynomial $x^q - 1$ can have only q roots in the field $\mathbb{F}_p$; the latter is proved by induction on the degree of the polynomial. Unfortunately, the intermediate polynomials needed for the induction are not sparse, hence they are exponentially sized objects, and cannot be used in bounded arithmetic (even extended by pigeonhole principles or counting functions). On the other hand, if we could manage to match the roots against the degree using a different counting argument, there is a good chance that a weak pigeonhole principle would suffice because of the large gap given by Lemma 4.9.

Notice that the same principle can be applied to the relationship of Fermat's little theorem to Euler's criterion: assuming the former, the extra condition $\exists a\ a^{(p-1)/2} \equiv -1 \pmod{p}$ from Theorem 4.6 is equivalent to asking the degree $(p-1)/2$ polynomial $x^{(p-1)/2} - 1$ to have less than $p - 1$ roots in $\mathbb{F}_p$, hence a solution to the degree-vs-roots problem would also answer the following problem:

Question 4.10 *Does Fermat's little theorem imply Euler's criterion over S_2?*

5 Quadratic reciprocity

In this section we prove the quadratic reciprocity theorem (including the supplementary laws) from the modulo-2 counting principle *Count*$_2$ (cf. [116]). Our proof is loosely based on Gauss's third proof of reciprocity, however we have streamlined the argument so that it only uses counting modulo 2 instead of bounded sums and products, and we made sure that we can construct explicit functions witnessing the parity of the sets we want to count modulo 2.

The basic form of the modulo-2 counting principle (also called the *equipartition principle* in [29]) states that we cannot partition an odd-length interval $2a + 1 = [0, 2a + 1)$ into two-element blocks. We can weaken the principle by representing the partition in a more explicit way. We do so by requiring a function f which assigns to each element of $2a + 1$ its partner in its block. Such a function f defines a partition into blocks of size at most two if and only if f is an involution (i.e., $f \circ f = \mathrm{id}$), and the partition has no blocks of size one iff f has no fixpoint. We thus state the counting principle as "every involution on $2a + 1$ contains a fixpoint":

Definition 5.1 If f is a function (possibly with parameters), $Count_2(f)$ is the axiom

$$\exists x \leq 2a\, (f(x) > 2a \vee f(f(x)) \neq x \vee f(x) = x).$$

If Γ is a set of definable functions, we define the schema $Count_2(\Gamma) = \{Count_2(f) \mid f \in \Gamma\}$.

Notice that $\mathit{Count}_2(\Delta_0)$ is in $I\Delta_0$ equivalent to the original version of the mod-2 counting (equipartition) principle: given a Δ_0 equivalence relation with two-element blocks, we can easily define the neighbourhood function f by a Δ_0-formula. We will, however, also use the principle for PV-functions in T^0_2, and in this context our version of the principle appears to be genuinely weaker. Note also that $I\Delta_0 + \mathit{Count}_2(\Delta_0)$ is contained in the two-sorted theory $V^0[2]$.

Mod-2 counting by involutions was used to prove Fermat's theorem on sums of two squares by Heath-Brown [89] and Zagier [181]. Similar mod-4 and mod-8 counting principles were employed by Berarducci and Intrigila [29] to prove the two supplementary laws of quadratic reciprocity.

Definition 5.2 If p is an odd prime and $p \nmid a$, we put

$$\left[\frac{a}{p}\right] = \begin{cases} 0, & a \equiv \square \pmod{p}, \\ 1, & a \not\equiv \square \pmod{p}, \end{cases}$$

so that $(a|p) = (-1)^{[a|p]}$. Unless stated otherwise, all functions are assumed to be defined by PV-functions (i.e., circuits) when we work over T^0_2, and Δ_0-definable when we work over $I\Delta_0$. Residues modulo p are usually taken from $P = [-(p-1)/2, (p-1)/2]$. We also put $P^+ = [1, (p-1)/2]$, $P^- = [-(p-1)/2, -1]$, and $P^+_0 = P^+ \cup \{0\}$. We treat P and friends as sets of residues rather than integers, so that, e.g., the formula $ax \in P^+$ means $(ax \bmod p) \in [1, (p-1)/2]$. We also use x^{-1} to refer to multiplicative inverse modulo p.

We begin with a version of Gauss's Lemma.

Lemma 5.3 *(in T^0_2 or $I\Delta_0$) Let p be an odd prime, and $p \nmid a$. There exists an involution on $P^- \cup \{x \in P^+ \mid ax \in P^+\}$ with $[a|p]$ fixpoints.*

Proof: We define

$$f(x) = \begin{cases} -x, & (x, ax \in P^+ \wedge x^{-1} \in P^-) \vee (x, ax \in P^- \wedge x^{-1} \in P^+), \\ x^{-1}, & x, x^{-1} \in P^-, \\ a^{-1}x^{-1}, & ax, x^{-1} \in P^+. \end{cases}$$

It is easy to see that the three conditions define a partition of $P^- \cup \{x \in P^+ \mid ax \in P^+\}$, and f is an involution on each part. f has no fixpoints in the first part, and one ($x = -1$) in the second part. A fixpoint in the third part is an x such that x^{-1} is a positive square root of a, which is unique if it exists. In total, f has one fixpoint if $[a|p] = 1$, and two if $[a|p] = 0$. In the latter case, we modify f so that the original fixpoints are mapped to each other. $\square$

Definition 5.4 If $X \subseteq t$, and $Y \subseteq s$, we use $X \,\dot\cup\, Y$ to denote disjoint union: if X and Y are disjoint, we may take $X \,\dot\cup\, Y = X \cup Y$; in general, we put

$$X \,\dot\cup\, Y := X \cup \{t + y \mid y \in Y\} \subseteq t + s.$$

If $f\colon X \to Z$ and $g\colon Y \to Z$, then $f \,\dot\cup\, g\colon X \,\dot\cup\, Y \to Z$ is defined in the obvious way.

Lemma 5.5 (*in* T^0_2 *or* $I\Delta_0$) *If p and q are distinct odd primes, there exists an involution on $(p+3)(q+3)/4-4$ with $[p|q]+[q|p]$ fixpoints.*

Proof: Let $f(x)=\langle x,\lfloor qx/p\rfloor\rangle$. Then f is a bijection

$$f\colon\{x\in P^+\mid qx\in P^+\}\approx\{\langle x,y\rangle\in P^+_0\times Q^+_0\mid 0<qx-py<p/2\},$$

with left projection as its inverse. Symmetrically, there is an invertible bijection

$$g\colon\{y\in Q^+\mid py\in Q^+\}\approx\{\langle x,y\rangle\in P^+_0\times Q^+_0\mid -q/2<qx-py<0\}.$$

By Lemma 5.3, there exists an involution h on

$$(P^-\mathbin{\dot\cup}Q^-)\mathbin{\dot\cup}\{x\in P^+\mid qx\in P^+\}\mathbin{\dot\cup}\{y\in Q^+\mid py\in Q^+\}$$

with $[p|q]+[q|p]$ fixpoints, thus $i=(\mathrm{id}\mathbin{\dot\cup}f\mathbin{\dot\cup}g)\circ h\circ(\mathrm{id}\mathbin{\dot\cup}f\mathbin{\dot\cup}g)^{-1}$ is an involution on

$$(P^-\mathbin{\dot\cup}Q^-)\mathbin{\dot\cup}\big\{\langle x,y\rangle\in(P^+_0\times Q^+_0)\smallsetminus\{\langle 0,0\rangle\}\mid -q/2<qx-py<p/2\big\}$$

with $[p|q]+[q|p]$ fixpoints. As

$$q\left(\frac{p-1}{2}-x\right)-p\left(\frac{q-1}{2}-y\right)=\frac{p-q}{2}-(qx-py),$$

the function $j(\langle x,y\rangle)=\langle(p-1)/2-x,(q-1)/2-y\rangle$ is an involutive bijection

$$j\colon\{\langle x,y\rangle\in P^+_0\times Q^+_0\mid qx-py<-q/2\}\approx\{\langle x,y\rangle\in P^+_0\times Q^+_0\mid p/2<qx-py\}.$$

Therefore $i\mathbin{\dot\cup}j$ is an involution on

$$P^-\mathbin{\dot\cup}Q^-\mathbin{\dot\cup}\big((P^+_0\times Q^+_0)\smallsetminus\{\langle 0,0\rangle\}\big)\approx\frac{p-1}{2}+\frac{q-1}{2}+\frac{p+1}{2}\frac{q+1}{2}-1=\frac{(p+3)(q+3)}{4}-4$$

with $[p|q]+[q|p]$ fixpoints. □

Lemma 5.6 (*in* T^0_2 *or* $I\Delta_0$) *Let p be an odd prime, and $p\nmid a,b$. There exists an involution on $2(p-1)\mathbin{\dot\cup}\{x\in P^+\mid abx\in P^-\}$ with $[a|p]+[b|p]$ fixpoints.*

Proof: By Lemma 5.3, there exist involutions on

$$P^-\mathbin{\dot\cup}\{x\in P^+\mid a^{-1}x\in P^+\}$$

and

$$P^-\mathbin{\dot\cup}\{x\in P^+\mid bx\in P^+\}\approx P^+\mathbin{\dot\cup}\{x\in P^-\mid bx\in P^-\}$$

with $[a^{-1}|p]$ and $[b|p]$ fixpoints, respectively. Their union f is an involution on

$$\begin{aligned}&(P^+\cup P^-)\mathbin{\dot\cup}\{x\mid x,a^{-1}x\in P^+\vee x,bx\in P^-\}\\&\qquad=(p-1)\mathbin{\dot\cup}\{x\mid a^{-1}x\in P^+,bx\in P^-\}\mathbin{\dot\cup}\{x\mid x,a^{-1}x,bx\in P^+\vee x,a^{-1}x,bx\in P^-\}.\end{aligned}$$

The function $x \mapsto -x$ is an involutive bijection between the disjoint sets

$$\{x \in P^+ \mid a^{-1}x \in P^- \vee bx \in P^-\} \approx \{x \in P^- \mid a^{-1}x \in P^+ \vee bx \in P^+\},$$

and its union with f is thus an involution on

$$(p-1) \mathbin{\dot\cup} \{x \mid a^{-1}x \in P^+, bx \in P^-\} \mathbin{\dot\cup} (P^+ \cup P^-) = 2(p-1) \mathbin{\dot\cup} \{x \mid a^{-1}x \in P^+, bx \in P^-\}.$$

We may lift it using the function $x \mapsto a^{-1}x$, which is an invertible bijection

$$\{x \mid a^{-1}x \in P^+, bx \in P^-\} \approx \{x \in P^+ \mid abx \in P^-\},$$

to obtain an involution on

$$2(p-1) \mathbin{\dot\cup} \{x \in P^+ \mid abx \in P^-\}.$$

The number of fixpoints is $[a^{-1}|p] + [b|p] = [a|p] + [b|p]$, as obviously $(a^{-1}|p) = (a|p)$. □

Theorem 5.7 *$T^0_2 + Count_2(PV)$ and $I\Delta_0 + Count_2(\Delta_0)$ prove the law of quadratic reciprocity*

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}, \tag{4}$$

the supplementary laws

$$\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2}, \tag{5}$$

$$\left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}, \tag{6}$$

and multiplicativity of the Legendre symbol

$$\left(\frac{a}{p}\right)\left(\frac{b}{p}\right) = \left(\frac{ab}{p}\right), \tag{7}$$

where p, q are distinct odd primes, and a, b are integers.

Proof: (4) follows from Lemma 5.5, as $(p+3)(q+3)/4 - 4 \equiv (p-1)(q-1)/4 \pmod 2$.

(5) is an immediate consequence of Lemma 5.3 for $a = -1$, as $\{x \in P^+ \mid -x \in P^+\} = \varnothing$.

(6): By Lemma 5.3, there exists an involution with $[2|p]$ fixpoints on

$$P^- \cup \{x \in P^+ \mid 2x \in P^+\} \approx \frac{p-1}{2} + \left\lfloor\frac{p}{4}\right\rfloor = p - 1 - \left\lceil\frac{p-1}{4}\right\rceil,$$

thus

$$\left[\frac{2}{p}\right] \equiv \left\lceil\frac{p-1}{4}\right\rceil \equiv \left.\begin{cases} 0, & p \equiv \pm 1 \pmod 8 \\ 1, & p \equiv \pm 3 \pmod 8 \end{cases}\right\} \equiv \frac{(p^2-1)}{8} \pmod 2.$$

(7): The identity holds trivially if p divides a or b, thus assume $p \nmid a, b$. By Lemmas 5.3 and 5.6, there exists an involution on $3(p-1)$ with $[a|p] + [b|p] + [ab|p]$ fixpoints, thus $[a|p] + [b|p] \equiv [ab|p] \pmod 2$. □

We remark that the proof of Lagrange's four-square theorem in $I\Delta_0 + iWPHP(I\Delta_0)$ by Berarducci and Intrigila [29] only used multiplicativity of the Legendre symbol (apart from $I\Delta_0$). Consequently, Lagrange's four-square theorem is also provable in $I\Delta_0 + Count_2(\Delta_0)$.

Recall that the *Jacobi symbol* $(a|n)$ is defined for any integer a and an odd natural number n by

$$\left(\frac{a}{n}\right) = \prod_i \left(\frac{a}{p_i}\right),$$

where

$$n = \prod_i p_i$$

is a prime factorization of n. We can introduce it in bounded arithmetic as follows.

We assume that we have fixed an efficient sequence coding function such that

$$|w| = O\Bigl(\mathrm{lh}(w) + \sum_{i<\mathrm{lh}(w)} |(w)_i|\Bigr)$$

for any sequence w. In particular, there is an L_{PA}-term $s(n)$ such that $w \le s(n)$ for every sequence w such that $(w)_i > 1$ for every $i < \mathrm{lh}(w)$, and $n = \prod_{i<\mathrm{lh}(w)}(w)_i$. (Recall that bounded products of natural numbers are Δ_0-definable in $I\Delta_0$ by Berarducci and D'Aquino [28].) Then an easy Δ_0-induction on n shows that

$$\exists p \le s(n) \left(Seq(p) \land \forall i < \mathrm{lh}(p)\, Prime((p)_i) \land \prod_{i<\mathrm{lh}(p)} (p)_i = n \right),$$

and furthermore p is unique up to permutation of indices. Then we can define the Jacobi symbol by the Δ_0-formula

$$\left(\frac{a}{n}\right) = \varepsilon \Leftrightarrow \exists p, w \le s(n) \Biggl[Seq(p) \land Seq(w) \land \mathrm{lh}(p) = \mathrm{lh}(w) \land \forall i < \mathrm{lh}(p) \left(Prime((p)_i) \land (w)_i = \left(\frac{a}{(p)_i}\right)\right) \land n = \prod_{i<\mathrm{lh}(p)} (p)_i \land \varepsilon = \prod_{i<\mathrm{lh}(w)} (w)_i \Biggr].$$

Note that the product $\prod_{i<\mathrm{lh}(w)}(w)_i$ may involve negative integers; however, it has logarithmic length, hence it can be easily evaluated by counting the number of minus signs in w. It readily follows that $I\Delta_0$ proves the existence and uniqueness of $(a|n)$.

In the case of S^1_2, we proceed in a similar way. Prime factorization of natural numbers is provable in S^1_2 by Jeřábek [93]. Given a sequence p of primes such that $n = \prod_i (p)_i$, we can define the sequence w such that $(w)_i = (a|(p)_i)$ using Σ^b_1-comprehension, as the Legendre symbol is $\mathcal{B}(\Sigma^b_1)$-definable. Then it is easy to see that the above formula gives a provably total Σ^b_2-definition of the Jacobi symbol in S^1_2.

Theorem 5.8 *The Jacobi symbol has a provably total Σ^b_2-definition in S^1_2, and a Δ_0-definition in $I\Delta_0$. For any integers a, b, and odd positive m, n, $S^1_2 + Count_2(PV)$ and $I\Delta_0 + Count_2(\Delta_0)$*

prove

$$\begin{aligned}
a \equiv b \pmod n \to \left(\frac{a}{n}\right) &= \left(\frac{b}{n}\right),\\
\left(\frac{a}{n}\right)\left(\frac{b}{n}\right) &= \left(\frac{ab}{n}\right),\\
\left(\frac{a}{n}\right)\left(\frac{a}{m}\right) &= \left(\frac{a}{nm}\right),\\
\gcd(a,n) \neq 1 \leftrightarrow \left(\frac{a}{n}\right) &= 0,\\
\left(\frac{n}{m}\right) &= \left(\frac{m}{n}\right)(-1)^{(n-1)(m-1)/4},\\
\left(\frac{-1}{n}\right) &= (-1)^{(n-1)/2},\\
\left(\frac{2}{n}\right) &= (-1)^{(n^2-1)/8},\\
n \equiv m \pmod{4a} \to \left(\frac{a}{n}\right) &= \left(\frac{a}{m}\right).
\end{aligned}$$

Proof: We will show the reciprocity law, the other properties can be proved easily using a similar strategy. If $\gcd(n,m) \neq 1$ then $(n|m) = (m|n) = 0$, hence we may assume $\gcd(n,m) = 1$. Pick a sequence p of primes such that $n = \prod_i (p)_i$.

Assume first that $m = q$ is prime. Using Σ^b_1-comprehension (in the case of S^1_2) or Δ_0-comprehension (in the case of $I\Delta_0$), we find sequences e and w such that $(e)_i = ((p)_i|q)$, $(w)_i = (q|(p)_i)$. Using Theorem 5.7, we have

$$\left(\frac{n}{q}\right)\left(\frac{q}{n}\right) = \prod_i (e)_i \prod_i (w)_i = \prod_i (-1)^{\frac{(q-1)((p)_i-1)}{4}} = (-1)^{\frac{q-1}{2}\sum_i \frac{(p)_i-1}{2}} = (-1)^{\frac{q-1}{2}\frac{n-1}{2}},$$

as $\sum_{i<k} \frac{1}{2}((p)_i - 1) \equiv \frac{1}{2}\big(\prod_{i<k}(p)_i - 1\big) \pmod 2$ by induction on k.

In general, we fix a sequence q of primes such that $m = \prod_j (q)_j$. As above, we find a sequence w such that $(w)_j = (n|(q)_j)$. In the case of $I\Delta_0$, we find a sequence e such that $(e)_j = ((q)_j|n)$ in the same way. In the case of S^1_2, we cannot do it directly, as $((q)_j|n)$ is only Σ^b_2. However, we can use Σ^b_1-comprehension to find a sequence s of length $\operatorname{lh}(p)\operatorname{lh}(q)$ such that $(s)_{i,j} = ((q)_j|(p)_i)$, and then $(e)_j = \prod_i (s)_{i,j}$ is constructible by a PV-function from s. Then we compute

$$\left(\frac{n}{m}\right)\left(\frac{m}{n}\right) = \prod_j (w)_j \prod_j (e)_j = \prod_j (-1)^{\frac{(n-1)((q)_j-1)}{4}} = (-1)^{\frac{n-1}{2}\sum_j \frac{(q)_j-1}{2}} = (-1)^{\frac{n-1}{2}\frac{m-1}{2}}$$

as before. □

Theorem 5.9 *$S^1_2 + Count_2(PV)$ proves that the Jacobi symbol is polynomial-time computable.*

Proof: Consider a PV-function formalizing the standard algorithm for computing $(a|b)$ (see Figure 5.1). As two odd numbers are subtracted on line 12, a is even on line 5 in every but

```
   **input:** integer $a$, odd positive $b$
1  $r \leftarrow 1$
2  **if** $a < 0$ **then:**
3      $a \leftarrow -a$
4      $r \leftarrow -r$ **if** $b \equiv -1 \pmod 4$
5  **while** $a > 0$ **do:**
6      **while** $a$ is even **do:**
7          $a \leftarrow a/2$
8          $r \leftarrow -r$ **if** $b \equiv \pm 3 \pmod 8$
9      **if** $a < b$ **then:**
10         $\langle a, b\rangle \leftarrow \langle b, a\rangle$
11         $r \leftarrow -r$ **if** $a \equiv b \equiv -1 \pmod 4$
12     $a \leftarrow a - b$
13 **if** $b > 1$ **then output** $0$ **else output** $r$
```

Figure 5.1: An algorithm for the Jacobi symbol

possibly the first iteration of the outer loop, in which case the division on line 7 is executed at least once. It follows that the total number of iterations is bounded by $|a| + |b|$, and the algorithm is polynomial-time.

Let $\langle a_i, b_i, r_i \mid i \leq k\rangle$ be the sequence of values of a, b, and r during the execution of the algorithm. We find a prime factorization of $\prod_i b_i$, and use it to compute a sequence $p = \langle p_{i,j} \mid i < k, j < d(i)\rangle$ of primes such that $b_i = \prod_{j<d(i)} p_{i,j}$ for every i. Using Σ^b_1-comprehension, there is a sequence $w = \langle w_{i,j} \mid i < k, j < d(i)\rangle$ such that $w_{i,j} = (a_i|p_{i,j})$. Then we can compute the sequence $v = \langle v_i \mid i < k\rangle$ by $v_i = \prod_{j<d(i)} w_{i,j}$, so that $v_i = (a_i|b_i)$. Put $e = (a|b)$. Armed with v, we can prove $e = r_i v_i$ by induction on $i \leq k$ using Theorem 5.8, which implies that the algorithm gives the correct output. $\square$

Acknowledgements

I would like to thank Phuong Nguyen and Alan Woods for useful suggestions and comments.

The research was supported by grant IAA1019401 of GA AV ČR, grant 1M0545 of MŠMT ČR, and a grant from the John Templeton Foundation.

Chapter IV

Integer factoring and modular square roots

Abstract

Buresh-Oppenheim proved that the NP search problem to find nontrivial factors of integers of a special form belongs to Papadimitriou's class PPA, and is probabilistically reducible to a problem in PPP. In this paper, we use ideas from bounded arithmetic to extend these results to arbitrary integers. We show that general integer factoring is reducible in randomized polynomial time to a PPA problem and to the problem WeakPigeon $\in$ PPP. Both reductions can be derandomized under the assumption of the generalized Riemann hypothesis. We also show (unconditionally) that PPA contains some related problems, such as square root computation modulo n, and finding quadratic nonresidues modulo n.

1 Introduction

Integer factoring is one of the best-known problems in complexity theory which is in NP, but is not known to be polynomial-time computable. In particular, the assumed hardness of factoring has various applications in cryptography. Papadimitriou [140] introduced several classes of search problems based on parity arguments and related combinatorial principles. He showed that many natural search problems from diverse areas of mathematics belong to one of these classes, and he posed as an open problem whether the same holds for integer factoring.

The first step to answer Papadimitriou's question was taken by Buresh-Oppenheim [35]. He proved that factoring of "good" integers (odd integers n such that -1 is not a quadratic residue modulo n) such that $n \equiv 1 \ (4)$ belongs to the search class PPA, and factoring of good integers is probabilistically poly-time reducible to a PPP problem. (Note that an odd integer is good iff it has a prime divisor $p \equiv -1 \ (4)$.)

The purpose of this paper is to exhibit similar reductions for factoring of arbitrary integers. We show that factoring is probabilistically poly-time reducible to a PPA problem, as well as to WeakPigeon, which is a PPP problem. (A similar probabilistic reduction of factoring to PPP was also independently found by Buresh-Oppenheim [36].) We isolate a convenient intermediate problem, which we call FacRoot: given integers n and a such that the Jacobi symbol $(a|n) = 1$, find either a proper divisor of n, or a square root of a modulo n. It is not hard to show that factoring is probabilistically poly-time reducible to FacRoot.

The main technical ingredient of our work is to demonstrate that FacRoot $\in$ PPA. The high-level idea of the proof comes from bounded arithmetic. Chapter III introduced an arithmetical theory $S^1_2 + \mathit{Count}_2(PV)$ related to PPA, and established that this theory can prove the quadratic reciprocity theorem and other properties of the Jacobi symbol, which together imply the soundness of the usual poly-time algorithm for the Jacobi symbol. In particular, $S^1_2 + \mathit{Count}_2(PV)$ proves the totality of FacRoot, and then an application of a garden-variety witnessing theorem yields FacRoot $\in$ PPA. However, since this paper is intended for a general computational complexity audience, we include a self-contained direct proof of this result, we do not assume any prior knowledge (or posterior, for that matter) of bounded arithmetic on the part of the reader.

All probabilistic reductions in this paper can be derandomized if we assume the generalized Riemann hypothesis (*GRH*). In particular, *GRH* implies that factoring is in PPA $\cap$ PPP (and moreover, it is poly-time reducible to WeakPigeon). We also show unconditionally that several problems concerning quadratic residues have deterministic Turing reductions to FacRoot, and as such are in PPA: for one, given n and a, we can find either a square root of a modulo n, or a suitable witness that a is a quadratic nonresidue. For another, given an odd n which is not a perfect square, we can find an a such that $(a|n) = -1$ (in particular, a is a quadratic nonresidue modulo n).

The paper is organized as follows. In Section 2, we review basic concepts used in the paper to fix the notation. Section 3 presents our main results, except for the somewhat complex proof of FacRoot $\in$ PPA, which is given separately in Section 4. Some concluding remarks follow in Section 5.

2 Preliminaries

An NP *search problem* is given by a poly-time computable relation $R(x, y)$ such that $R(x, y)$ implies $\|y\| \leq \|x\|^c$ for some constant c, the problem is to find a y satisfying $R(x, y)$ given x. (We use $\|x\|$ to denote the length of x; most of our algorithms work with integers, and we reserve $|x|$ for the absolute value of x. We also warn the reader that we will often call our binary integers n, we will not use the convention that n implicitly denotes the length of the input.) For brevity, we may use R to denote the search problem itself. A search problem R is *total* if for every x there exists a y such that $R(x, y)$. Unless indicated otherwise, all search problems below will be assumed to be total NP search problems.

We will often specify NP search problems in the form "given an x such that $P(x)$, find a y satisfying $R(x, y)$", where P is a poly-time condition. In order to make it formally a total search problem, this formulation will be understood to denote the problem associated with the relation $(\neg P(x) \wedge y = 0) \vee (P(x) \wedge R(x, y))$.

A search problem R is *many-one reducible* to a search problem S, written as $R \leq_m S$, if there are poly-time functions f, g such that $S(f(x), y)$ implies $R(x, g(x, y))$. R is *Turing-reducible* to S, written as $R \leq_T S$, if there exists a poly-time oracle Turing machine M (where the oracle returns strings rather than yes/no answers) such that on input x, M computes a y solving $R(x, y)$ whenever all answers of the oracle are correct solutions of S. The class of all search problems R such that $R \leq_T S$ will be denoted FP^S. If C is a class of search problems, we write $R \leq_m C$ if $R \leq_m S$ for some $S \in C$, and similarly for $R \leq_T C$, FP^C, as well as other reduction notions mentioned below.

Let a circuit $C\colon \mathbf{2}^n \to \mathbf{2}^n$ (here, $\mathbf{2} = \{0, 1\}$) encode an undirected graph $G = \langle V, E\rangle$, where $V = \mathbf{2}^n \smallsetminus \{0^n\}$, and $\{u, v\} \in E$ iff $u, v \in V$, $u \neq v$, $C(u) = v$, and $C(v) = u$. Notice that G is a partial matching. Lonely is the following search problem: given C, find $u \in V$ unmatched by G. The class PPA (for "polynomial parity argument") consists of all search problems many-one reducible to Lonely. (This is not Papadimitriou's definition of PPA, it comes from [23], where it is shown to be equivalent to the original one.) By abuse of notation, we will also use Lonely to denote the following variant of the problem. Let $f(a, x)$, $g(a)$ be poly-time functions such that for every a, $g(a)$ is an odd natural number, and the function $f_a(x) := f(a, x)$ is an involution (i.e., $f_a(f_a(x)) = x$) on the integer interval $[0, g(a))$. Then the problem is, given a to find an $x < g(a)$ which is a fixpoint of f_a (i.e., $f_a(x) = x$). We will often use the fact that PPA is closed under Turing reductions:

Theorem 2.1 (Buss and Johnson [43]) $\mathrm{FP}^{\mathrm{PPA}} = \mathrm{PPA}$. □

The class PPP (for "polynomial pigeonhole principle") consists of problems many-one reducible to Pigeon, which is the following problem: given a circuit $C\colon \mathbf{2}^n \to \mathbf{2}^n$, find either a pair $u \neq v$ such that $C(u) = C(v)$, or a u such that $C(u) = 0^n$. If $p(n)$ is any polynomial such that $p(n) > n$ for every n, let $\textsc{WeakPigeon}_{2^n}^{2^{p(n)}}$ denote the following problem: given a circuit $C\colon \mathbf{2}^{p(n)} \to \mathbf{2}^n$, find $u \neq v$ such that $C(u) = C(v)$. We define $\textsc{WeakPigeon} := \textsc{WeakPigeon}_{2^n}^{2^{n+1}}$; the choice of $n + 1$ here does not matter:

Lemma 2.2 *For any polynomial p as above,* $\textsc{WeakPigeon} \equiv_m \textsc{WeakPigeon}_{2^n}^{2^{p(n)}}$.

Proof: Given a circuit $C(\vec{x}, u)\colon \mathbf{2}^n \times \mathbf{2} \to \mathbf{2}^n$, we put $m = p(n) - n$, and we construct a circuit $D\colon \mathbf{2}^n \times \mathbf{2}^m \to \mathbf{2}^n$ by $D(\vec{x}, u_0, \dots, u_{m-1}) = C(\cdots(C(C(\vec{x}), u_0), u_1) \dots, u_{m-1})$. Given $\langle\vec{x}, \vec{u}\rangle \neq \langle\vec{x}', \vec{u}'\rangle$ such that $D(\vec{x}, \vec{u}) = D(\vec{x}', \vec{u}')$, we find the largest $i < m$ such that $\langle\vec{y}, u_i\rangle \neq \langle\vec{y}', u_i'\rangle$, where $\vec{y}^{(\prime)} = C(\cdots(C(C(\vec{x}^{(\prime)}), u_0^{(\prime)}), u_1^{(\prime)}) \dots, u_{i-1}^{(\prime)})$. Then $C(\vec{y}, u_i) = C(\vec{y}', u_i')$. □

The class of all search problems many-one reducible to WeakPigeon does not seem to have an established name in the literature, although it clearly deserves one. In analogy with PPP, we can call it PWPP for "polynomial weak pigeonhole principle". Note that neither PPP nor PWPP is known to be closed under Turing reductions. The proof of Lemma 2.2 also implies that problems of the following kind belong to PWPP; we will denote them all as WeakPigeon by abuse of notation. Let $\varepsilon > 0$ be a constant, and f, g poly-time function such that for any a, $g(a) > 0$, and $f_a(x) := f(a, x)$ maps the interval $\bigl[0, \lceil(1 + \varepsilon)g(a)\rceil\bigr)$ into $\bigl[0, g(a)\bigr)$. Then the problem is, given a, to find $u < v < \lceil(1 + \varepsilon)g(a)\rceil$ such that $f_a(u) = f_a(v)$.

Apart from $\leq_m$ and $\leq_T$, we will also need randomized reductions. We will use several different versions to be able to state our results precisely; the definitions below are not standard, but we believe they are quite natural.

For any constant $0 < \varepsilon < 1$, we say that R is *probabilistically many-one reducible to S with error ε*, written as $R \le_m^{\mathrm{RP},\varepsilon} S$, if there is a polynomial p and poly-time functions $f(x,r)$ and $g(x,r,y)$ such that for every x,

$$\Pr_{\|r\|=p(\|x\|)}[\forall y\,[S(f(x,r),y) \Rightarrow R(x,g(x,r,y))]] \ge 1-\varepsilon.$$

We say that R is *probabilistically many-one reducible to S with controlled error*, written as $R \le_m^{\mathrm{RP}} S$, if there is a polynomial p and poly-time functions $f(x,1^k,r)$ and $g(x,1^k,r,y)$ such that for every x and k,

$$\Pr_{\|r\|=p(\|x\|,k)}[\forall y\,[S(f(x,1^k,r),y) \Rightarrow R(x,g(x,1^k,r,y))]] \ge 1-2^{-k}.$$

R is *probabilistically Turing-reducible to S*, written as $R \le_T^{\mathrm{RP}} S$, if there exists a polynomial p and a poly-time oracle Turing machine M such that

$$\Pr_{\|r\|=p(\|x\|)}[\text{every sound run of } M(x,r) \text{ solves } R(x,y)] \ge 1/2,$$

where a run is sound if all oracle answers are correct solutions of S. Note that the constant $1/2$ here is arbitrary, as we can decrease the error from any constant $\varepsilon > 0$ to any other constant (or to controlled error as above) in the usual way: we can check solutions of R, hence we can run the machine several times with independent choices of r, and return the first correct solution to the search problem. We denote by TFRP^S the class of all R such that $R \le_T^{\mathrm{RP}} S$. We observe that we can split a randomized Turing reduction as a randomized many-one reduction followed by a deterministic Turing reduction; this is particularly useful when S is from a Turing-closed class such as PPA.

Lemma 2.3 $\mathrm{TFRP}^S \le_m^{\mathrm{RP}} \mathrm{FP}^S$.

Proof: Assume that $R \le_T^{\mathrm{RP}} S$ and M^S is the Turing machine from the definition. Let T be the following search problem: given x and r, find a sound run of $M^S(x,r)$. It is easy to see that T is a total NP search problem, and $R \le_m^{\mathrm{RP}} T \le_T S$. □

Lemma 2.4 $\mathrm{TFRP}^{\mathrm{TFRP}^S} = \mathrm{TFRP}^S$.

Proof: In view of Lemma 2.3 and the obvious transitivity of $\le_m^{\mathrm{RP}}$, it suffices to show that TFRP^S is closed under deterministic Turing reductions. Let thus $T \in \mathrm{TFRP}^S$, and M^T be a poly-time oracle machine solving $R(x,y)$. Since answers of the oracle have polynomial length, the total number of sound runs of M on input x is bounded by $2^{\|x\|^c}$ for some constant c. Using the above-mentioned amplification of success rate, we can find a randomized poly-time machine N^S solving T with error $2^{-\|x\|^{c+1}}$. If we then use N to answer M's oracle queries while reusing the same pool of random bits for every call, all but a fraction of $2^{\|x\|^c}2^{-\|x\|^{c+1}} \ll 1$ of the random choices will be good for every possible run of the combined machine. □

A many-one reduction of R to S is supposed to construct a valid instance of S from whose solution it can recover a solution to the original problem. In the case of $\le_m^{\mathrm{RP}}$, the reduction algorithm succeeds in doing this only with some bounded probability. It will be also useful to consider stronger notions of reduction where we can check before consulting the oracle whether the particular choice of random bits leads to the desired result. The reduction function may abandon the computation with some bounded probability, but if it does not, then any valid solution of S gives a solution of R. Alternatively, we could repeat the computation until we find a "good" instance of S, and only then pass the query to the oracle; in this way, the reduction always succeeds, but only its expected running time is polynomial.

Formally, R is *probabilistically zero-error many-one reducible to* S, written as $R \le_m^{\mathrm{ZPP}} S$, if there is a polynomial p, poly-time functions $f(x,r)$ and $g(x,r,y)$, and a poly-time predicate $h(x,r)$, such that

(i) $\Pr_{\|r\|=p(\|x\|)}[h(x,r)] \ge 1/2$,

(ii) if $h(x,r)$ and $S(f(x,r),y)$, then $R(x,g(x,r,y))$.

Similarly, R is *probabilistically zero-error Turing-reducible to* S, written as $R \le_T^{\mathrm{ZPP}} S$, if there is a polynomial p, a poly-time predicate $h(x,r)$, and a poly-time oracle Turing machine M, such that (i), and if $h(x,r)$, then every sound run of $M^S(x,r)$ solves $R(x,y)$. Again, the constant $1/2$ is arbitrary, we can amplify the success rate from any constant $\varepsilon > 0$ to $1-2^{-k}$ (even for many-one reductions). Let TFZPP^S denote the class of all problems R such that $R \le_T^{\mathrm{ZPP}} S$. Note that if there is no oracle, TFZPP = TFRP.

Factoring is the following search problem: given a composite integer n, find a nontrivial divisor of n. We define FullFac to be the following problem: given an integer $n > 0$, find a sequence $\langle p_i : i < k\rangle$ of primes such that $n = \prod_{i<k} p_i$ (here and below, the empty product is defined to be 1). Note that Factoring and FullFac are total NP search problems as primality testing is poly-time (Agrawal, Kayal, and Saxena [1]). Clearly, Factoring $\le_m$ FullFac $\le_T$ Factoring.

We will denote the divisibility relation by $d \mid n$, modular congruences by $a \equiv b\ (n)$, and greatest common divisors by (a,b). An integer a is a *quadratic residue* modulo n if $a \equiv b^2\ (n)$ for some b. The *Legendre symbol* is defined for any integer a and an odd prime p by

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & p \mid a, \\ 1 & p \nmid a \text{ and } a \text{ is a quadratic residue mod } p, \\ -1 & p \nmid a \text{ and } a \text{ is a quadratic nonresidue mod } p. \end{cases}$$

More generally, the *Jacobi symbol* is defined for any odd $n > 0$ by

$$\left(\frac{a}{n}\right) = \prod_{i<k} \left(\frac{a}{p_i}\right),$$

where $n = \prod_{i<k} p_i$ is the prime factorization of n. We will also write $(a|n)$ instead of $\left(\frac{a}{n}\right)$ for typographical convenience. A *Dirichlet character* of modulus n is a group homomorphism $\chi\colon (\mathbb{Z}/n\mathbb{Z})^\times \to \mathbb{C}^\times$. A character is *principal* if it only assumes the value 1, and *real* if it takes values in $\{1,-1\}$. Characters can be lifted to mappings $\mathbb{Z} \to \mathbb{C}$ by putting $\chi(a) = 0$ when $(a,n) \ne 1$. Note that for any odd positive n, $\chi_n(x) = (x|n)$ is a real character of modulus n (in particular, $(a|n)(b|n) = (ab|n)$), which is principal iff n is a perfect square. The characters

```
$r \leftarrow 1$
**while** $a \neq 0$ **do:**
    **if** $a < 0$ **then:**
        $a \leftarrow -a$
        $r \leftarrow -r$ **if** $n \equiv -1 \ (4)$
    **while** $a$ is even **do:**
        $a \leftarrow a/2$
        $r \leftarrow -r$ **if** $n \equiv \pm 3 \ (8)$
    swap $a$ and $n$
    $r \leftarrow -r$ **if** $a \equiv n \equiv -1 \ (4)$
    reduce $a$ modulo $n$ so that $|a| < n/2$
**if** $n > 1$ **then output** $0$ **else output** $r$
```

Figure 2.1: An algorithm for the Jacobi symbol $(a|n)$

χ_n are called *quadratic*. The *quadratic reciprocity theorem* states that for any coprime odd $n, m > 0$,

$$\left(\frac{n}{m}\right)\left(\frac{m}{n}\right) = \begin{cases} -1 & \text{if } n \equiv m \equiv -1 \quad (4) \\ 1 & \text{otherwise.} \end{cases}$$

Together with the supplementary laws

$$\left(\frac{-1}{n}\right) = \begin{cases} 1 & n \equiv 1 \quad (4) \\ -1 & n \equiv -1 \quad (4) \end{cases} \qquad \left(\frac{2}{n}\right) = \begin{cases} 1 & n \equiv \pm 1 \quad (8) \\ -1 & n \equiv \pm 3 \quad (8) \end{cases}$$

it implies that the Jacobi symbol is poly-time computable (see Figure 2.1).

The *generalized Riemann hypothesis*[1] (*GRH*) states that for every Dirichlet character χ, all zeros of its associated L-function $L(\chi, s)$ in the critical strip $0 < \mathrm{Re}(s) < 1$ satisfy $\mathrm{Re}(s) = 1/2$. Let GRH_q denote the special case of *GRH* for quadratic characters χ. We will use the following result of Bach [18], refining the work of Ankeny [10].

Theorem 2.5 *Assume GRH_q. If χ is a nonprincipal quadratic character with modulus n, there exists $0 < a < 2(\ln n)^2$ such that $\chi(a) \neq 1$.* □

3 Search complexity of factoring

In this section, we are going to describe our main result (Theorem 3.7) on the relationship of factoring to the classes PPA and PPP (PWPP). Rather than working directly with Factoring, it will be convenient to consider other related problems.

[1] Also called the extended Riemann hypothesis (*ERH*). The nomenclature of various extensions of *RH* varies wildly in the literature. We chose to denote the *RH* for Dirichlet L-functions by *GRH* as this name seems to be more specific, whereas *ERH* is often used for other generalizations of *RH*, such as the *RH* for Dedekind ζ-functions, or L-functions of Hecke characters.

Definition 3.1 Let FacRoot denote the following problem: given an odd integer $n > 0$ and an integer a such that $(a|n) = 1$, find either a nontrivial divisor of n, or a square root of a modulo n.

We also give names to some special cases of FacRoot. FacRootMul denotes the problem, given odd $n > 0$ and integers a and b, to find a nontrivial divisor of n or a square root of one of a, b, or ab modulo n.

WeakFacRoot is the following problem: given an odd $n > 0$ and a, b such that $(a|n) = 1$ and $(b|n) = -1$, find a nontrivial divisor of n, or a square root of a modulo n.

We start with basic dependencies between these problems.

Lemma 3.2

(i) WeakFacRoot $\le_m$ FacRootMul $\le_m$ FacRoot*;*

(ii) WeakFacRoot $\le_m$ Factoring.

Proof:

(i): WeakFacRoot is a special case of FacRootMul, since $(a|n) = 1$ and $(b|n) = -1$ imply that neither b nor ab is a quadratic residue modulo n. Given an instance of FacRootMul, the multiplicativity of the Jacobi symbol implies that $(x|n) = 1$ for some $x \in \{a, b, ab\}$. We can choose such an x as the Jacobi symbol is poly-time computable, and then we pass it to FacRoot.

(ii): If n is prime, we can compute a square root of a modulo n in polynomial time using the Shanks–Tonelli algorithm. This algorithm is deterministic if we provide it with a quadratic nonresidue, which we can: b. If n is composite, we pass it to Factoring. □

Lemma 3.3

(i) FacRoot $\le_m^{\mathrm{ZPP}}$ WeakFacRoot*;*

(ii) Factoring $\le_m^{\mathrm{RP},1/2}$ FacRoot*;*

(iii) Factoring $\le_m^{\mathrm{RP},1/2}$ WeakFacRoot.

Proof:

(i): If n is a perfect square, we can return $\sqrt{n}$ as its nontrivial divisor (unless it is 1, in which case we can return 0 as the square root of a). Otherwise χ_n is a nonprincipal real character, hence with probability at least 1/2, a randomly chosen $0 < b < n$ either shares a factor with n (in which case we can return (n, b) as a nontrivial divisor) or satisfies $(b|n) = -1$, and we can pass it to WeakFacRoot.

(ii): If n is even or a perfect power, we can factor it directly, hence we may assume n is odd and it has $k \ge 2$ distinct prime divisors. We consider the following reduction. We choose a random $0 < a < n$. If $(a, n) \ne 1$, we can return it as a nontrivial divisor of n, otherwise we pass n, a to a FacRoot oracle.

Since χ_n is a nonprincipal real character, we have $(a|n) = 1$ for a half of all residues from $(\mathbb{Z}/n\mathbb{Z})^\times$. On the other hand, if $n = \prod_{i<k} p_i^{e_i}$, where the p_i are distinct primes, then a coprime

to n is a quadratic residue modulo n iff $(a|p_i) = 1$ for every $i < k$. Using the Chinese remainder theorem, a fraction 2^{-k} of $(\mathbb{Z}/n\mathbb{Z})^\times$ are quadratic residues. Thus, with probability at least $1/2 - 2^{-k} \geq 1/4$, the chosen a either shares a factor with n, or it satisfies $(a|n) = 1$ while not being a quadratic residue, hence the FACROOT oracle must give us a factor of n.

We can amplify the success probability to $1/2$ by observing that residues a such that $(a|n) = 1$ are poly-time samplable. We assume w.l.o.g. that n is not a perfect square. The reduction works as follows. We choose random $0 < a, b < n$. If $(n, a) \neq 1$ or $(n, b) \neq 1$, we can factorize n. Otherwise, we let c be the first residue from the list a, b, ab which satisfies $(c|n) = 1$, and we call FACROOT(n, c). It is easy to see that the induced distribution of c is the uniform distribution over $\{c < n : (c|n) = 1\}$, hence conditioned on $(a, n) = (b, n) = 1$, c is a quadratic nonresidue with probability $1 - 2^{1-k} \geq 1/2$.

(iii): FACROOT $\leq_m^{\mathrm{RP}}$ WEAKFACROOT by (i) and amplification of the success rate of $\leq_m^{\mathrm{ZPP}}$, hence FACTORING $\leq_m^{\mathrm{RP},1/2+\varepsilon}$ WEAKFACROOT for any $\varepsilon > 0$ by (ii). We can get rid of the ε by observing that the proof of (ii) actually shows FACTORING $\leq_m^{\mathrm{RP},1/2-1/\sqrt{n}}$ FACROOT, taking into account residues that share a factor with n. We can reduce the error of the $\leq_m^{\mathrm{ZPP}}$ reduction in (i) to $1/\sqrt{n}$, hence FACTORING $\leq_m^{\mathrm{RP},1/2}$ WEAKFACROOT. □

We remark that there is another well-known randomized reduction of factoring to square root computation modulo n due to Rabin [154], but it is suited for a different model. In the notation above, the basic idea of Rabin's reduction is that we choose a random $1 < a < n$, and if it is coprime to n, we pass n, a^2 to the FACROOT oracle. If the oracle were implemented as a (deterministic or randomized) algorithm working independently of the reduction without access to its random coin tosses, we would have a $1/2$ chance that the root b of a^2 returned by the oracle satisfies $a \not\equiv \pm b \ (n)$, allowing us to factorize n. However, this does not work in our setup. According to the definition of a search problem reduction, the reduction function must be able to cope with *any* valid answer to the oracle query—there is no implied guarantee that oracle answers are computed independently of the environment. In particular, it may happen the oracle is devious enough to always return the root $b = a$ we already know.

What we need now is to show that FACROOT or some of its variants belongs to PPA and PWPP.

Theorem 3.4 FACROOT $\in$ PPA.

We will prove Theorem 3.4 in the next section, as the argument is a bit involved.

For the pigeonhole principle, we have the following reduction, whose idea comes from the proof of the multiplicativity of the Legendre symbol in $I\Delta_0 + \mathit{WPHP}(\Delta_0)$ by Berarducci and Intrigila [29].

Theorem 3.5 FACROOTMUL $\in$ PWPP.

Proof: Assume we are given an odd $n > 1$, and integers a, b. If a or b shares a factor with n, we can return (n, a) or (n, b), resp., as a nontrivial divisor of n, we thus assume both are coprime to n. Consider the following poly-time function $f\colon \{0,1,2\} \times [1,(n-1)/2] \to [1,n-1]$:

$$f(i,x) = \begin{cases} a_i x^2 \bmod n & \text{if } (n,x) = 1, \\ x & \text{otherwise,} \end{cases}$$

where $a_0 = 1$, $a_1 = a$, $a_2 = b$. Since the domain of f is $3/2$ times larger than its range, we can use WeakPigeon to find a collision $f(i,x) = f(j,y)$, $\langle i,x\rangle \neq \langle j,y\rangle$. We may assume $(n,x) = (n,y) = 1$, as otherwise we can factor n. If $i = j$, then $x^2 \equiv y^2 \ (n)$, but $x \not\equiv \pm y \ (n)$, hence $(n, x-y)$ is a nontrivial divisor of n. If $i < j$, then $a_j a_i^{-1} \equiv (xy^{-1})^2 \ (n)$ (where the inverses are also modulo n), hence xy^{-1} is a square root of a, b, or ba^{-1} modulo n. In the last case, axy^{-1} is a square root of ab. $\square$

We mention that essentially the same reduction of Factoring to WeakPigeon by means of FacRootMul was used in a different context in [95, Thms. 4.1–2], and a similar reduction was independently discovered by Buresh-Oppenheim [36].

While we do not know whether PWPP is closed under general Turing reductions, the next lemma shows that it is closed under *nonadaptive* Turing reductions.

Lemma 3.6 *The following problem, denoted* WeakPigeon$^{\|}$*, is in* PWPP*: given a sequence $\langle C_i : i < m\rangle$ of circuits $C_i\colon \mathbf{2}^{n_i+1} \to \mathbf{2}^{n_i}$, find sequences $\langle u_i : i < m\rangle$ and $\langle v_i : i < m\rangle$ such that $u_i, v_i \in \mathbf{2}^{n_i}$, $u_i \neq v_i$, and $C_i(u_i) = C_i(v_i)$ for each $i < m$.*

Proof: Put $n = \max_i n_i$. We can pad each C_i to n output bits by considering the circuit $C_i'\colon \mathbf{2}^{n-n_i} \times \mathbf{2}^{n_i+1} \to \mathbf{2}^{n-n_i} \times \mathbf{2}^{n_i}$ defined by $C_i'(x,u) = \langle x, C_i(u)\rangle$, hence we may assume $n = n_i$ without loss of generality. By Lemma 2.2, we can amplify each C_i to a circuit $D_i\colon \mathbf{2}^{mn+1} \to \mathbf{2}^n$, and we define a circuit $D\colon \mathbf{2}^{mn+1} \to (\mathbf{2}^n)^m$ by $D(u) = \langle D_i(u) : i < m\rangle$. Using a call to WeakPigeon, we find $u \neq v$ such that $D(u) = D(v)$. Then $D_i(u) = D_i(v)$ for each i, and we can compute $u_i \neq v_i$ such that $C_i(u_i) = C_i(v_i)$. $\square$

We obtain the main result of this paper by putting everything together:

Theorem 3.7

(i) Factoring, FullFac $\le_m^{\mathrm{RP}}$ PPA*;*

(ii) Factoring $\le_m^{\mathrm{RP}}$ PWPP $\subseteq$ PPP *and* FullFac $\le_m^{\mathrm{RP}}$ $\mathrm{FP}^{\mathrm{PWPP}} \subseteq \mathrm{FP}^{\mathrm{PPP}}$.

Proof: (i): FullFac is in $\mathrm{TFRP}^{\textsc{FacRoot}}$ by Lemmas 3.3 and 2.4, hence in $\mathrm{TFRP}^{\mathrm{PPA}}$ by Theorem 3.4. This implies FullFac $\le_m^{\mathrm{RP}}$ $\mathrm{FP}^{\mathrm{PPA}} = \mathrm{PPA}$ by Lemma 2.3 and Theorem 2.1.

(ii): We have Factoring $\le_m^{\mathrm{RP},1/2}$ PWPP by Lemma 3.3 and Theorem 3.5. Given k in unary, we can reduce the error to 2^{-k} with k parallel calls to a WeakPigeon oracle, which implies Factoring $\le_m^{\mathrm{RP}}$ WeakPigeon$^{\|}$ $\in$ PWPP by Lemma 3.6. As in (i), we have FullFac $\le_T^{\mathrm{RP}}$ PWPP, hence FullFac $\le_m^{\mathrm{RP}}$ $\mathrm{FP}^{\mathrm{PWPP}}$ by Lemma 2.3. $\square$

It would be desirable to derandomize the results in Theorem 3.7. We are only able to do it under an extra assumption.

Theorem 3.8 *Assume GRH_q.*

(i) $\textsc{Factoring} \equiv_m \textsc{FacRoot} \equiv_m \textsc{WeakFacRoot} \equiv_m \textsc{FacRootMul}$*;*

(ii) $\textsc{Factoring}, \textsc{FullFac} \in \mathrm{PPA}$*;*

(iii) $\textsc{Factoring} \in \mathrm{PWPP}$*,* $\textsc{FullFac} \in \mathrm{FP}^{\mathrm{PWPP}}$*.*

Proof: It suffices to derandomize the reductions in Lemma 3.3 ((i),(ii)). For $\textsc{FacRoot} \le_m \textsc{WeakFacRoot}$, note that Theorem 2.5 guarantees that we can find a suitable $b < 2(\ln n)^2 = O(\|n\|^2)$.

For $\textsc{Factoring} \le_m \textsc{FacRoot}$, it suffices to show that for any odd n which is not a prime power, there exists an $0 < a < (\ln n)^{O(1)}$ such that either $(a, n) > 1$, or $(a|n) = 1$ and a is a quadratic nonresidue modulo n; the latter means that $(a|p) = -1$ for some prime $p \mid n$.

We can assume that $(a, n) = 1$ for every $0 < a < 2(\ln n)^2$, otherwise we are done. Let p be a prime divisor of n such that, if possible, the exponent of p in the prime factorization of n is even, so that n/p is not a perfect square. Then $\chi_{n/p}$ is a nonprincipal quadratic character, and there is $0 < u < 2(\ln(n/p))^2$ such that $(u|n/p) = -1$ by Theorem 2.5. This implies $(u|n) = -(u|p)$. If $(u|n) = 1$, we can take $a = u$. Otherwise, we have $(u|n) = -1$ and $(u|p) = 1$. Since χ_p is also a nonprincipal quadratic character, there is $0 < v < 2(\ln p)^2$ such that $(v|p) = -1$. If $(v|n) = 1$, we can take $a = v$, otherwise we take $a = uv$. Either way, $a < 4(\ln p)^2(\ln(n/p))^2 < \frac{1}{4}(\ln n)^4$. $\square$

We can use $\textsc{FacRoot}$ with constant a to obtain special cases of factoring that are unconditionally in deterministic PPA, see Example 4.6. In fact, we can factor n as long as there exists a quadratic nonresidue $a = (\log n)^{O(1)}$ such that $(a|n) = 1$. We can express this more perspicuously as follows.

Definition 3.9 Let $s > 0$. An integer n is *s-strongly composite,* if we can write $n = n_0 n_1$ so that neither n_0 nor n_1 is a quadratic residue modulo s.

Notice that an odd integer is 4good in the sense of [35] iff it is 4-strongly composite.

Theorem 3.10 *For any constant c, the following problem is in* PPA*: given an $n > 0$ which is s-strongly composite for $s = \lfloor (\log n)^c \rfloor !$, find a nontrivial divisor of n.*

Proof: We can assume w.l.o.g. that n is coprime to $\lfloor (\log n)^c \rfloor !$ (hence odd). It suffices to show that there exists an a with $|a| \le (\log n)^{2c}$ such that $(a|n_0) = (a|n_1) = -1$. Since n_i is a quadratic nonresidue modulo s, it is also a quadratic nonresidue modulo s_i, where $s_i = 8$, or s_i is an odd prime divisor of s, i.e., $s_i \le (\log n)^c$.

Assume first that both n_0, n_1 are quadratic nonresidues modulo s_0. If s_0 is odd, we put $a = s_0^* := (-1)^{(s_0-1)/2} s_0$. Then $(a|n_i) = (n_i|s_0) = -1$ by quadratic reciprocity. If $s_0 = 8$, i.e., $n_0, n_1 \not\equiv 1 \pmod{8}$, we choose $m \in \{3, 5, 7\}$ such that $m \not\equiv n_0, n_1 \pmod{8}$, and we put

$$a = \begin{cases} -2 & m = 3, \\ -1 & m = 5, \\ 2 & m = 7. \end{cases}$$

Then $(a|n_0) = (a|n_1) = -1$.

If both n_0, n_1 are quadratic nonresidues modulo s_1, we proceed similarly.

Assume that n_i is a quadratic residue modulo s_{1-i} for $i = 0, 1$. Put

$$a_i = \begin{cases} s_i^* & s_i \text{ is odd,} \\ -1 & s_i = 8, n_i \equiv 3, 7 \quad (8), \\ 2 & s_i = 8, n_i \equiv 5 \quad (8), \end{cases}$$

and $a = a_0 a_1$. Then $(a_i|n_i) = -1$ and $(a_{1-i}|n_i) = 1$, hence $(a|n_i) = -1$. □

Conversely, one can show that if a is a quadratic nonresidue such that $(a|n) = 1$, then n is s-strongly composite for any s divisible by $4a$.

In Theorem 3.10, we do not need s to have the exact form given there: it is only essential that the prime factorization of s is known.

It is not clear whether one can fully unconditionally derandomize Theorem 3.7. While no deterministic polynomial-time algorithm to find quadratic nonresidues is known without *GRH*, in PPA we can do better:

Lemma 3.11 *The following problem is in* $\mathrm{FP}^{\textsc{FacRoot}} \subseteq \mathrm{PPA}$*: given an odd* $n > 1$*, find an* a *such that* $(a|n) = -1$*, or a nontrivial divisor of* n*.*

Proof: Consider the following algorithm. Put $a = -1$. While $(a|n) = 1$, repeat the following steps: call the FacRoot oracle; if it provides a factor of n, we are done, otherwise we replace a with its square root modulo n.

The algorithm must halt within $\log_2 n$ iterations: if a is a 2^kth root of -1, its order in $(\mathbb{Z}/n\mathbb{Z})^\times$ is $2^{k+1} < n$. □

Notice that, conversely, FacRoot is Turing-reducible to WeakFacRoot together with the problem from Lemma 3.11.

In fact, FacRoot does the dual job of factoring and computing square roots. In Theorem 3.7 we have exploited its factoring capacity by supplying it with quadratic nonresidues, but we can also use it the other way round to obtain algorithms for finding square roots and quadratic nonresidues modulo arbitrary integers. We start with the latter.

Theorem 3.12 *The following problem is in* $\mathrm{FP}^{\textsc{FacRoot}} \subseteq \mathrm{PPA}$*: given an odd* n *which is not a perfect square, find an* a *such that* $(a|n) = -1$*.*

Proof: The algorithm maintains a sequence $\langle n_i : i < k \rangle$ of integers $n_i > 1$ such that $n = \prod_{i<k} n_i$, and a sequence $\langle a_i : i < k \rangle$, where some of the a_i may be undefined, but if a_i is defined, then $(a_i|n_i) = -1$. We initialize it with $k = 1$, $n_0 = n$, a_0 undefined, and we repeat in arbitrary order the following steps until neither is applicable any more:

- If $n_i \neq n_j$ are such that $(n_i, n_j) > 1$, we delete n_i, n_j from the sequence and replace them with (n_i, n_j) (two copies), $n_i/(n_i, n_j)$, and $n_j/(n_i, n_j)$, omitting those equal to 1 (this can happen only for one of the four numbers, hence the length of the sequence always increases). The a_i entries corresponding to the new numbers are undefined.

- If a_i is undefined, we call as an oracle the search problem from Lemma 3.11 on n_i. If it returns a nontrivial divisor of n_i, we expand the n_j sequence as in the previous step. Otherwise, it provides a value for a_i.

Since $k \le \log n$, the algorithm must halt in $O(\|n\|)$ steps. When it does, all a_i are defined, and the n_i entries are pairwise equal or coprime, hence we can write $n = \prod_{i\in I} n_i^{e_i}$ for some $I \subseteq \{0, \dots, k-1\}$ and $e_i > 0$, where n_i, $i \in I$, are pairwise coprime. Since n is not a perfect square, we can pick $i \in I$ such that e_i is odd. By the Chinese remainder theorem, we can compute an a such that $a \equiv a_i \ (n_i^{e_i})$ and $a \equiv 1 \ (n/n_i^{e_i})$. Then

$$\left(\frac{a}{n}\right) = \prod_{j\in I} \left(\frac{a}{n_j}\right)^{e_j} = (-1)^{e_i} = -1. \qquad \square$$

Corollary 3.13 *The following problem is in* $\mathrm{FP}^{\text{FacRoot}} \subseteq \mathrm{PPA}$*: given $n > 2$, find an a coprime to n which is a quadratic nonresidue modulo n.*

Proof: If n is a power of 2, we can return 3. Otherwise, we can write $n = 2^e m^{2^k}$, where m is odd and not a perfect square. By Theorem 3.12, we can find a such that $(a|m) = -1$. By adding m to a if necessary, we can make sure a is odd, hence $(n, a) = 1$. Since a is a quadratic nonresidue modulo $m \mid n$, it is also a nonresidue modulo n. $\square$

Another problem we are going to reduce to FacRoot is the computation of square roots modulo n. A priori it is not clear how to formulate it as a total NP search problem, as the quadratic residuosity problem is neither known nor assumed to be poly-time decidable. We can remedy this by requiring the search problem to find something sensible also for quadratic nonresidues.

Definition 3.14 Let n be a positive integer. If $(a, n) = 1$, a divisor $m \mid n$ is a *coprime nonsquare witness for a modulo n* if

- m is odd and $\left(\frac{a}{m}\right) = -1$, or
- $m = 4$ and $a \equiv 3 \ (4)$, or
- $m = 8$ and $a \equiv 5 \ (8)$.

If a is an arbitrary integer, an m is a *nonsquare witness for a modulo n*, if m is not a perfect square, m is odd or 2, and there are e, b, and $j < e$ such that $m^e \mid n$, $a = m^j b$, $(m, b) = 1$, and if j is even, m (if odd) or 4 or 8 (if $m = 2$) is a coprime nonsquare witness for b modulo m^{e-j}.

It is easy to see that the property of being a nonsquare witness is poly-time decidable.

Let Root denote the following search problem: given $n > 0$ and a, find either a square root of a modulo n, or a nonsquare witness for a modulo n.

Lemma 3.15 *If there exists a nonsquare witness for a modulo n, then a is a quadratic non-residue modulo n.*

Proof: If m is a coprime nonsquare witness for a, then a is a quadratic nonresidue modulo m, and a fortiori modulo n.

Let m be a nonsquare witness for a, and let e, b, and j be as in Definition 3.14. Assume for contradiction $a \equiv (uc)^2\ (n)$, where $(m,c) = 1$, and $u \mid m^k$ for some k. We have $m^j \mid (uc)^2$, hence $m^j \mid u^2$. Moreover, if we write $u^2 = m^j v$, then $b \equiv vc^2\ (m^{e-j})$, hence $(m,v) = 1$, i.e., $v = 1$ and $m^j = u^2$. Since m is not a perfect square, this implies j is even. However, $b \equiv c^2$ (m^{e-j}) contradicts the fact that b has a coprime nonsquare witness modulo m^{e-j}. □

Notice that Root is a generalization of FacRoot: a nonsquare witness for a modulo n is a nontrivial divisor of n, unless n is odd and $(a|n) = -1$.

Theorem 3.16 $\textsc{Root} \in \mathrm{FP}^{\textsc{FacRoot}} \subseteq \mathrm{PPA}$.

Proof: Write $n = 2^e m$ with m odd. In the first stage of our algorithm, we keep a sequence $\langle n_i : i < k\rangle$ of integers $n_i > 1$ such that $m = \prod_{i<k} n_i$, and a sequence of integers $\langle u_i : i < k\rangle$ where some u_i may be undefined. We maintain the property that whenever u_i is defined, we can write $a = n_i^{j_i} a_i$ for some j_i so that $(a_i, n_i) = 1$, and we have $a_i \equiv u_i^2\ (n_i)$. We start with $k = 1$, $n_0 = m$ and u_0 undefined, and we repeat the following steps until none of them are applicable any more:

- If $n_i \neq n_j$ are such that $(n_i, n_j) > 1$, we delete n_i, n_j from the sequence and replace them with two copies of (n_i, n_j), $n_i/(n_i, n_j)$, and $n_j/(n_i, n_j)$ as in the proof of Theorem 3.12.
- If n_i is a perfect square, we replace n_i with two copies of $\sqrt{n_i}$.
- If $a = n_i^{j_i} a_i$ where $n_i \nmid a_i$, but $(n_i, a_i) > 1$, we replace n_i with (n_i, a_i) and $n_i/(n_i, a_i)$.
- If $a = n_i^{j_i} a_i$ where $(a_i|n_i) = 1$, but u_i is undefined, we call a FacRoot oracle on n_i, a_i. If it returns a nontrivial divisor $d \mid n_i$, we replace n_i with d and n_i/d. Otherwise, it returns a square root of a_i modulo n_i, which we store as u_i.

This stage terminates after $O(\|n\|)$ steps. When it does, we can write $m = \prod_{i\in I} n_i^{e_i}$ for some $e_i > 0$, $I \subseteq \{0, \dots, k-1\}$, where n_i, $i \in I$, are pairwise coprime, none of them is a perfect square, and we have $a = n_i^{j_i} a_i$ for some j_i and $(a_i, n_i) = 1$. For each i, we try to compute a square root z_i of a modulo $n_i^{e_i}$ as follows:

- If $j_i \geq e_i$, we put $z_i = 0$.
- If $j_i < e_i$, and j_i is odd or $(a_i|n_i) = -1$, we return n_i as a nonsquare witness for a.
- If $j_i < e_i$ is even and $(a_i|n_i) = 1$, then u_i is defined, and $u_i^2 \equiv a_i\ (n_i)$. We put $z_i = n_i^{j_i/2} v_i$, where $v_i^2 \equiv a_i\ (n_i^{e_i})$ is computed using Hensel's lifting, which is an iteration of the following procedure: if we have u such that $u^2 \equiv a_i\ (n_i^c)$, we compute $w \equiv (2u)^{-1}\ (n_i^c)$, and we put $u' = (u^2 + a_i)w$. Then $u'^2 \equiv a_i\ (n_i^{2c})$.

We also try to find a square root z of a modulo 2^e. We write $a = 2^j b$ with b odd, and then:

- If $j \geq e$, we put $z = 0$.

- If $j < e$, we return 2 as a nonsquare witness for a whenever one of the following cases happens: j is odd, or $e - j \geq 2$ and $b \equiv 3 \ (4)$, or $e - j \geq 3$ and $b \equiv 5 \ (8)$.
- Otherwise, $j < e$ is even, and $1^2 \equiv b \ (2^{\min\{e-j,3\}})$. We put $z = 2^{j/2}v$, where $v^2 \equiv b \ (2^{e-j})$; if $e - j > 3$, we compute v using the following variant of Hensel's lifting. If we have u such that $u^2 \equiv b \ (2^c)$, we compute $w \equiv u^{-1} \ (2^{c-2})$, and we put $u' = ((u^2 + b)/2)w$. Then $u'^2 \equiv b \ (2^{2c-2})$.

Finally, using the Chinese remainder theorem, we compute x such that $x \equiv z \ (2^e)$ and $x \equiv z_i$ $(n_i^{e_i})$ for every i, then $x^2 \equiv a \ (n)$. □

4 FacRoot is in PPA

The purpose of this section is to prove Theorem 3.4. As already mentioned in the introduction, the original idea of the proof comes from previous work of the author on the provability of the quadratic reciprocity theorem in variants of bounded arithmetic, and in fact, FacRoot $\in$ PPA is a simple corollary of these results. This connection is described in detail in Section 4.1. In order to make this paper more self-contained, we give a direct combinatorial proof of Theorem 3.4 in Section 4.2. Readers uncomfortable with bounded arithmetic may safely skip straight there.

4.1 Bounded arithmetic

We assume familiarity with basic facts about subsystems of bounded arithmetic, in particular Buss's theory S^1_2. We refer the reader to [40, 116] for more background.

Chapter III introduced a theory $S^1_2 + \mathit{Count}_2(PV)$, axiomatized over S^1_2 by the following principle: for every number a and circuit C, C does not define an involution on $\{0, \dots, 2a\}$ without fixpoints. Notice that the axiom is Σ^b_1, and the corresponding search problem is a minor variant of Lonely.

Lemma 4.1 *If $S^1_2 + \mathit{Count}_2(PV) \vdash \forall x \, \exists y \, \varphi(x, y)$, where $\varphi \in \Sigma^b_1$, then the search problem to find a y satisfying $\varphi(x, y)$ given x is in* PPA.

Proof: By the assumption, S^1_2 proves

$$\exists a, C \, \forall u \leq 2a \, (C(C(u)) = u \neq C(u) \leq 2a) \vee \exists y \, \varphi(x, y),$$

hence $S^1_2(h)$ proves its Herbrandization

$$\exists a, C \, (h(a, C) \leq 2a \to C(C(h(a, C))) = h(a, C) \neq C(h(a, C)) \leq 2a) \vee \exists y \, \varphi(x, y).$$

This is an $\exists\Sigma^b_1(h)$ formula, hence using Parikh's theorem and Buss's witnessing theorem, there exists a polynomial-time oracle function f^h such that

$$(1) \qquad \exists a, C \, (h(a, C) \leq 2a \to C(C(h(a, C))) = h(a, C) \neq C(h(a, C)) \leq 2a) \vee \varphi(x, f^h(x))$$

holds in $\mathbb{N}$ for any choice of h. Let us run f on an input x with an oracle solving the PPA-problem corresponding to Count_2 in place of h, and let y be its output. We may assume that

f never asks the same question more than once, hence the oracle answers in any particular run can be extended to a function h which satisfies

$$h(a,C) \le 2a \wedge (C(C(h(a,C))) \ne h(a,C) \vee h(a,C) = C(h(a,C)) \vee C(h(a,C)) > 2a).$$

Then (1) implies $\varphi(x,y)$. Thus, the search problem associated to φ is in $\mathrm{FP}^{\mathrm{PPA}} = \mathrm{PPA}$ using Theorem 2.1. □

Let $J(a,n)$ denote a PV-function formalizing the algorithm in Figure 2.1. As shown in Theorem III.5.9, $S^1_2 + \mathit{Count}_2(PV)$ proves that $J(a,n)$ agrees with the definition of the Jacobi symbol in terms of factorization of n and quadratic residues. In particular, the theory proves that for prime n, $J(a,n) = 1$ implies that a is a quadratic residue, which can be expressed as the following Σ^b_1 formula:

Theorem 4.2 *$S^1_2 + \mathit{Count}_2(PV)$ proves*

$$J(a,n) = 1 \to \exists x\,(x^2 \equiv a \pmod{n}) \vee \exists u,v < n\,(uv = n). \qquad \square$$

Theorem 3.4 readily follows.

4.2 Explicit algorithm

Before turning to FacRoot proper, we will describe PPA algorithms for some of its special cases which we will need as ingredients in the main construction.

We introduce some notation for conciseness. If n is a fixed odd integer $n > 1$, we consider

$$N = \{x : |x| < n/2, (n,x) = 1\}$$

as a set of unique representatives of $(\mathbb{Z}/n\mathbb{Z})^\times$. We also write $N^+ = \{x \in N : x > 0\}$, $N^- = \{x \in N : x < 0\}$, $N_0 = N \cup \{0\}$, and similarly for N^+_0, N^-_0. We assume operations on residues are computed modulo n with a result in N, so that, e.g., $ab^{-1} \in N^+$ means that $a \equiv bx \ (n)$ for some $x \in N^+$.

Lemma 4.3 *There is a poly-time function $f(n,a,x)$ such that for any odd $n > 1$ and an integer a coprime to n, the function $f_{n,a}(x) = f(n,a,x)$ defines an involution on*

$$\{x \in N^- : ax \in N^-\} \cup N^+_0$$

whose fixpoints are of the form x^{-1}, where

(i) *$x \in N^+ \smallsetminus \{1\}$ and $x^2 = 1$, or*

(ii) *$x \in N^-$ and $x^2 = a$.*

Proof: We define $f'_{n,a}$ on $\{x \in N^- : ax \in N^-\} \cup N^+$ by

$$f'_{n,a}(x) = \begin{cases} x^{-1} & x, x^{-1} \in N^+, \\ a^{-1}x^{-1} & ax, x^{-1} \in N^-, \\ -x & (x, ax \in N^+ \wedge x^{-1} \in N^-) \vee (x, ax \in N^- \wedge x^{-1} \in N^+). \end{cases}$$

It is easy to see that the three conditions define a partition of $\{x \in N^- : ax \in N^-\} \cup N^+$, and $f'_{n,a}$ is an involution on each part. The fixpoints of $f'_{n,a}$ in the first two parts have the forms (i) (without the restriction $x \neq 1$) and (ii), respectively, and there are no fixpoints in the third part. Finally, we put

$$f_{n,a}(x) = \begin{cases} 1 & x = 0, \\ 0 & x = 1, \\ f'_{n,a}(x) & x \neq 0, 1. \end{cases} \qquad \square$$

Definition 4.4 For a constant a, let $\textsc{FacRoot}_a$ denote the following special case of $\textsc{FacRoot}$: given an odd positive n such that $(a|n) = 1$, find either a nontrivial divisor of n, or a square root of a modulo n.

Lemma 4.5 $\textsc{FacRoot}_{-1}$ *and* $\textsc{FacRoot}_2$ *are in* PPA.

Proof: Given $n \equiv \pm 1 \;(8)$, observe that

$$\{x \in N^- : 2x \in N^-\} = N \cap [-(n - 2 \pm 1)/4, -1].$$

We define an involution r on $[-(n - 2 \pm 1)/4, (n - 1)/2]$ by

$$r(x) = \begin{cases} x & x \neq 0, (x, n) \neq 1, \\ f_{n,2}(x) & \text{otherwise.} \end{cases}$$

The domain of r is an interval of size $(3n \pm 1)/4$, which is odd, hence we can use $\textsc{Lonely}$ to find a fixpoint x of r. Using Lemma 4.3, we see that either x^{-1} is a square root of 2, or it is a square root of 1 distinct from ± 1, or $(x, n) \neq 1$. In the last two cases, we can factorize n.

For $\textsc{FacRoot}_{-1}$, we define similarly an involution on $[0, (n - 1)/2]$ using $f_{n,-1}$. $\square$

Using a similar construction, it is possible to show $\textsc{FacRoot}_a \in$ PPA for every constant a. We skip the details, as we will not directly need this fact, and Theorem 3.4 is more general. However, notice that $\textsc{FacRoot}_{-1} \in$ PPA restates Buresh-Oppenheim's original result, and any constant a yields a similar special case of factoring:

Example 4.6 The following search problems are in PPA.

Given $n \equiv \pm 1 \;(8)$ such that 2 is a quadratic nonresidue modulo n (i.e., n has a divisor $p \equiv \pm 3 \;(8)$), find a nontrivial divisor of n.

Given $n \equiv 1 \;(3)$ such that -3 is a quadratic nonresidue modulo n (i.e., n has a divisor $p \equiv 2 \;(3)$), find a nontrivial divisor of n.

Lemma 4.7 $\textsc{FacRootMul}$ (*and thus* $\textsc{WeakFacRoot}$) *is in* PPA.

Proof: Let $n > 1$ be odd, and a, b coprime to n. Define

$$g(x) = \begin{cases} \langle 0, -x \rangle & x \in N_0^+, \\ \langle 1, -x \rangle & x, ax, b^{-1}x \in N^-, \\ \langle 2, -x \rangle & x, ax \in N^-, b^{-1}x \in N^+. \end{cases}$$

Then g is a poly-time bijection from $\{x \in N^- : ax \in N^-\} \cup N_0^+$ onto

$$\begin{aligned} A = (\{0\} \times N_0^-) &\cup (\{1\} \times \{x : x, ax, b^{-1}x \in N^+\}) \\ &\cup (\{2\} \times \{x \in N^+ : ax \in N^+, b^{-1}x \in N^-\}) \end{aligned}$$

with a poly-time inverse. Similarly,

$$h(x) = \begin{cases} \langle 0, x\rangle & x \in N^+, \\ \langle 1, x\rangle & x = 0 \text{ or } x, b^{-1}x, ax \in N^-, \\ \langle 2, x\rangle & x, b^{-1}x \in N^-, ax \in N^+ \end{cases}$$

is a bijection from $\{x \in N^- : b^{-1}x \in N^-\} \cup N_0^+$ onto

$$\begin{aligned} B = (\{0\} \times N^+) &\cup (\{1\} \times (\{0\} \cup \{x : x, ax, b^{-1}x \in N^-\})) \\ &\cup (\{2\} \times \{x \in N^- : ax \in N^+, b^{-1}x \in N^-\}), \end{aligned}$$

$x \mapsto \langle 2, bx\rangle$ is a bijection from $\{x \in N^- : abx \in N^-\} \cup N_0^+$ onto

$$C = \{2\} \times (\{0\} \cup \{x : ax \in N^- \vee b^{-1}x \in N^+\}),$$

and $\langle 1, x\rangle \mapsto \langle 1, -x\rangle$ is a fixpoint-free involution on

$$D = \{1\} \times \{x \in N : x, ax, b^{-1}x \text{ do not have the same sign}\}.$$

We can thus define a poly-time involution r on $\{0, 1, 2\} \times [-(n-1)/2, (n-1)/2]$ by

$$r(e, x) = \begin{cases} g(f_{n,a}(g^{-1}(e, x))) & \langle e, x\rangle \in A, \\ h(f_{n,b^{-1}}(h^{-1}(e, x))) & \langle e, x\rangle \in B, \\ \langle 2, bf_{n,ab}(b^{-1}x)\rangle & \langle e, x\rangle \in C, \\ \langle 1, -x\rangle & \langle e, x\rangle \in D, \\ \langle e, x\rangle & x \neq 0, (x, n) > 1. \end{cases}$$

Since $3n$ is odd, we can use LONELY to find a fixpoint $\langle e, x\rangle$ of r. We cannot have $\langle e, x\rangle \in D$. If $x \neq 0$, $(x, n) > 1$, we can factor n. If $\langle e, x\rangle \in A$, then $y := g^{-1}(e, x)$ is a fixpoint of $f_{n,a}$. Thus, either $y^2 = 1$, $y \neq \pm 1$, in which case we can factor n, or y^{-1} is a square root of a. Similarly, if $\langle e, x\rangle \in B \cup C$, we can factor n, or compute a square root of b or ab. □

Lemma 4.7 is enough to prove our main result, Theorem 3.7. However, we will proceed with the proof of Theorem 3.4, as we are interested in the possibility of unconditional derandomization of the reduction of factoring to PPA, and placing FACROOT in PPA can be seen as a partial step towards that goal. Moreover, randomized versions of Theorems 3.12 and 3.16 would not be interesting.

Lemma 4.8 *The following problems are in* PPA.

(i) FacRootOdd*: given an odd $n > 0$, a sequence $\langle a_i : i < k\rangle$ of integers coprime to n such that k is odd, and a square root x of $\prod_{i<k} a_i$ modulo n, find a nontrivial divisor of n, or a square root of some a_i modulo n.*

(ii) FacRootEven*: given an odd $n > 0$, and a sequence $\langle a_i : i < k\rangle$ of integers coprime to n such that k is even, find a nontrivial divisor of n, or a square root of $\prod_{i<k} a_i$ or of some a_i modulo n.*

Proof: (i): Put $I = \{0, \dots, k-1\}$ and $y = 1$, and repeat the following steps. If $I = \{i\}$, return xy^{-1} as a square root of a_i. If $|I| > 1$, pick $i, j \in I$, $i \neq j$, and call FacRootMul on n, a_i, a_j. If it gives us a nontrivial divisor of n, or a square root of a_i or a_j, we return it. Otherwise, it provides a square root z of $a_i a_j$. We multiply y by z, remove i, j from I, and repeat the loop.

(ii): Put $x = a_k = \prod_{i<k} a_i$, and call FacRootOdd. □

Definition 4.9 QuadRec is the following problem: given odd coprime $n, m > 0$ such that $n \equiv 1 \ (4)$, and a square root a of n modulo m, find a nontrivial divisor of n or m, or a square root of m modulo n.

Notice that QuadRec is a special case of FacRoot: the input data ensure $(n|m) = 1$, hence $(m|n) = 1$ by quadratic reciprocity.

Lemma 4.10 QuadRec $\in$ PPA.

Proof: We may assume $n, m > 1$ and $a \in M^-$, so that $b = a^{-1} \in M$ is a fixpoint of $f_{m,n}$. Put $n_2 = (n+1)/2$, $m_2 = (m+1)/2$. The function

$$g(x) = \begin{cases} \langle x, m_2\rangle & x \in N_0^+, \\ \langle -x, \lfloor -mx/n\rfloor\rangle & x, mx \in N^- \end{cases}$$

is a poly-time bijection with poly-time inverse from $\{x \in N^- : mx \in N^-\} \cup N_0^+$ onto

$$A = \bigl(N_0^+ \times \{m_2\}\bigr) \cup \bigl\{\langle x, y\rangle \in N_0^+ \times [0, m_2) : mx - ny \in N^+\bigr\},$$

where $mx - ny$ is *not* evaluated modulo n, but literally. Likewise,

$$h(y) = \begin{cases} \langle n_2, y\rangle & y \in M_0^+, \\ \langle \lfloor -ny/m\rfloor, -y\rangle & y, ny \in M^- \end{cases}$$

is a bijection from $\{y \in M^- : ny \in M^-\} \cup M_0^+$ onto

$$B = \bigl(\{n_2\} \times M_0^+\bigr) \cup \bigl\{\langle x, y\rangle \in [0, n_2) \times M_0^+ : mx - ny \in M^-\bigr\}.$$

The function $k(x, y) = \langle n_2 - 1 - x, m_2 - 1 - y\rangle$ is a poly-time involution with no fixpoints on

$$C = \bigl\{\langle x, y\rangle \in [0, n_2) \times [0, m_2) : mx - ny \geq n_2 \text{ or } mx - ny \leq -m_2\bigr\}.$$

We define a poly-time involution r on $([0,n_2]\times[0,m_2])\smallsetminus\{\langle n_2,m_2\rangle\}$ by

$$r(x,y)=\begin{cases} g(f_{n,m}(g^{-1}(x,y))) & \langle x,y\rangle\in A,\\ h(f_{m,n}(h^{-1}(x,y))) & \langle x,y\rangle\in B\smallsetminus\{h(b)\},\\ \langle 0,0\rangle & \langle x,y\rangle=h(b),\\ h(b) & x=y=0,\\ k(x,y) & \langle x,y\rangle\in C,\\ \langle x,y\rangle & \text{otherwise.}\end{cases}$$

Notice that if $x\in[0,n_2)$ and $y\in[0,m_2)$ are such that $mx-ny=0$, then $x=y=0$, as $(n,m)=1$. It follows that the last clause in the definition of r applies to elements of the set

$$\begin{aligned} D=\bigl(([1,n_2)\smallsetminus N^+)\times\{m_2\}\bigr)\cup\bigl(\{n_2\}\times([1,m_2)\smallsetminus M^+)\bigr)\\ \cup\bigl\{\langle x,y\rangle\in[0,n_2)\times[0,m_2): mx-ny\in([1,n_2)\smallsetminus N^+)\cup((-m_2,-1]\smallsetminus M^-)\bigr\}.\end{aligned}$$

The domain of r has odd size $(n_2+1)(m_2+1)-1$, hence using Lonely, we can find a fixpoint $\langle x,y\rangle$ of r. If $\langle x,y\rangle\in A$, it gives us a square root of m modulo n, or a square root of 1 distinct from ±1, in which case we can factorize n. If $\langle x,y\rangle\in B$, we get a square root of n modulo m distinct from $\pm a$, or a square root of 1 distinct from ±1, and both cases give a factor of m. If $\langle x,y\rangle\in D$, (n,x) or (m,y) is a nontrivial divisor of n or m, respectively. □

We are ready now to prove Theorem 3.4. Assume we are given an odd $n>0$, and an integer a such that $(a|n)=1$. We first compute the sequences $\langle a_i:i\le t\rangle$, $\langle n_i:i\le t\rangle$ of values of a and n during the execution of the algorithm in Figure 2.1. That is, we put $\langle a_0,n_0\rangle=\langle a,n\rangle$, and then we define $\langle a_i,n_i\rangle$ by induction on i as follows. If $|a_i|>n_i/2$, we let $n_{i+1}=n_i$, and $a_{i+1}\equiv a_i\ (n_i)$ such that $|a_{i+1}|<n_i/2$. If $0<|a_i|<n_i/2$, we define

$$\langle a_{i+1},n_{i+1}\rangle=\begin{cases}\langle -a_i,n_i\rangle & a_i<0,\\ \langle a_i/2,n_i\rangle & a_i>0\text{ is even,}\\ \langle n_i,a_i\rangle & a_i>0\text{ is odd.}\end{cases}$$

We stop when we reach $a_t=0$. Since $(a|n)=1$, we have $(a_i,n_i)=1$ for each i, in particular $n_t=1$. Notice that $t=O(\|n\|)$. Write $R=\{i<t: a_i\text{ is odd}, 0<a_i<n_i/2\}$.

In the main part of the algorithm, we maintain a double sequence $\langle n_{i,j}:i\le t, j<s_i\rangle$ of integers $n_{i,j}>1$ such that $n_i=\prod_{j<s_i}n_{i,j}$, and $n_{i,j}\le n_{i,j'}$ for $j<j'$. Moreover, we maintain sequences $\langle u_{i,j}:i\le k,j<s_i\rangle$, $\langle v_{i,j,k},w_{i,j,k}:i\in R,j<s_i,k<s_{i+1}\rangle$, where some of the $u_{i,j}$, $v_{i,j,k}$, and $w_{i,j,k}$ may be undefined. Where they are defined, we have $u_{i,j}^2\equiv a_i\ (n_{i,j})$, $v_{i,j,k}^2\equiv n_{i+1,k}\ (n_{i,j})$, and $w_{i,j,k}^2\equiv n_{i,j}\ (n_{i+1,k})$, respectively.

We initialize the sequences with $s_i=1$, $n_{i,0}=n_i$ for $n_i>1$, $s_i=0$ for $n_i=1$, and all $u_{i,j}$, $v_{i,j,k}$, and $w_{i,j,k}$ undefined. We repeat in arbitrary order the following updating steps until none of them is applicable any more.

- Assume $n_i=n_{i+1}$, $n_{i,j}\ne n_{i+1,k}$, and $d=(n_{i,j},n_{i+1,k})>1$. If $d\ne n_{i,j}$, we increase s_i, replace $n_{i,j}$ with d and $n_{i,j}/d$, and undefine all associated $u_{i,j}$, $v_{i-1,l,j}$, and $w_{i-1,l,j}$. If $d\ne n_{i+1,k}$, we deal with it similarly. Notice that we cannot have $n_{i,j}=d=n_{i+1,k}$.

Moreover, if this step is not applicable, then $n_i = n_{i+1}$ implies that $s_i = s_{i+1}$ and $\langle n_{i,j} : j < s_i\rangle$ and $\langle n_{i+1,k} : k < s_{i+1}\rangle$ are permutations of each other, hence in view of their monotonicity, we have $n_{i,j} = n_{i+1,j}$ for each j.

- For $i < t$ such that $\langle n_{i,j} : j < s_i\rangle = \langle n_{i+1,k} : k < s_{i+1}\rangle$ (which implies $n_i = n_{i+1}$):
  - If $a_i \equiv a_{i+1}\ (n_i)$, and exactly one of $u_{i,j}$, $u_{i+1,j}$ is defined, we define the other to the same value.
  - If $a_i = \alpha a_{i+1}$, $\alpha \in \{-1, 2\}$, $(\alpha|n_{i,j}) = -1$, and neither $u_{i,j}$ nor $u_{i+1,j}$ is defined, we call FacRootMul$(n_{i,j}, a_i, a_{i+1})$. If it returns a nontrivial divisor of $n_{i,j}$, we expand the $n_{i,j}$ sequence as in the first step. Otherwise, it gives a square root of a_i or a_{i+1} modulo $n_{i,j}$, which we store as $u_{i,j}$ or $u_{i+1,j}$, respectively.
  - If $a_i = \alpha a_{i+1}$, $\alpha \in \{-1, 2\}$, $(\alpha|n_{i,j}) = 1$, and exactly one of $u_{i,j}$ or $u_{i+1,j}$ is defined, we call FacRoot$_\alpha(n_{i,j})$. If it returns a nontrivial divisor of $n_{i,j}$, we expand the $n_{i,j}$ sequence. Otherwise, it gives $\beta^2 \equiv \alpha\ (n_{i,j})$, and we define $u_{i,j} := \beta u_{i+1,j}$ or $u_{i+1,j} := \beta^{-1} u_{i,j}$, respectively.
- For $i \in R$:
  - If $u_{i,j}$ is defined and $|I|$ is odd, where $I = \{k < s_{i+1} : v_{i,j,k} \text{ is undefined}\}$, we put $x = u_{i,j} \prod_{k \notin I} v_{i,j,k}^{-1}$, and call FacRootOdd on $n_{i,j}, \langle n_{i+1,k} : k \in I\rangle, x$. If it returns a factor of $n_{i,j}$, we expand the $n_{i,j}$ sequence. Otherwise, it returns a square root of some $n_{i+1,k}$, $k \in I$, modulo $n_{i,j}$, which we store as $v_{i,j,k}$.
  - If $u_{i,j}$ is undefined and $|I|$ is even, where I is as above, we call FacRootEven on $n_{i,j}, \langle n_{i+1,k} : k \in I\rangle$. If it returns a factor of $n_{i,j}$, we expand the $n_{i,j}$ sequence. If it returns a square root of some $n_{i+1,k}$, $k \in I$, modulo $n_{i,j}$, we store it as $v_{i,j,k}$. Otherwise, it returns a square root x of $\prod_{k \in I} n_{i+1,k}$, and then we define $u_{i,j} = x \prod_{k \notin I} v_{i,j,k}$.
  - If $u_{i+1,k}$ is defined and $|I|$ is odd, or $u_{i+1,k}$ is undefined and $|I|$ is even, where $I = \{j < s_i : w_{i,j,k} \text{ is undefined}\}$, we proceed in a similar way to expand the $n_{i+1,k}$ sequence or to define some $w_{i,j,k}$ or $u_{i+1,k}$.
  - If $n_{i,j} \equiv -1\ (4)$, $(n_{i+1,k}|n_{i,j}) = 1$, and $v_{i,j,k}$ is undefined, we call WeakFacRoot on $n_{i,j}, n_{i+1,k}, -1$. If it returns a factor of $n_{i,j}$, we expand the $n_{i,j}$ sequence, otherwise it returns a square root of $n_{i+1,k}$ modulo $n_{i,j}$, which we store as $v_{i,j,k}$.
  - If $n_{i+1,k} \equiv -1\ (4)$, $(n_{i,j}|n_{i+1,k}) = 1$, and $w_{i,j,k}$ is undefined, we proceed similarly.
  - If $n_{i,j} \equiv 1\ (4)$, $w_{i,j,k}$ is defined, and $v_{i,j,k}$ is undefined, we call QuadRec on $n_{i,j}, n_{i+1,k}, w_{i,j,k}$. If it returns a factor of $n_{i,j}$ or $n_{i+1,k}$, we expand the $n_{i,j}$ or $n_{i+1,k}$ sequence (respectively), otherwise it returns a square root of $n_{i+1,k}$ modulo $n_{i,j}$, which we store as $v_{i,j,k}$.
  - If $n_{i+1,k} \equiv 1\ (4)$, $v_{i,j,k}$ is defined, and $w_{i,j,k}$ is undefined, we proceed similarly.

In each step, either $\sum_{i\leq t} s_i \leq O(\|n\|^2)$ strictly increases, or it stays the same, and we define some previously undefined $u_{i,j}$, $v_{i,j,k}$, or $w_{i,j,k}$. It follows that the update procedure stops after $\|n\|^{O(1)}$ steps.

Let us write $[a_i|n_{i,j}] = 1$ if $u_{i,j}$ is defined, and $[a_i|n_{i,j}] = -1$ otherwise. We define $[n_{i+1,k}|n_{i,j}]$ and $[n_{i,j}|n_{i+1,k}]$ similarly using $v_{i,j,k}$ and $w_{i,j,k}$, respectively. Notice that $[a_i|n_{i,j}] = 1$ implies $(a_i|n_{i,j}) = 1$, and likewise for $[n_{i+1,k}|n_{i,j}]$, $[n_{i,j}|n_{i+1,k}]$.

Lemma 4.11 *When the update procedure stops, the following properties hold.*

(i) *If $n_i = n_{i+1}$, then $s_i = s_{i+1}$ and $n_{i,j} = n_{i+1,j}$.*

(ii) *If $n_i = n_{i+1}$ and $a_i \equiv a_{i+1} \pmod{n_i}$, then $[a_i|n_{i,j}] = [a_{i+1}|n_{i+1,j}]$.*

(iii) *If $n_i = n_{i+1}$ and $a_i = \alpha a_{i+1}$, $\alpha \in \{-1, 2\}$, then*

$$\left[\frac{a_{i+1}}{n_{i+1,j}}\right] = \left(\frac{\alpha}{n_{i,j}}\right)\left[\frac{a_i}{n_{i,j}}\right].$$

(iv) *If $i \in R$, then*

$$\left[\frac{a_i}{n_{i,j}}\right] = \prod_{k<s_{i+1}} \left[\frac{n_{i+1,k}}{n_{i,j}}\right], \qquad \left[\frac{a_{i+1}}{n_{i+1,k}}\right] = \prod_{j<s_i} \left[\frac{n_{i,j}}{n_{i+1,k}}\right].$$

(v) *If $i \in R$ and $n_{i,j} \equiv n_{i+1,k} \equiv -1 \pmod 4$, then*

$$\left[\frac{n_{i+1,k}}{n_{i,j}}\right]\left[\frac{n_{i,j}}{n_{i+1,k}}\right] = -1.$$

(vi) *If $i \in R$ and $n_{i,j} \equiv 1 \pmod 4$ or $n_{i+1,k} \equiv 1 \pmod 4$, then*

$$\left[\frac{n_{i+1,k}}{n_{i,j}}\right]\left[\frac{n_{i,j}}{n_{i+1,k}}\right] = 1.$$

Proof:

(i), (ii), and (iv) are clear.

(iii): The statement is clear if $(\alpha|n_{i,j}) = 1$. If $(\alpha|n_{i,j}) = -1$, the inapplicability of update steps implies that $[a_i|n_{i,j}] = 1$ or $[a_{i+1}|n_{i+1,j}] = 1$. We cannot have both, since this would imply $(a_i|n_{i,j}) = (a_{i+1}|n_{i,j}) = 1$, hence $(\alpha|n_{i,j}) = 1$.

(v): By quadratic reciprocity, exactly one of $(n_{i+1,k}|n_{i,j}) = 1$, $(n_{i,j}|n_{i+1,k}) = 1$ holds. The inapplicability of update steps then implies that $[n_{i+1,k}|n_{i,j}] = 1$ or $[n_{i,j}|n_{i+1,k}] = 1$. We cannot have both, as this would mean $(n_{i+1,k}|n_{i,j}) = (n_{i,j}|n_{i+1,k}) = 1$.

(vi): The statement is clear if $n_{i,j} \equiv n_{i+1,k} \equiv 1 \pmod 4$. Assume $n_{i,j} \equiv 1 \pmod 4$ and $n_{i+1,k} \equiv -1 \pmod 4$, the other case is symmetric. By the inapplicability of update steps, $[n_{i,j}|n_{i+1,k}] = 1$ implies $[n_{i+1,k}|n_{i,j}] = 1$. On the other hand, if $[n_{i+1,k}|n_{i,j}] = 1$, then $(n_{i+1,k}|n_{i,j}) = 1$, hence $(n_{i,j}|n_{i+1,k}) = 1$ by quadratic reciprocity, thus $[n_{i,j}|n_{i+1,k}] = 1$ by the inapplicability of update steps. □

Using Lemma 4.11, we can show

$$\prod_{j<s_i}\left[\frac{a_i}{n_{i,j}}\right]=\left(\frac{a_i}{n_i}\right)$$

by reverse induction on i. The induction step for $i\in R$ goes as follows:

$$\begin{aligned}\prod_{j<s_i}\left[\frac{a_i}{n_{i,j}}\right]&=\prod_{\substack{j<s_i\\k<s_{i+1}}}\left[\frac{n_{i+1,k}}{n_{i,j}}\right]\\&=\prod_{\substack{j<s_i\\k<s_{i+1}}}\left[\frac{n_{i,j}}{n_{i+1,k}}\right](-1)^{(n_{i,j}-1)(n_{i+1,k}-1)/4}\\&=(-1)^{(a_{i+1}-1)(n_{i+1}-1)/4}\prod_{k<s_{i+1}}\left[\frac{a_{i+1}}{n_{i+1,k}}\right]\\&=(-1)^{(a_{i+1}-1)(n_{i+1}-1)/4}\left(\frac{a_{i+1}}{n_{i+1}}\right)\\&=\left(\frac{n_{i+1}}{a_{i+1}}\right)=\left(\frac{a_i}{n_i}\right).\end{aligned}$$

In particular, either $s_0>1$, in which case $n_{0,0}$ is a nontrivial divisor of n, or $s_0=1$ and $[a_0|n_{0,0}]=1$, where $a_0=a$ and $n_{0,0}=n$, in which case $u_{0,0}^2\equiv a\ (n)$. This completes the proof of Theorem 3.4.

5 Conclusion

We have shown that integer factoring has randomized reductions to the classes PPA and PPP (more precisely, PWPP). We also provided evidence that there in fact exist deterministic reductions, namely this is true under the widely believed assumption of the generalized Riemann hypothesis for quadratic Dirichlet characters.

Problem 5.1 *Is* FACTORING *in* PPA, PPP, *or* $\mathrm{FP}^{\mathrm{PPP}}$*?*

Some of our other results can be seen as partial indication that such an unconditional deterministic reduction might be possible at least in the case of PPA. In particular, the fact that FACROOT $\in$ PPA bypasses the randomized reduction of WEAKFACROOT to FACROOT, and we have shown that PPA contains the search problems to find square roots modulo arbitrary integers (which is probabilistically Turing-equivalent to factoring) and to find quadratic nonresidues (which is easily solvable in randomized polynomial time). Nevertheless, it remains open whether Problem 5.1 can be resolved unconditionally.

Another interesting question is whether the methods used for the reduction of factoring to PPA can be pushed down to the class PPAD $\subseteq$ PPA. Note that many natural problems are known to be complete for PPAD, such as computing Nash equilibria [51].

Problem 5.2 *Does* FACTORING *have some form of reduction to* PPAD*?*

Acknowledgements

I would like to thank Josh Buresh-Oppenheim for a clarification of his work, and Rahul Savani for a useful suggestion.

The research was supported by grant IAA100190902 of GA AV ČR, Center of Excellence CE-ITI under the grant P202/12/G061 of GA ČR, and RVO: 67985840. Part of the research was done while visiting the Isaac Newton Institute in Cambridge.

Chapter V

On theories of bounded arithmetic for NC^1

Abstract

We develop an arithmetical theory VNC^1_* and its variant $\overline{VNC}^1_*$ that correspond to "slightly nonuniform" NC^1. Our theories sit between VNC^1 and VL, and allow evaluation of log-depth bounded fan-in circuits under limited conditions. Propositional translations of $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formulas provable in $\overline{VNC}^1_*$ admit L-uniform polynomial-size Frege proofs.

1 Introduction

In proof complexity, there is a well-known general correspondence between theories of bounded arithmetic, complexity classes, and propositional proof systems (see e.g. [116, 56, 65, 68]). A theory T corresponds to a complexity class C if the provably total computable functions of T are the C-functions. A propositional proof system P corresponds to T if the propositional translations of theorems of T of certain complexity have polynomial-size proofs in P, and T proves a reflection principle for P.

Here we are particularly concerned about theories corresponding to variants of the class NC^1. Several theories corresponding to *uniform* NC^1 (i.e., ALOGTIME, U_E-uniform NC^1) and to the Frege propositional proof system have been described in the literature: an equational theory ALV by Clote [53], theories AID and $AID + \Sigma^b_0\text{-}CA$ by Arai [11], and a second-order theory VNC^1 by Cook and Morioka [61]. (All these theories are more or less equivalent: VNC^1 is $RSUV$-isomorphic to $AID + \Sigma^b_0\text{-}CA$, which is in turn a conservative extension of ALV.)

Uniform NC^1 is a robust and well-behaved complexity class, but it is too strict for certain applications, namely those involving circuit evaluation. Nonuniform complexity classes usually consist of languages definable by a family of polynomial-size Boolean circuits satisfying certain requirements (e.g., concerning their depth, fan-in, or available connectives): this holds for example for nonuniform AC^k, NC^k, TC^0, P; in particular, nonuniform NC^1-languages are given by a family of bounded fan-in circuits of logarithmic depth. Typically, the corresponding uniform class consists of languages definable by a sufficiently uniform family of the same kind of circuits, and moreover, the class includes the universal language which evaluates circuits of this kind described in a natural way by binary strings. This is not true for NC^1. Even DLOGTIME-uniform (i.e., U_D-uniform) families of log-depth circuits define a class (presumably) still larger than uniform NC^1; we can only define uniform NC^1 using circuits by employing the more complicated description by so-called extended connection languages of Ruzzo [162]. Likewise, the universal evaluator for log-depth circuits is (presumably) not in NC^1 (even nonuniform).

Consequently, VNC^1 (and friends) do not prove that one can evaluate log-depth circuits, or even a uniformly given (say, definable by a Σ^B_0-formula) sequence of log-depth circuits. There are situations where evaluation of such circuits would be desirable in an NC^1-theory. The particular application we have in mind, and the main motivation for this work, is Chapter VI, which aims at formalizing a version of the Ajtai–Komlós–Szemerédi sorting network in bounded arithmetic (under the assumption that we can formalize construction of suitable expander graphs). On the one hand, we need the formalization to proceed in an NC^1-theory, and in particular, in a theory which translates to polynomial-time Frege proofs: the point is that this implies polynomial simulation of the sequent calculus (i.e., Frege) by the monotone sequent calculus MLK, using results of Atserias, Galesi, and Pudlák [14]. On the other hand, the sorting network is essentially a monotone log-depth circuit which we need to evaluate; it is uniformly described, but its extended connection language is not available.

To address these issues, we introduce new theories VNC^1_* and $\overline{VNC}^1_*$, corresponding to a subclass of NC^1 slightly larger than uniform NC^1, which allow evaluation of sufficiently uniform families of log-depth circuits. We work with second-order theories in the spirit of Zambella [182]. The theory VNC^1_* is formulated in the usual language of second-order bounded arithmetic; it includes V^0, and a derivation rule allowing to evaluate a kind of monotone log-depth bounded fan-in circuits described by formulas without second-order parameters which are provably Δ^B_1. The theory $\overline{VNC}^1_*$ has a richer language $L_{\overline{VNC}^1_*}$ including comprehension function symbols for Σ^B_0-formulas, and function symbols for evaluation of monotone log-depth bounded fan-in circuits described by open formulas (in the extended language) without second-order parameters.

In Section 4, we prove basic properties of our new theories: VNC^1_* contains VNC^1 and is contained in VL, $\overline{VNC}^1_*$ is an open theory conservatively extending VNC^1_* (more precisely, it is an extension of VNC^1_* by Σ^B_1-definitions), VNC^1_* is Σ^B_1-axiomatizable, $\exists\Sigma^B_1$-formulas provable in $\overline{VNC}^1_*$ are witnessed by terms in $\overline{VNC}^1_*$ (in particular, provably Δ^B_1-formulas of $\overline{VNC}^1_*$ are equivalent to open formulas), the provably total computable functions of VNC^1_* include uniform (and even U_D-uniform) NC^1-functions, and are included in L-uniform NC^1-functions, and VNC^1_* extended by the axiom of choice for $\exists\Sigma^B_1$-formulas is $\exists\Sigma^B_1$-conservative over VNC^1_*. To show the latter, we prove a general theorem on conservativity of the axiom of choice over theories meeting certain requirements. In Section 5 we show that propositional translations of $\Sigma^B_0(L_{\overline{VNC}^1_*})$-theorems of $\overline{VNC}^1_*$ have L-uniform polynomial-size Frege proofs.

2 Complexity classes

We recall that a (bounded fan-in) circuit in n inputs is a directed acyclic graph whose nodes are labelled by gate types $\wedge$, $\vee$, $\neg$, or input variables x_i, $i < n$. Input nodes have fan-in 0, $\neg$-gates have fan-in 1, and $\wedge$ and $\vee$-gates have fan-in 2. One node of the circuit is designated as the output node. The circuit computes a Boolean function $f\colon 2^n \to 2$ in the obvious way. The depth of a circuit is the maximal length of a path in the circuit. A formula is a circuit in which all nodes save the output have fan-out 1.

If C is any class of languages, we define FC to be the class of functions $f(\vec{x})$ such that $|f(\vec{x})|$ is at most polynomial in $|\vec{x}|$, and the bit-graph

$$\{\langle \vec{x}, i\rangle \mid \text{the } i\text{th bit of } f(\vec{x}) \text{ is } 1\}$$

is in C. We will sometimes call functions $f \in FC$ just C-functions.

A language L is in *nonuniform* NC^1 if there exists a family $\{C_n \mid n \in \omega\}$ of circuits such that C_n computes the characteristic function of $L \cap 2^n$, and the depth of C_n is $O(\log n)$ (in short, C_n is a log-depth circuit). Equivalently, L is in nonuniform NC^1 if it is computable in a similar way by a family of polynomial-size formulas.

Let U be a complexity class. A language L is in U*-uniform* NC^1 if it is computable by a sequence $\{C_n \mid n \in \omega\}$ of log-depth circuits such that, given n in unary, we can compute the description of C_n by a U-function. Since this definition may be sensitive to details of the chosen representation of circuits, we make it more precise using the terminology of Ruzzo [162]. Given a node x in a circuit C, we fix an ordering of its input nodes, and denote by $x(i)$ the ith input of x. The *direct connection language* $L_{DC}(C)$ of a family of circuits $C = \{C_n \mid n \in \omega\}$, where C_n has n inputs, is a set of tuples $\langle n, x, p, y\rangle$, where n is an integer given in unary, x is a binary string identifying a node in a circuit, $p \in \{\varepsilon, 0, 1\}$, and y is either another string denoting a node, or a gate type from $\{x_i, \wedge, \vee, \neg\}$. It is defined by

$$L_{DC}(C) = \{\langle n, x, \varepsilon, t\rangle \mid \text{node } x \text{ in } C_n \text{ is a } t\text{-gate}\} \cup \{\langle n, x, p, y\rangle \mid p \in \{0,1\}, x(p) = y \text{ in } C_n\}.$$

We define U-uniform NC^1 to consist of languages L computable by a family C of log-depth circuits with node labels of length $|x| = O(\log n)$ such that $L_{DC}(C) \in U$. DLOGTIME-uniform NC^1 is usually called U_D*-uniform*, where DLOGTIME $=$ DTIME$(O(\log n))$. Here and below, Turing machines supposed to work in sublinear time do not have the usual input tape. Instead, there is a special index type, and read states. If the machine enters a read state with a, k written on the index tape, where a is a symbol of the input alphabet, and k is a binary integer, it continues in one of two given states according to whether the kth symbol of the input is a.

Fully *uniform* NC^1 (also called U_E-uniform) is defined as ALOGTIME, the languages computable by an alternating Turing machine in $O(\log n)$ steps. Uniform NC^1 is not known to coincide with U-uniform NC^1 for any natural class U. However, we can define it using circuits as follows. We extend the $x(i)$ notation so that if p is a binary string, $x(p)$ is the node we obtain by following the path which starts in x, and moves to the left or right input according to successive bits of p. The *extended connection language* $L_{EC}(C)$ of a family $C = \{C_n \mid n < \omega\}$ of circuits is defined by

$$\begin{aligned} L_{EC}(C) = \{\langle n, x, \varepsilon, t\rangle &\mid \text{node } x \text{ in } C_n \text{ is a } t\text{-gate}\} \\ &\cup \{\langle n, x, p, y\rangle \mid p \in \{0,1\}^*, 0 < |p| \le \log n, x(p) = y \text{ in } C_n\}. \end{aligned}$$

Then a language L is in uniform NC^1 if and only if it is computable by a family C of log-depth circuits such that $L_{EC}(C)$ is computable in DLOGTIME. The class does not change if we allow $L_{EC}(C)$ to be in AC^0 or ALOGTIME. Here, (uniform) AC^0 can be defined as languages computable by an alternating Turing machine in time $O(\log n)$ with $O(1)$ alternations.

Buss [38] has shown that one can evaluate in uniform NC^1 Boolean formulas represented as strings in the usual infix notation. We can define the extended connection language for a single circuit (rather than sequence) in a natural way, and represent it as a polynomial-size string. Log-depth circuits in this representation can be also evaluated in uniform NC^1 (this is implicit in Ruzzo [162]). On the other hand, evaluation of log-depth circuits represented by the direct connection language (or equivalent form) is not known to be possible even in nonuniform NC^1, but it can be done in logarithmic space. (One reason why formulas are easier to evaluate than circuits is that they carry more structure due to linear order on their symbols: it is not hard to see that given a formula in the usual notation, we can compute its L_{EC} in uniform TC^0, which immediately implies it can be evaluated in uniform NC^1 if it has logarithmic depth. The hard part of Buss's algorithm is to deal with formulas of arbitrary depth.) Regarding the former, we observe the following reduction of a combinatorial problem which is apparently not in nonuniform NC^1:

Proposition 2.1 *The following problem is many-one* AC^0*-reducible to evaluation of bounded fan-in log-depth circuits (described by* L_{DC}*). Given a directed graph* G *on* n *vertices with bounded out-degree, vertices* $x, y \in G$*, and a number* $d \leq \log n$*, determine whether* y *is reachable from* x *in at most* d *steps.*

Proof: Without loss of generality assume that G contains all self-loops. We construct a circuit with $d+1$ layers, where each layer is labeled by nodes of G. Every node u on layer $l+1$ is a disjunction gate, and its inputs are nodes v on layer l such that $u \to v$ is an edge of G. We initialize the bottom layer by assigning 1 to node y, and 0 to all other nodes, and we evaluate the circuit. Then the value of node x on the top layer is 1 iff y is reachable from x in d steps. □

A kind of converse also holds: it can be shown that an algorithm for the problem described in Proposition 2.1, even restricted to graphs with out-degree 1 (this problem is denoted by $\mathrm{REACH}_1(\log n)$ in Allender and Barrington [8]), can be used to transform a direct connection language of a log-depth circuit to its extended connection language, which can be evaluated in uniform NC^1. (That is, using an appropriate notion of relativization of uniform circuit classes, evaluation of bounded fan-in log-depth circuits is many-one $(\mathrm{AC}^0)^{\mathrm{REACH}_1(\log n)}$-reducible to NC^1, and in particular, it is in $(\mathrm{NC}^1)^{\mathrm{REACH}_1(\log n)}$.) The complexity of $\mathrm{REACH}_1(\log n)$ is briefly discussed in [8]; in particular, they observe that it lies in the class FOLL introduced by Barrington, Kadau, Lange, and McKenzie [20], consisting of languages computable by uniform families of polynomial-size unbounded fan-in circuits of depth $O(\log\log n)$.

3 Theories

We will work with second-order (i.e., two-sorted) arithmetical theories as in [182, 68], but for convenience we include the function $|x| = \lceil \log_2(x+1) \rceil$ among the basic symbols. Our theories thus have two sorts of variables: numbers, denoted by lowercase letters, and finite sets or strings, denoted by uppercase letters. The basic language is $L_0 = \langle 0, \mathrm{s}, +, \cdot, |x|, \le, \in, |X|\rangle$. The theory *BASIC* consists of the axioms

$$\begin{array}{ll}
x + 0 = x & x + \mathrm{s}\,y = \mathrm{s}(x+y) \\
x \cdot 0 = 0 & x \cdot \mathrm{s}\,y = x \cdot y + x \\
\mathrm{s}\,y \le x \to y < x & x \ne 0 \to \exists y\, x = \mathrm{s}\,y \\
x \in X \to x < |X| & \mathrm{s}\,x = |X| \to x \in X \\
|0| = 0 & x \ne 0 \to |x + x| = \mathrm{s}|x| \\
\forall x\,(x \in X \leftrightarrow x \in Y) \to X = Y & |\mathrm{s}(x+x)| = \mathrm{s}|x|
\end{array}$$

where $x < y$ is an abbreviation for $x \le y \wedge x \ne y$. We also write $X(x)$ for $x \in X$. We define the constants $1 = \mathrm{s}\,0$, $2 = \mathrm{s}\,\mathrm{s}\,0$, $3 = \mathrm{s}\,\mathrm{s}\,\mathrm{s}\,0$, $\dots$, and we will often write $x+1$ for $\mathrm{s}\,x$ (the two expressions being equal by the *BASIC* axioms). We introduce the bounded quantifiers

$$\begin{aligned}
\exists x \le t\,\varphi &\Leftrightarrow \exists x\,(x \le t \wedge \varphi), \\
\forall x \le t\,\varphi &\Leftrightarrow \forall x\,(x \le t \to \varphi), \\
\exists X \le t\,\varphi &\Leftrightarrow \exists X\,(|X| \le t \wedge \varphi), \\
\forall X \le t\,\varphi &\Leftrightarrow \forall X\,(|X| \le t \to \varphi),
\end{aligned}$$

where t is a term not involving x or X (respectively), and similarly for strict inequalities. A formula is bounded if it uses only bounded quantifiers. A bounded L_0-formula without set quantifiers is called Σ^B_0 or Π^B_0. Inductively, Σ^B_{i+1} consists of formulas of the form

$$\exists X_1 \le t_1 \dots \exists X_n \le t_n\,\varphi$$

for $\varphi \in \Pi^B_i$, and Π^B_{i+1} consists of formulas of the form

$$\forall X_1 \le t_1 \dots \forall X_n \le t_n\,\varphi$$

for $\varphi \in \Sigma^B_i$. A formula is Σ^1_1 if it consists of a block of second-order existential quantifiers followed by a Σ^B_0-formula. A predicate is Σ^B_0-definable in the standard model iff it is computable in AC^0, and for $i > 0$, the Σ^B_i-definable (Π^B_i-definable) predicates coincide with the levels Σ^P_i (Π^P_i) of the polynomial hierarchy. Note that we use Σ^B_i and Π^B_i to denote formulas of the basic language L_0 only. If we expand the definition to allow atomic formulas in a richer language L, we will call the corresponding classes $\Sigma^B_i(L)$ and $\Pi^B_i(L)$, respectively.

If Γ is a set of formulas, the Γ-comprehension axiom is the schema

$$(\Gamma\text{-}\mathit{COMP}) \qquad \exists X \le x\, \forall u < x\,(u \in X \leftrightarrow \varphi),$$

where $\varphi \in \Gamma$ has no free occurrence of X. We define the theory V^0 as $\mathit{BASIC} + \Sigma^B_0\text{-}\mathit{COMP}$.

The theory VNC^1 is axiomatized over V^0 by

$$\begin{aligned}
&\exists Y \le 2a\, \forall x < a\, [(Y(x+a) \leftrightarrow I(x)) \\
&\qquad \wedge (Y(x) \leftrightarrow ((G(x) \wedge (Y(2x) \vee Y(2x+1))) \vee (\neg G(x) \wedge Y(2x) \wedge Y(2x+1))))].
\end{aligned}$$

The meaning is that we can evaluate a monotone formula laid out in a balanced binary tree with $2a-1$ nodes, represented by nonzero numbers below $2a$ so that nodes $0 < x < a$ are conjunction or disjunctions (according to $G(x)$) of nodes $2x$ and $2x+1$, and nodes $a \le x < 2a$ are truth constants given by I.

The theory VL is axiomatized over V^0 by the axiom

$$\forall x < a\, \exists! y < a\, F(x,y) \to \exists P\, [(P)_0 = 0 \wedge \forall v < a\, F((P)_v, (P)_{v+1})],$$

where P encodes a sequence of numbers, and $(P)_v$ is the vth member of the sequence (see [68] for details of the sequence coding). The meaning is that we can iterate a number function, or equivalently, that we can trace a path in a directed graph where each node has out-degree 1.

Let $\varphi(d,x,y)$ be a formula, possibly with other free variables. We put

$$\varphi^*(d,x,y) \Leftrightarrow \varphi(d,x,y) \wedge (\forall z < y\, \neg\varphi(d,x,z) \vee \forall z > y\, \neg\varphi(d,x,z)),$$

$$\begin{aligned}\mathrm{eval}(n,m,\varphi,I,Y) \Leftrightarrow \forall x < n\, [&(Y(0,x) \leftrightarrow I(x)) \\ &\wedge \forall d < m\, (Y(d+1,x) \leftrightarrow ((2 \mid d \wedge \exists y < n\, (\varphi^*(d,x,y) \wedge Y(d,y))) \\ &\qquad \vee (2 \nmid d \wedge \forall y < n\, (\varphi^*(d,x,y) \to Y(d,y)))))],\end{aligned}$$

where $Y(d,x)$ stands for $dn+x \in Y$. (By abuse of notation, we include φ among the arguments of eval to indicate the dependence of eval on φ, even though φ is a formula, not a variable. Note that free variables of eval include parameters of φ, i.e., its free variables other than d,x,y.) The meaning of eval is that Y is the evaluation of a bounded fan-in monotone circuit described by φ on input I. The circuit consists of $m+1$ layers, each with n nodes. Nodes on layer 0 are truth constants given by I. Layers $d > 0$ consist of alternating disjunction (odd d) and conjunction (even d) gates. Gates on level d can only use nodes on level $d-1$ as inputs. The formula $\varphi(d,x,y)$ means that node x on level $d+1$ uses node y on level d as input. The formula φ^* is actually employed instead of φ to force each gate to have at most two inputs.

We define VNC^1_* to be the closure of V^0 under the derivation rule

$$(\Delta^B_1\text{-}SCV) \qquad \frac{\varphi \leftrightarrow \neg\varphi'}{\exists Y \le (|m|+1)n\ \mathrm{eval}(n,|m|,\varphi,I,Y)},$$

where φ and φ' are Σ^B_1-formulas with no free set variables. (A Σ^B_1-formula provably equivalent to a Π^B_1-formula in a theory T will be called a $\Delta^B_1(T)$-formula. SCV stands for “shallow circuit value”.)

The language $L_{\overline{VNC}^1_*}$ contains L_0, and a function symbol $C_\varphi(n,\vec{x},\vec{X})$ for each Σ^B_0-formula $\varphi(u,\vec{x},\vec{X})$ (with all free variables indicated). Moreover, it is closed under the following rule: for each open $L_{\overline{VNC}^1_*}$-formula $\varphi(\vec{p},d,x,y)$ without free set variables (but with arbitrary free number variables, viz $\vec{p}$), we include a function symbol $Y_\varphi(\vec{p},n,m,I)$. We will usually denote $C_\varphi(n,\vec{x},\vec{X})$ by $\{u < n \mid \varphi(u,\vec{x},\vec{X})\}$.

$\overline{VNC}^1_*$ is a theory in $L_{\overline{VNC}^1_*}$ consisting of the axioms of $BASIC$, the axiom

$$(\Sigma^B_0\text{-}\overline{COMP}) \qquad u \in C_\varphi(n,\vec{x},\vec{X}) \leftrightarrow u < n \wedge \varphi(u,\vec{x},\vec{X})$$

for each Σ^B_0-formula $\varphi(u,\vec{x},\vec{X})$, and the axiom

$$(\textit{Open-}\overline{SCV}) \qquad |Y_\varphi(\vec{p},n,m,I)| \le (|m|+1)n \wedge \mathrm{eval}(n,|m|,\varphi,I,Y_\varphi(\vec{p},n,m,I))$$

for each open $L_{\overline{VNC}^1_*}$-formula $\varphi(\vec{p},d,x,y)$. (That is, C_φ is the bounded comprehension term for φ, or as a string, the truncated characteristic function of φ. Y_φ gives a string containing evaluation of all gates of the circuit described by φ, just like the variable Y in Δ^B_1-SCV above.)

Notice that $\overline{VNC}^1_*$ contains V^0.

4 Properties of VNC^1_* and $\overline{VNC}^1_*$

The Δ^B_1-SCV and *Open-*$\overline{SCV}$ axioms provide evaluation of a certain type of circuits, but they were designed to be formally simple rather than feature-rich. We will introduce a more elaborate setting for convenient evaluation of log-depth circuits.

We will describe circuits using the following data:

- Numbers k, m, and s, where k is the number of input bits, m is the number of layers, and s is the size of each layer (we assume all layers have been padded with unused gates to have the same size).
- A function $T\colon m\times s\to\{\ulcorner\vee\urcorner,\ulcorner\wedge\urcorner,\ulcorner\neg\urcorner\}\cup\{\ulcorner x_i\urcorner\mid i<k\}$ indicating the type of each node, where we put e.g. $\ulcorner\vee\urcorner=0$, $\ulcorner\wedge\urcorner=1$, $\ulcorner\neg\urcorner=2$, and $\ulcorner x_i\urcorner=i+3$, and we represent T by its graph (a set $T\le ms(k+3)$): i.e., $T(d,x,p)$ iff xth node on layer d has type p.
- A formula $\varphi(d,x,d',x')$ (possibly with other parameters) which states that node x' on layer d' is an input of gate x on layer d.

In order for a circuit to be well-formed, we demand that any gate uses only nodes on lower layers as inputs (but not necessarily from the adjacent layer), and all nodes have the correct number of inputs: 1 for negation nodes, 0 for input nodes, and at most 2 for conjunction and disjunction gates. Notice that we allow $\wedge$ and $\vee$ gates with no inputs, which compute the truth constants $\bot$ and $\top$, or with one input, which act as the identity function. The formula

$$\begin{aligned}
\mathrm{Circ}(k,m,s,T,\varphi)\Leftrightarrow{}& \forall d<m\,\forall x<s\,\exists!p<k+3\,T(d,x,p)\\
&\wedge\forall d,d'<m\,\forall x,x'<s\,(\varphi(d,x,d',x')\to d'<d)\\
&\wedge\forall d,d_0,d_1,d_2<m\,\forall x,x_0,x_1,x_2<s\\
&\qquad\qquad\Bigl(\bigwedge_{i<3}\varphi(d,x,d_i,x_i)\to\bigvee_{i<j}(d_i=d_j\wedge x_i=x_j)\Bigr)\\
&\wedge\forall d,d_0,d_1<m\,\forall x,x_0,x_1<s\\
&\qquad\qquad\Bigl(T(d,x,\ulcorner\neg\urcorner)\wedge\bigwedge_{i<2}\varphi(d,x,d_i,x_i)\to d_0=d_1\wedge x_0=x_1\Bigr)\\
&\wedge\forall d<m\,\forall x<s\,(T(d,x,\ulcorner\neg\urcorner)\to\exists d'<m\,\exists x'<s\,\varphi(d,x,d',x'))\\
&\wedge\forall d,d'<m\,\forall x,x'<s\,\forall i<k\,(T(d,x,\ulcorner x_i\urcorner)\to\neg\varphi(d,x,d',x')).
\end{aligned}$$

formalizes these requirements. The formula

$$\begin{aligned}\mathrm{Eval}(k,m,s,T,\varphi,I,Y) \Leftrightarrow \forall d < m\,\forall x < s\,\Big(&Y(d,x) \leftrightarrow\\ &(T(d,x,\ulcorner\vee\urcorner) \wedge \exists d' < m\,\exists x' < s\,(\varphi(d,x,d',x') \wedge Y(d',x')))\\ &\vee (T(d,x,\ulcorner\wedge\urcorner) \wedge \forall d' < m\,\forall x' < s\,(\varphi(d,x,d',x') \to Y(d',x')))\\ &\vee (T(d,x,\ulcorner\neg\urcorner) \wedge \exists d' < m\,\exists x' < s\,(\varphi(d,x,d',x') \wedge \neg Y(d',x')))\\ &\vee \exists i < k\,(T(d,x,\ulcorner x_i\urcorner) \wedge I(i))\Big)\end{aligned}$$

states that Y is an evaluation of the circuit described by k,m,s,T,φ on input $I \le k$.

Remark 4.1 Note that any Σ_0^B-formula φ is equivalent in $\overline{VNC}{}^1_*$ to an open formula, e.g., $0 \in \{u < 1 \mid \varphi\}$ (where u is not free in φ). We will prove later (Corollary 4.7) that the same also holds for $\Sigma_0^B(L_{\overline{VNC}{}^1_*})$-formulas.

Theorem 4.2

(i) *If φ is a $\Delta_1^B(VNC^1_*)$-formula without free set variables, then VNC^1_* proves*

$$\mathrm{Circ}(k,|m|,s,T,\varphi) \to \exists! Y \le |m|s\ \mathrm{Eval}(k,|m|,s,T,\varphi,I,Y).$$

(ii) *If φ is an open $L_{\overline{VNC}{}^1_*}$-formula without free set variables, then there exists an $L_{\overline{VNC}{}^1_*}$-term Y such that $\overline{VNC}{}^1_*$ proves*

$$\mathrm{Circ}(k,|m|,s,T,\varphi) \to \mathrm{Eval}(k,|m|,s,T,\varphi,I,Y(\vec{p},k,m,s,T,I)),$$

where $\vec{p}$ are the parameters of φ.

Proof: Uniqueness of Y can be proved by straightforward Σ_0^B-induction, the problem is to show its existence. We will reduce evaluation of the circuit to another circuit in the simplified framework of eval, which can be evaluated using the axioms Δ_1^B-*SCV* or *Open*-$\overline{SCV}$. For the sake of clarity we will use w and friends to denote nodes in the simulated circuit (described by $T(d,w,p)$ and $\varphi(d,w,d',w')$), whereas x,y will refer to nodes in the newly constructed eval-style circuit. We subject the original circuit to the following transformations:

- The input layer of the new circuit will consist of bits $I(j)$ of the original input string I, their negations $\neg I(j)$, and bits $T(d,w,p)$ of T.
- We introduce a dual node $w^\neg$ to each node w in the circuit, in order to allow making the new circuit monotone.
- We replicate each node on all layers to overcome the restriction that each gate may only use nodes of its immediately preceding layer as inputs in the new circuit.

- If w is a node with possible inputs w_0, w_1, we include in the new circuit the following gadgets (suppressing for simplicity the mention of layers, i.e., the first variable d of T):

$$\begin{aligned} w &= \bigvee_{j<k} (T(w, \ulcorner x_j \urcorner) \wedge I(j)) \vee (T(w, \ulcorner \neg \urcorner) \wedge w_0^\neg) \\ &\qquad \vee (T(w, \ulcorner \wedge \urcorner) \wedge w_0 \wedge w_1) \vee (T(w, \ulcorner \vee \urcorner) \wedge (w_0 \vee w_1)), \\ w^\neg &= \bigvee_{j<k} (T(w, \ulcorner x_j \urcorner) \wedge \neg I(j)) \vee (T(w, \ulcorner \neg \urcorner) \wedge w_0) \\ &\qquad \vee (T(w, \ulcorner \vee \urcorner) \wedge w_0^\neg \wedge w_1^\neg) \vee (T(w, \ulcorner \wedge \urcorner) \wedge (w_0^\neg \vee w_1^\neg)). \end{aligned}$$

  More precisely, we put $O(|k|)$ layers to the bottom of the circuit which compute the disjunctions $\bigvee_{j<k}(T(w, \ulcorner x_j \urcorner) \wedge (\neg) I(j))$ arranged in a balanced binary tree, and we replace each node in the original circuit with the constant-size remaining part of its gadget.

- We introduce padding to shift the nodes so that odd layers consist of disjunctions, and even layers of conjunctions.

We proceed with the formal details to verify that we can arrange the result in such a way that the wires of the new circuit are described by a Δ_1^B-formula or an open $L_{\overline{VNC}^1_*}$-formula without set parameters, as required by the axioms.

Our new circuit will have $m' + 1 := 2 + 2|k| + 6|m|$ layers, each of $n' := 2k + (5k + 7)|m|s$ nodes.

Nodes $i(0, j) := j < k$ on each layer represent the input bits $I(j)$, nodes $i(1, j) := k + j$ give $\neg I(j)$, and nodes $t(d, w, p) := 2k + (ds + w)(k + 3) + p$ give $T(d, w, p)$ for $d < |m|$, $w < s$, $p < k + 3$. Nodes

$$r(\varepsilon, d, w, u) := 2k + (k + 3)|m|s + ((\varepsilon|m| + d)s + w)(2k - 1) + u$$

for $\varepsilon < 2$, $d < |m|$, $w < s$, and $u < 2k - 1$ are used to compute $\bigvee_{j<k}(T(d, w, \ulcorner x_j \urcorner) \wedge I^\varepsilon(j))$, where $I^0 = I$, $I^1 = \neg I$. Finally, nodes

$$n(\varepsilon, d, w, u) := 2k + (5k + 1)|m|s + ((\varepsilon|m| + d)s + w)3 + u$$

for $\varepsilon < 2$, $d < |m|$, $w < s$, $u < 3$ represent node w (if $\varepsilon = 0$) or $w^\neg$ (if $\varepsilon = 1$) on layer d in the original circuit, as well as its associated gadget.

The layers are laid out as follows. Layer 0 is the input layer, initialized to

$$I' = \{i(0, j) \mid I(j)\} \cup \{i(1, j) \mid \neg I(j)\} \cup \{t(d, w, p) \mid T(d, w, p)\}.$$

Layer 1 is a copy of layer 0 (as we need conjunctions at the bottom of our new circuit, and odd layers are disjunctions). Layers 2 to $2|k| + 1$ are used to compute $\bigvee_{j<k}(T(d, w, \ulcorner x_j \urcorner) \wedge I^\varepsilon(j))$ into node $r(\varepsilon, d, w, 0)$. On layer 2, we put $T(d, w, \ulcorner x_j \urcorner) \wedge I^\varepsilon(j)$ to node $r(\varepsilon, d, w, k - 1 + j)$. Odd layers 3 to $2|k| + 1$ then consist of disjunctions arranged in a balanced binary tree, where the children of node $r(\varepsilon, d, w, u)$, $u < k - 1$, are $r(\varepsilon, d, w, 2u + 1)$ and $r(\varepsilon, d, w, 2u + 2)$. Even layers 4 to $2|k|$ copy the previous layer. The remaining layers $2|k| + 2$ to $2|k| + 1 + 6|m|$ do the main simulation of the original circuit. Let $l(D,v) = 2|k| + 2 + 6D + v$ for $D < |m|$, $v \leq 5$. Node w on layer d of the original circuit is simulated by node $n(0,d,w,0)$ on layers $l(D,5)$ for all $D \geq d$, and its negation $w^\neg$ is in node $n(1,d,w,0)$. They are also replicated on the next layer $l(D+1,0)$ as $n(\varepsilon,d,w,2)$. Other nodes $n(\varepsilon,d,w,u)$, $u \leq 2$, on layers $l(D,v)$, $v \leq 4$, are parts of the gadget needed to compute w or $w^\neg$.

Let us abbreviate $r = r(0,0,0,0) = 2k + (k+3)|m|s$, $n = n(0,0,0,0) = 2k + (5k+1)|m|s$. For convenience, we define the functions

$$\varepsilon_r(x) = \left\lfloor \frac{x-r}{|m|s(2k-1)} \right\rfloor, \qquad d_r(x) = \left\lfloor \frac{x-r}{s(2k-1)} \right\rfloor \bmod |m|,$$
$$w_r(x) = \left\lfloor \frac{x-r}{2k-1} \right\rfloor \bmod s, \qquad u_r(x) = (x-r) \bmod (2k-1),$$

so that $x = r(\varepsilon_r(x), d_r(x), w_r(x), u_r(x))$ for any $r \leq x < n$. Similarly, we can define functions $\varepsilon_n, d_n, w_n, u_n, D_l, v_l$ so that $x = n(\varepsilon_n(x), d_n(x), w_n(x), u_n(x))$ for any $x \geq n$, and $d' = l(D_l(d'), v_l(d'))$ for any $d' \geq l(0,0)$.

Wires of the new circuit are described by the formula

$$\begin{aligned}\varphi'(d',x,y) \Leftrightarrow{} & (\mathit{Copy}(d',x) \wedge x = y)\\ & \vee (r \leq x < n \wedge 0 < d' \leq 2|k| \wedge \mathit{Disj}(d',x,y))\\ & \vee (x \geq n \wedge d' > 2|k| \wedge \mathit{Gadget}(v_l(d'),x,y))\end{aligned}$$

(recall from the definition of eval that this means that node y on layer d' is an input of node x on layer $d'+1$). The first line takes care of nodes whose value needs to be copied over to the next layer, the second line handles the computation of $R(\varepsilon,d,w) = \bigvee_{j<k}(T(d,w,\ulcorner x_j \urcorner) \wedge I^\varepsilon(j))$ at the bottom of the circuit, and the third line implements the gadgets doing the main simulation.

The disjunction $R(\varepsilon,d,w)$ is computed by initializing nodes $r(\varepsilon,d,w,(k-1)+j)$ on layer 2 to $T(d,w,\ulcorner x_j \urcorner) \wedge I^\varepsilon(j)$, and making node $x = r(\varepsilon,d,w,u)$ on layer $d'+1$ to be the disjunction of nodes $x+u+1, x+u+2$ from layer d', for every even positive $d' \leq 2|k|$:

$$\begin{aligned}\mathit{Disj}(d',x,y) \Leftrightarrow{} & (2 \mid d' \wedge u_r(x) < k-1 \wedge y \in \{x+u_r(x)+1, x+u_r(x)+2\})\\ & \vee (d' = 1 \wedge u_r(x) \geq k-1\\ & \qquad \wedge y \in \{i(\varepsilon_r(x), u_r(x)-(k-1)), t(d_r(x), w_r(x), \ulcorner x_{u_r(x)-(k-1)} \urcorner)\})\end{aligned}$$

Moreover, we need to copy the whole tree from odd layers $d' \leq 2|k|$ to the next (i.e., conjunction) layer, and the initial nodes $r(\varepsilon,d,w,(k-1)+j)$ through layers $d' \leq 2|k|$. We also need to copy over the input bits on all layers of the circuit, and the computed values of $R(\varepsilon,d,w)$ above layer $2|k|$:

$$\begin{aligned}\mathit{Copy}(d',x) \Leftrightarrow{} & x < r\\ & \vee (r \leq x < n \wedge (d' = 0 \vee d' > 2|k| \vee (2 \nmid d' \wedge d' \neq 1) \vee (2 \mid d' \wedge u_r(x) \geq k-1)))\end{aligned}$$

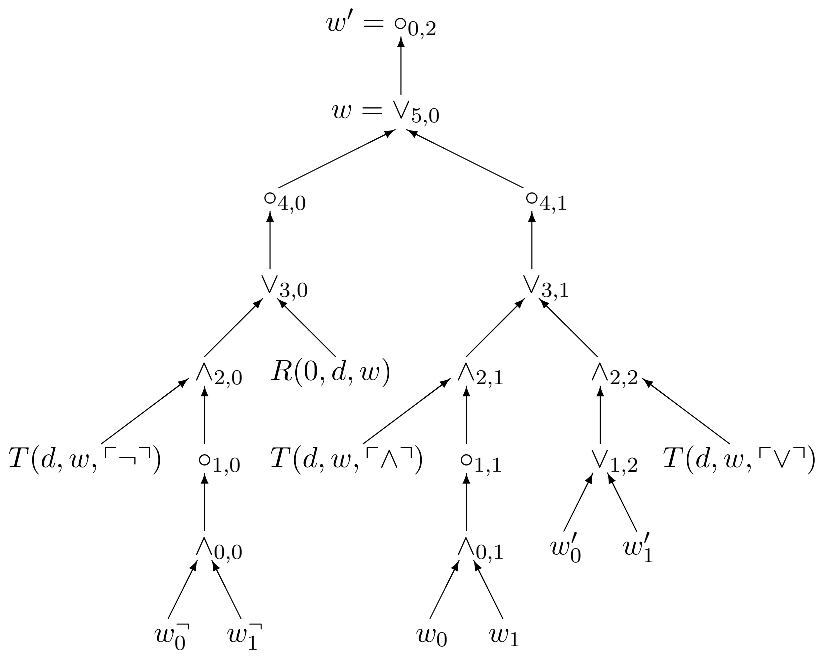

Figure 4.1: Simulation of one node of the original circuit

The main part of the simulating circuit is given by

$$\begin{aligned}
\mathit{Gadget}(v,x,y) \Leftrightarrow{} & \langle y,v,u_n(x)\rangle = \langle r(\varepsilon_n(x),d_n(x),w_n(x),0),2,0\rangle \\
& \vee \langle y,v,u_n(x)\rangle = \langle t(d_n(x),w_n(x),\ulcorner\neg\urcorner),1,0\rangle \\
& \vee \langle y,v,u_n(x)\rangle = \langle t(d_n(x),w_n(x),\ulcorner\wedge\urcorner),1,1+\varepsilon_n(x)\rangle \\
& \vee \langle y,v,u_n(x)\rangle = \langle t(d_n(x),w_n(x),\ulcorner\vee\urcorner),1,2-\varepsilon_n(x)\rangle \\
& \vee \langle y-x,v,u_n(x)\rangle \in \{\langle 0,0,0\rangle,\langle 0,0,1\rangle,\langle 0,1,0\rangle,\langle 0,1,1\rangle,\langle 0,1,2\rangle, \\
& \qquad \langle 0,2,0\rangle,\langle 0,2,1\rangle,\langle 1,2,1\rangle,\langle 0,3,0\rangle,\langle 0,3,1\rangle, \\
& \qquad \langle 0,4,0\rangle,\langle 1,4,0\rangle,\langle -2,5,2\rangle\} \\
& \vee (\mathit{Edge}(v,x,y) \wedge \varphi(d_n(x),w_n(x),d_n(y),w_n(y)))
\end{aligned}$$

$$\begin{aligned}
\mathit{Edge}(v,x,y) \Leftrightarrow{} & (\langle v,u_n(x),u_n(y)\rangle = \langle 5,0,0\rangle \wedge \varepsilon_n(x) \neq \varepsilon_n(y)) \\
& \vee (\langle v,u_n(x),u_n(y)\rangle \in \{\langle 5,1,0\rangle,\langle 0,2,2\rangle\} \wedge \varepsilon_n(x) = \varepsilon_n(y))
\end{aligned}$$

The layout of the gadget is explained in Figure 4.1 for the case $\varepsilon_n(x) = 0$. Nodes $x = n(0,d,w,u)$ on layers $l(D,v)$ are labelled with connectives subscripted with v, u, where $\circ$ stands for one-argument $\wedge$ or $\vee$ employed to satisfy the restriction that odd layers are disjunctions and even layers are conjunctions. Plain w marks $n(0,d,w,0)$ on layer $l(D,5)$, and w' its copy $n(0,d,w,2)$ on layer $l(D+1,0)$. Let the children of node w on layer d in the original circuit be nodes w_0, w_1 on layers d_0, d_1 (one or both of them could be missing). The labels w_0, w_1, $w^\neg_0$, $w^\neg_1$, w'_0, w'_1 mark nodes $n(0,d_0,w_0,0)$, $n(0,d_1,w_1,0)$, $n(1,d_0,w_0,0)$, $n(1,d_1,w_1,0)$ on layer $l(D-1,5)$ and nodes $n(0,d_0,w_0,2)$, $n(0,d_1,w_1,2)$ on layer $l(D,0)$, respectively. Note that the condition

$\mathrm{Circ}(k, |m|, s, T, \varphi)$ ensures that w has only one child if $T(d, w, \ulcorner\neg\urcorner)$.

Notice that integer division and mod are Σ^B_0-definable. It is thus easy to see that $\varphi' \in \Delta^B_1(\mathit{VNC}^1_*)$ if $\varphi \in \Delta^B_1(\mathit{VNC}^1_*)$, and, using Remark 4.1, that φ' is equivalent to an open $L_{\overline{\mathit{VNC}^1_*}}$-formula if φ is. By Δ^B_1-*SCV* or *Open*-$\overline{\mathit{SCV}}$, there exists Y' such that $\mathrm{eval}(n', m', \varphi', I', Y')$. It is tedious, but completely straightforward, to verify that parts of Y' correspond to an evaluation of the original circuit as described above, hence $\mathrm{Eval}(k, |m|, s, T, \varphi, I, Y)$, where

$$Y = \{\langle d, x\rangle \mid Y'(m', n(0, d, x, 0))\}.$$

In the case of $\overline{\mathit{VNC}^1_*}$, we can compute I' from I and T by a comprehension function symbol, compute Y' using the $Y_{\varphi'}$ function, and compute Y from Y' by another comprehension function, hence Y is given by a term in the original data. $\square$

Corollary 4.3 *VNC^1_* and $\overline{\mathit{VNC}^1_*}$ contain VNC^1.* $\square$

Definition 4.4 Let Γ be a set of formulas. A c-ary set function $F(X_0, \dots, X_{c-1})$ is *computable by a family of Γ-definable shallow circuits* (*computable by Γ-circuits* for short) if there are L_0-terms $s(n)$, $m(n)$, and $o(n)$, a Σ^B_0-formula $\tau(n, d, x, p)$, and a Γ-formula $\varphi(n, d, x, d', x')$, such that

- $s(n) \geq cn$, $s(n) \geq o(n)$, $m(n) > 0$,
- $\mathrm{Circ}(cn, |m(n)|, s(n), T(n), \varphi)$, where $T(n) = \{(s(n)d + x)(cn + 3) + p \mid \tau(n, d, x, p)\}$,
- if $\vec{X}$ are sets such that $|X_i| \leq n$, $I = \{in + u \mid i < c, u \in X_i\}$, and

  $$\mathrm{Eval}(cn, |m(n)|, s(n), T(n), \varphi, I, Y),$$

  then

  $$F(\vec{X}) = \{u < o(n) \mid Y(|m(n)| - 1, u)\}. \qquad (1)$$

A function $F(\vec{u}, \vec{X})$ or $f(\vec{u}, \vec{X})$ with number inputs and/or output is computable by Γ-circuits, if the same holds for the set function $F'(\vec{U}, \vec{X})$ which we obtain by representing every number x by the set $\{u \mid u < x\}$. A predicate $\psi(\vec{u}, \vec{X})$ is computable by Γ-circuits if its characteristic function

$$\chi_\psi(\vec{u}, \vec{X}) = \begin{cases} \{0\} & \text{if } \psi(\vec{u}, \vec{X}), \\ \varnothing & \text{if } \neg\psi(\vec{u}, \vec{X}) \end{cases}$$

is. In other words, if we can fix $o(n) = 1$ in the above definition, and replace (1) with

$$\psi(\vec{u}, \vec{X}) \leftrightarrow Y(|m(n)| - 1, 0).$$

The next lemma is a key technical result needed to show various properties of VNC^1_* and $\overline{\mathit{VNC}^1_*}$, e.g., that $\overline{\mathit{VNC}^1_*}$ is a conservative extension of VNC^1_*.

Lemma 4.5 *Let $\alpha(\vec{X},\vec{x})$ be a $L_{\overline{VNC}^1_*}$-term, or a $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula. Then α is provably in $\overline{VNC}^1_*$ computable by $Open(L_{\overline{VNC}^1_*})$-circuits, and provably in VNC^1_* computable by $\Delta^B_1(VNC^1_*)$-circuits in such a way that VNC^1_* proves the defining axiom of α.*

Moreover, the graph of α or of its characteristic function is definable in VNC^1_* by a Σ^B_1-formula $\exists Y \leq t\,\vartheta(\vec{X},\vec{x},\varepsilon,Y)$ with $\vartheta \in \Sigma^B_0$, and provably in VNC^1_*, we can compute some Y satisfying the formula from $\vec{x},\vec{X}$ by $\Delta^B_1(VNC^1_*)$-circuits.

Proof: We proceed by induction on the complexity of α (defined in such a way that the complexity of C_φ and Y_φ is larger than that of φ, and in the case of Y_φ, also φ^*).We will show two cases, and leave the rest to the reader.

Let α be the formula $\exists x_c \leq t(\vec{X},\vec{x})\,\beta(\vec{X},\vec{x},x_c)$, and fix a polynomial $q(n)$ such that $t(\vec{X},\vec{x}) < q(n)$ whenever $|\vec{X}|,\vec{x} \leq n$. By the induction hypothesis, we can compute the formula $\alpha' = x_c \leq t(\vec{X},\vec{x}) \wedge \beta(\vec{X},\vec{x},x_c)$ by $Open(\overline{VNC}^1_*)$-circuits or $\Delta^B_1(VNC^1_*)$-circuits described by s', m', τ', and φ'. We construct circuits for α by taking $q(n)$ copies of the circuit for α', fixing the value of x_c to the representation of i in the ith copy, and computing the disjunction of the outputs (arranged in a binary tree, as in the proof of Theorem 4.2). To be exact, we put $s(n) = q(n)s'(n)$ (assuming $s'(n) \geq 2$), $m(n) = 4m'(n)q(n)$ (so that $|m(n)| \geq |m'(n)| + |q(n)| + 1$),

$$\begin{aligned}\tau(n,d,x,p) \Leftrightarrow{} & (d \geq |m'| \wedge p = \ulcorner\vee\urcorner)\\ & \vee (d < |m'| \wedge \tau'(n,d,x \bmod s',p) \wedge \neg\exists j < n\, p = \ulcorner x_{cn+j}\urcorner)\\ & \vee (d < |m'| \wedge \exists j < n\,(\tau'(n,d,x \bmod s',\ulcorner x_{cn+j}\urcorner)\\ & \qquad \wedge ((j < \lfloor x/s'\rfloor \wedge p = \ulcorner\wedge\urcorner) \vee (j \geq \lfloor x/s'\rfloor \wedge p = \ulcorner\vee\urcorner)))),\end{aligned}$$

$$\begin{aligned}\varphi(n,d,x,d',x') \Leftrightarrow{} & (d < |m'| \wedge \varphi'(n,d,x \bmod s',d',x' \bmod s') \wedge \lfloor x/s'\rfloor = \lfloor x'/s'\rfloor)\\ & \vee (d = |m'| \wedge d' = d-1 \wedge q-1 \leq x < 2q-1 \wedge x' = (x-(q-1))s')\\ & \vee (d > |m'| \wedge d' = d-1 \wedge x \geq q-1 \wedge x' = x)\\ & \vee (d > |m'| \wedge d' = d-1 \wedge x < q-1 \wedge 1 \leq x'-2x \leq 2),\end{aligned}$$

where we write m', s', q for $m'(n)$, $s'(n)$, $q(n)$. Note that x_{cn+j} represents the jth bit in x_c. The definition of τ thus ensures that each input node corresponding to x_c in the original circuit for α' is replaced with a disjunction or conjunction gate in the new circuit; since the gate has no inputs, it actually computes the constant $\bot$ or $\top$, respectively, accomplishing the above mentioned replacement of x_c with the representation of i. Clearly, τ is Σ^B_0, φ is $Open(L_{\overline{VNC}^1_*})$ or $\Delta^B_1(VNC^1_*)$ as appropriate, and it is easy to see that the circuit defined by s, m, τ, φ computes α.

Let $\exists Y \leq u\,\vartheta(\vec{X},\vec{x},x_c,\varepsilon,Y)$ be a Σ^B_1-definition of the graph $\chi_{\alpha'}(\vec{X},\vec{x},x_c) = \varepsilon$ of the characteristic function of α', such that Y is computable from $\vec{X},\vec{x},x_c$ by $\Delta^B_1(VNC^1_*)$-circuits. Consider the Σ^B_1-formula

$$\begin{aligned}(2)\quad & |\varepsilon| \leq 1 \wedge \exists Z \leq uq(n)\,\big[\forall x_c < q(n)\,\big(\vartheta(\vec{X},\vec{x},x_c,\varnothing,Z^{[x_c]}) \vee \vartheta(\vec{X},\vec{x},x_c,\{0\},Z^{[x_c]})\big)\\ & \qquad \wedge \big(0 \in \varepsilon \leftrightarrow \exists x_c < q(n)\,\vartheta(\vec{X},\vec{x},x_c,\{0\},Z^{[x_c]})\big)\big],\end{aligned}$$

where $n = \sum_i |X_i| + \sum_i x_i$, and $Z^{[x]}$ denotes $\{y < u \mid xu + y \in Z\}$. We take $q(n)$ parallel copies of the circuit computing Y, and wire the x_c inputs in the ith copy to the representation of i, as above in the construction of the circuit for α. The resulting circuit computes Z satisfying

$$\forall x_c < q(n)\ (\vartheta(\vec{X}, \vec{x}, x_c, \varnothing, Z^{[x_c]}) \vee \vartheta(\vec{X}, \vec{x}, x_c, \{0\}, Z^{[x_c]}))$$

from $\vec{X}, \vec{x}$. Given Z, it is easy to see that (2) is equivalent to $\chi_\alpha(\vec{X}, \vec{x}) = \varepsilon$.

Let us turn to the case $\alpha = Y_\psi(\vec{p}(\vec{X}, \vec{x}), s(\vec{X}, \vec{x}), m(\vec{X}, \vec{x}), I(\vec{X}, \vec{x}))$, where $\psi(\vec{p}, d, x, y)$ is an open $\overline{VNC}^1_*$-formula. By the induction hypothesis, we can compute the terms $\vec{p}$, s, m, and I by suitable circuits. Let $q(n)$ be a polynomial such that $\vec{p}(\vec{X}, \vec{x}), s(\vec{X}, \vec{x}), m(\vec{X}, \vec{x}), |I(\vec{X}, \vec{x})| < q(n)$ whenever $|\vec{X}|, \vec{x} \le n$.

As with other compound terms, the expected idea of how to evaluate α by a circuit would be to take circuits evaluating $\vec{p}(\vec{X}, \vec{x})$, $s(\vec{X}, \vec{x})$, $m(\vec{X}, \vec{x})$, $I(\vec{X}, \vec{x})$, and plug them into a circuit evaluating Y_ψ. However, we cannot do this directly: computing Y_ψ amounts to simulation of the circuit C described by ψ, and the parameters $\vec{p}, s, m$ are not *inputs* of C, they actually affect the *shape* of C ($|m|$ is its depth, s the size of each layer, and $\vec{p}$ are free variables of the formula ψ describing edges of C). In order to evaluate C, we must first fix these parameters to some constants. We cannot quite do this either, because the terms $m(\vec{X}, \vec{x})$ etc. are not really constant. However, we have a polynomial bound $q(n)$ on their possible value, hence what we can do is to evaluate in parallel polynomially many variants of C, one for each choice of $\vec{p}, s, m < q(n)$. (Note that here we use essentially that ψ has no set free variables: if we had a set parameter P instead of $\vec{p}$, we would have to evaluate C for exponentially many possible choices of P.) Then we have to select the real result among results of all these circuits: this is done using selector functions $h_{\vec{p},s,m}(\vec{X}, \vec{x})$ indexed by $\vec{p}, s, m < q(n)$, whose value is 1 iff $\vec{p}, s, m$ agree with the real values of $\vec{p}(\vec{X}, \vec{x})$, $s(\vec{X}, \vec{x})$, $m(\vec{X}, \vec{x})$.

Explicitly, we construct circuits computing α as follows:

- We compute $s(\vec{X}, \vec{x}), m(\vec{X}, \vec{x}), I(\vec{X}, \vec{x}), \vec{p}(\vec{X}, \vec{x})$ using their respective circuits. We denote the jth bit of the result by s_j, m_j, i_j, p^r_j (we index elements of the $\vec{p}$ sequence by superscripts, to avoid clashes with bit subscripts).
- For every $\vec{p}, s, m < q(n)$, we evaluate in parallel the eval-style circuit defined by s, $|m|$, and $\psi^*(\vec{p}, \cdot, \cdot, \cdot)$ on input I. That is, we take the circuit with $|m| + 1$ layers, each of size s. The bottom layer is initialized to the first s bits i_j, and the other layers are alternating disjunctions and conjunctions, where yth node on dth layer is an input to xth node on $(d + 1)$st layer iff $\psi^*(\vec{p}, d, x, y)$. We denote the value of the xth node on dth layer by $v_{\vec{p},s,m,d,x}$.
- For each $\vec{p}, s, m < q(n)$, we compute in parallel the selector $h_{\vec{p},s,m}$ which states that $\bigwedge_r (p^r(\vec{X}, \vec{x}) = p^r) \wedge s(\vec{X}, \vec{x}) = s \wedge m(\vec{X}, \vec{x}) = m$. This can be done using

$$h_{\vec{p},s,m} = \bigwedge_r (p^r_{p^r-1} \wedge \neg p^r_{p^r}) \wedge s_{s-1} \wedge \neg s_s \wedge m_{m-1} \wedge \neg m_m,$$

  where we omit the conjuncts with index -1 (i.e., treat them as $\top$).

- We compute in parallel the output bits

$$o_{d,x} = \bigvee_{\vec{p},s,m<q(n)} (h_{\vec{p},s,m} \wedge v_{\vec{p},s,m,d,x}).$$

We spare the reader the formal definitions of the τ and φ formulas describing the circuit, and leave it to their imagination to verify that τ is Σ^B_0, and φ is a Boolean combination of Σ^B_0-formulas and formulas obtained from ψ^* by substituting Σ^B_0-definable functions like division with remainder for some of its free variables. By the induction hypothesis, ψ^* is equivalent to a $\Delta^B_1(VNC^1_*)$- and $Open(L_{\overline{VNC^1_*}})$-formula, therefore so is φ. It is easy to see that the circuit indeed computes α.

We also have to describe the graph of $\alpha(\vec{X},\vec{x})$ in VNC^1_* by a Σ^B_1-formula such that witnesses to the existential second-order quantifier can be computed by $\Delta^B_1(VNC^1_*)$-circuits. (Note that now we do not have to compute α itself, its value Y is given to us.) This is easy: it suffices to take as a witness of α the sequence of witnesses of $\vec{p}(\vec{X},\vec{x})$, $s(\vec{X},\vec{x})$, $m(\vec{X},\vec{x})$, $I(\vec{X},\vec{x})$, the value of $I(\vec{X},\vec{x})$, and witnesses of $\psi(\vec{p},d,x,y)$ for each d,x,y needed to describe the circuit evaluated by Y_ψ.

Formally, let ϑ be Σ^B_0-formula such that the graph $\chi_\psi(\vec{p},d,x,y) = \varepsilon$ of the characteristic function of ψ is equivalent to $\exists W \leq t\, \vartheta(\vec{p},d,x,y,\varepsilon,W)$, and W is computable by $\Delta^B_1(VNC^1_*)$-circuits by the induction hypothesis. Consider the formula

$$\begin{aligned}(3)\qquad & \exists Z \leq (q(n))^3 t\, \exists I, \vec{p}, s, m \leq q(n)\, \Big(\mathrm{eval}(s,|m|,\xi,I,Y)\\ & \wedge \bigwedge_r p^r(\vec{X},\vec{x}) = p^r \wedge s(\vec{X},\vec{x}) = s \wedge m(\vec{X},\vec{x}) = m \wedge I(\vec{X},\vec{x}) = I\\ & \wedge \forall d < |m|\, \forall x,y < s\, (\vartheta(\vec{p},d,x,y,\varnothing,Z^{[d,x,y]}) \vee \vartheta(\vec{p},d,x,y,\{0\},Z^{[d,x,y]}))\Big),\end{aligned}$$

where

$$\xi(d,x,y) \Leftrightarrow \vartheta(\vec{p},d,x,y,\{0\},Z^{[d,x,y]}),$$

$n = \sum_i |X_i| + \sum_i x_i$, and $Z^{[d,x,y]}$ denotes $\{u < t \mid ((dq(n)+x)q(n)+y)t+u \in Z\}$. If we replace $p^r(\vec{X},\vec{x})$, $s(\vec{X},\vec{x})$, $m(\vec{X},\vec{x})$, and $I(\vec{X},\vec{x})$ with their Σ^B_1-definitions which exist by the induction hypothesis and prenex the second-order existential quantifiers, we obtain a Σ^B_1-formula, which we can further normalize to the form with only one second-order quantifier using a pairing function. Given $\vec{X},\vec{x}$, we can compute a witness to this formula by $\Delta^B_1(VNC^1_*)$-circuits as follows. We compute (using the induction hypothesis) the values of $\vec{p}$, s, m, and I, and witnesses to the second-order quantifiers used in their graphs. Then we take the circuit computing W, and evaluate in parallel its $q(n)^3$ copies for all fixed values $d,x,y < q(n)$ to obtain a Z such that

$$\forall d < |m|\, \forall x,y < s\, (\vartheta(\vec{p},d,x,y,\varnothing,Z^{[d,x,y]}) \vee \vartheta(\vec{p},d,x,y,\{0\},Z^{[d,x,y]})).$$

Given such Z, we have $\xi(d,x,y) \leftrightarrow \psi(\vec{p},d,x,y)$, hence $\mathrm{eval}(s,|m|,\xi,I,Y)$ is valid for $Y = Y_\psi(\vec{p},s,m,I)$, and only for this Y. Thus, (3) defines the graph of α, and witnesses for its second-order quantifiers can be computed by $\Delta^B_1(VNC^1_*)$-circuits. □

Corollary 4.6 *$\overline{VNC}^1_*$ is contained in an extension of VNC^1_* by Σ^B_1-definitions. In particular, $\overline{VNC}^1_*$ is conservative over VNC^1_*.* □

Corollary 4.7 *Every $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula is in $\overline{VNC}^1_*$ equivalent to an open formula.* □

Corollary 4.8 *$\overline{VNC}^1_*$ proves $\Sigma^B_0(L_{\overline{VNC}^1_*})$-COMP, and $\Sigma^B_0(L_{\overline{VNC}^1_*})$-IND. Moreover, there are comprehension terms $F(a,\vec{x},\vec{X}) = \{u < a \mid \varphi(u,\vec{x},\vec{X})\}$ for $\Sigma^B_0(\overline{VNC}^1_*)$-formulas φ.*

Proof: Induction follows from comprehension. Let $\varphi(u,\vec{x},\vec{X})$ be a $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula, and let $n = a + \sum_i x_i + \sum_i |X_i|$. By Lemma 4.5, φ is computable by an $Open(L_{\overline{VNC}^1_*})$-circuit on inputs of size n. We take a parallel copies of the circuit as in the proof of Lemma 4.5, and wire the output of the ith circuit to the ith new output bit. We evaluate the circuit on the input which sets $\vec{x}$ and $\vec{X}$ in each copy to the value of the respective parameters, and sets u to the representation of i in the ith copy. Then the output of the new circuit is $\{u < a \mid \varphi\}$. The circuit is described by an open formula, hence its value is computable by an $L_{\overline{VNC}^1_*}$-term using Theorem 4.2. □

Theorem 4.9 *$\overline{VNC}^1_*$ is an open theory.*

Proof: For any $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula φ, let $\overline{\varphi}$ be an open formula equivalent to φ in $\overline{VNC}^1_*$ by Corollary 4.7. We may assume that $\varphi = \overline{\varphi}$ if φ is already open. Let T be the set of formulas which contains

$$\overline{\varphi \vee \psi} \leftrightarrow \overline{\varphi} \vee \overline{\psi}$$

and similarly for other Boolean connectives, and the formulas

$$\overline{\varphi}(x) \wedge x \le t \to \overline{\exists x \le t\, \varphi(x)},$$
$$\overline{\exists x \le t\, \varphi(x)} \to \overline{\varphi}(|S|) \wedge |S| \le t,$$

where S is a term (with the same free variables as $\exists x \le t\, \varphi$) such that $\overline{VNC}^1_*$ proves that $S = \{x < t \mid \varphi(x+1)\}$ (such a term exists by Corollary 4.8).

Clearly, T is an open subtheory of $\overline{VNC}^1_*$, and every $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula is in T equivalent to an open formula. As $\overline{VNC}^1_*$ is $\Sigma^B_0(L_{\overline{VNC}^1_*})$-axiomatized, it is equivalent to an open extension of T. □

Theorem 4.10 *If $\overline{VNC}^1_*$ proves $\exists Y\, \varphi(\vec{x},\vec{X},Y)$, where φ is a Σ^1_1-formula, then there exists an $L_{\overline{VNC}^1_*}$-term F such that $\overline{VNC}^1_*$ proves $\varphi(\vec{x},\vec{X},F(\vec{x},\vec{X}))$.*

Proof: Write $\varphi = \exists \vec{Z}\, \vartheta(\vec{x},\vec{X},Y,\vec{Z})$ with $\vartheta \in \Sigma^B_0(L_{\overline{VNC}^1_*})$. By Corollary 4.7, ϑ is equivalent to an open formula. By Theorem 4.9 and Herbrand's theorem, there exist terms F_r, G^j_r such that $\overline{VNC}^1_*$ proves

$$\vartheta(\vec{x},\vec{X},F_0(\vec{x},\vec{X}),\vec{G}_0(\vec{x},\vec{X})) \vee \cdots \vee \vartheta(\vec{x},\vec{X},F_c(\vec{x},\vec{X}),\vec{G}_c(\vec{x},\vec{X}))$$

for some c. Put

$$\alpha_r \Leftrightarrow \vartheta(\vec{x},\vec{X},F_r(\vec{x},\vec{X}),\vec{G}_r(\vec{x},\vec{X})) \wedge \bigwedge_{s<r} \neg\vartheta(\vec{x},\vec{X},F_s(\vec{x},\vec{X}),\vec{G}_s(\vec{x},\vec{X})),$$

and let p be a polynomial such that $|F_r|, |G^j_r| \le p(\vec{x}, |\vec{X}|)$. By Corollary 4.8, there are terms F and G^j such that $\overline{VNC}^1_*$ proves

$$F(\vec{x}, \vec{X}) = \Bigl\{u < p(\vec{x}, |\vec{X}|) \Bigm| \bigvee_r (\alpha_r \wedge u \in F_r(\vec{x}, \vec{X}))\Bigr\},$$
$$G^j(\vec{x}, \vec{X}) = \Bigl\{u < p(\vec{x}, |\vec{X}|) \Bigm| \bigvee_r (\alpha_r \wedge u \in G^j_r(\vec{x}, \vec{X}))\Bigr\},$$

Clearly, $\overline{VNC}^1_*$ proves

$$\bigvee_r \alpha_r,$$
$$\alpha_r \to F(\vec{x}, \vec{X}) = F_r(\vec{x}, \vec{X}),$$
$$\alpha_r \to G^j(\vec{x}, \vec{X}) = G^j_r(\vec{x}, \vec{X}),$$

hence also

$$\vartheta(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}), \vec{G}(\vec{x}, \vec{X})),$$

which implies $\varphi(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$. □

Corollary 4.11 *Every $\Delta^B_1(\overline{VNC}^1_*)$-formula is in $\overline{VNC}^1_*$ equivalent to an open formula.*

Proof: Given $\varphi \in \Delta^B_1(\overline{VNC}^1_*)$ (or even $\Delta^1_1(\overline{VNC}^1_*)$), we apply Theorem 4.10 to the formula $\exists Y\,(0 \in Y \leftrightarrow \varphi)$. We obtain a term F such that the open formula $0 \in F(\vec{x}, \vec{X})$ is equivalent to φ. □

Corollary 4.12 *$\overline{VNC}^1_*$ contains VNC^1_*, thus VNC^1_* is the L_0-fragment of $\overline{VNC}^1_*$.*

Proof: By Corollary 4.11, $\overline{VNC}^1_*$ is closed under Δ^B_1-SCV. □

Corollary 4.13 *VNC^1_* is Σ^B_1-axiomatizable.*

Proof: We can take axioms stating the totality of Σ^B_1-definitions of $L_{\overline{VNC}^1_*}$-functions by Corollary 4.6, and a translation of an open axiom system for $\overline{VNC}^1_*$ to L_0, which exists by Theorem 4.9. The resulting theory exhausts VNC^1_* by Corollary 4.12.

Alternatively, assume that a Σ^B_1-formula $\varphi = \exists \vec{Z} \le t\, \vartheta(\vec{p}, d, x, y, \vec{Z})$ is equivalent to a Π^B_1-formula $\neg\exists \vec{Z} \le t\, \lambda(\vec{p}, d, x, y, \vec{Z})$ in VNC^1_* (and therefore in $\overline{VNC}^1_*$). Then φ is equivalent to an open $\overline{VNC}^1_*$-formula by Corollary 4.11, hence by the proof of Lemma 4.5, VNC^1_* proves

$$\begin{aligned}(4)\quad \exists Y \le (|m|+1)n\, \exists Z \le |m|n^2 t\, \bigl[&\mathrm{eval}(n, |m|, \xi, I, Y) \\ &\wedge \forall d < |m|\, \forall x, y < n\, \bigl(\vartheta(\vec{p}, d, x, y, Z^{[d,x,y]}) \vee \lambda(\vec{p}, d, x, y, Z^{[d,x,y]})\bigr)\bigr],\end{aligned}$$

where

$$\xi(d, x, y) \Leftrightarrow \vartheta(\vec{p}, d, x, y, Z^{[d,x,y]}).$$

Clearly, (4) is a Σ^B_1-formula, and it implies

$$\exists Y \le (|m|+1)n\ \mathrm{eval}(n, |m|, \varphi, I, Y)$$

over V^0, hence we can axiomatize VNC^1_* by (4) for all such φ over V^0. □

Theorem 4.14 *VNC^1_* is contained in VL.*

Proof: We need to show that VL is closed under the Δ^B_1-SCV rule. If $\varphi \in \Delta^B_1(VL)$, then φ is, provably in VL, log-space computable, hence VL proves comprehension for φ (see [68]). It thus suffices to show that VL proves

$$\forall n, m, E, I\, \exists Y \text{ eval}(n, |m|, E, I, Y).$$

We will prove this by formalizing in VL the standard log-space algorithm for evaluation of log-depth circuits.

Fix $d_0 \le |m|$ and $x_0 < n$, we will describe how to evaluate the node x_0 on layer d_0 of the circuit. The idea of the algorithm is to make a depth-first traversal of the circuit, evaluating the nodes along the way, and taking short cuts when we have enough information to determine the value of a particular node. The states of the algorithm will be described by numbers below some a, and we will define the graph of the transition function $F\colon a \to a$ of the algorithm; computation of the algorithm will then be simulated by iterating F using the VL axiom. The states of the algorithm will have the following form, with $\langle\downarrow, 1, x_0\rangle$ being the initial state:

(i) $\langle\circ, b\rangle$, where $b < 2$. This is the final state, b is the result of the computation.

(ii) $\langle\downarrow, s, x\rangle$, where $x < n$, $0 < s < 2^{|m|+1}$. We have just descended one layer down the circuit (or we are starting the search in node x_0 on layer d_0). The path from $\langle d_0, x_0\rangle$ to the current node is recorded by a sequence encoded by s: if the binary expansion of s is $1s_0 \dots s_{k-1}$, then s_i is 0 (1) if we have descended to the left-most (right-most, resp.) child at the ith branching (i.e., at ith layer below the top). The current node is node x on layer $d_0 - k = d_0 - |s| + 1$.

(iii) $\langle\uparrow, s, b, t, i, x\rangle$, where $0 < s < 2^{|m|+1}$, $b < 2$, $t < 2$, $i < |s|$, $x < n$. We have ascended up from a child node. Again, s describes the path to the current node. b is the computed value of the child, and t is 0 if the child was the left-most child, or 1 otherwise. In this situation, we do not know the number of the node we are in, as it cannot be uniquely inferred from the child node; we can however recover it from the sequence s. We compute the node number in a loop with $|s| - 1$ steps, we use i as the loop counter, and x to keep track of the node number. We will obtain the current node number in x when $i = |s| - 1$.

We pick sufficiently large a so that all states above are encoded by a number below a. The function F is Σ^B_0-defined by

$$F(\langle\circ, b\rangle) = \langle\circ, b\rangle$$

$$F(\langle\downarrow, s, x\rangle) = \begin{cases} \langle\circ, I(x)\rangle & d_0 = 0 \\ \langle\uparrow, \lfloor s/2\rfloor, I(x), s \bmod 2, 0, x_0\rangle & |s| - 1 = d_0 > 0 \\ \langle\uparrow, \lfloor s/2\rfloor, (d_0 - |s|) \bmod 2, s \bmod 2, 0, x_0\rangle & |s| - 1 < d_0, \\ & \forall y < n\, \neg E(d_0 - |s|, x, y) \\ \langle\downarrow, 2s, l(d_0 - |s| + 1, x)\rangle & |s| - 1 < d_0, \\ & \exists y < n\, E(d_0 - |s|, x, y) \end{cases}$$

$$F(\langle\uparrow, s, b, t, i, x\rangle) = \begin{cases} \langle\uparrow, s, b, t, i+1, l(d_0 - i, x)\rangle & i < |s| - 1, s_i = 0 \\ \langle\uparrow, s, b, t, i+1, r(d_0 - i, x)\rangle & i < |s| - 1, s_i = 1 \\ \langle\circ, b\rangle & i = |s| - 1 = 0, \\ & t = 1 \text{ or } d_0 - |s| \not\equiv b \pmod 2 \\ \langle\uparrow, \lfloor s/2 \rfloor, b, s \bmod 2, 0, x_0\rangle & i = |s| - 1 > 0, \\ & t = 1 \text{ or } d_0 - |s| \not\equiv b \pmod 2 \\ \langle\downarrow, 2s+1, r(d_0 - |s| + 1, x)\rangle & i = |s| - 1 > 0, \\ & t = 0, d_0 - |s| \equiv b \pmod 2 \end{cases}$$

where

$$l(d, x) = \min\{y < n \mid E(d-1, x, y)\},$$
$$r(d, x) = \max\{y < n \mid E(d-1, x, y)\},$$

and F is defined arbitrarily on other numbers below a. By the VL axiom, there exists a sequence P such that $(P)_0 = \langle\downarrow, 1, x_0\rangle$ and $(P)_{v+1} = F((P)_v)$ for all $v < a$. We leave to the reader to verify that P determines a correct partial evaluation of the original circuit, in particular, $(P)_a = \langle\circ, b\rangle$, where b is the value of node x_0 on layer d_0.

In order to evaluate the whole circuit at once, we take a copy of the above algorithm for every $d_0 \le |m|$ and $x_0 < n$, and "concatenate" them in such a way that a final state $\langle\circ, b\rangle$ of node $\langle d_0, x_0\rangle$ is followed by the initial state $\langle\downarrow, 1, x'_0\rangle$ of the next node $\langle d'_0, x'_0\rangle$. We leave the details to the reader. □

Definition 4.15 A function $F(\vec{x}, \vec{X})$ is a *provably total computable function* of a theory $T \supseteq V^0$, if there exists a Σ^1_1-formula $\varphi(\vec{x}, \vec{X}, Y)$ which defines the graph of F in the standard model such that

$$T \vdash \exists! Y\, \varphi(\vec{x}, \vec{X}, Y).$$

Complexity classes like NC^1 can be adapted to the second-order setting in a straightforward way: we represent sets by binary strings, and we write numbers in unary (i.e., as in Definition 4.4).

Corollary 4.16 *The provably total computable functions of VNC^1_* and $\overline{VNC}^1_*$ include the uniform* NC^1*-functions, and are contained in the* L*-uniform* NC^1*-functions.*

Proof: Uniform NC^1-functions are provably total already in VNC^1. On the other hand, assume that $F(\vec{x}, \vec{X})$ is provably total in $\overline{VNC}^1_*$. By Theorem 4.10, F is definable by an $L_{\overline{VNC}^1_*}$-term, hence it is computable by $\Delta^B_1(VNC^1_*)$-circuits using Lemma 4.5. As $VNC^1_* \subseteq VL$, the formula φ defining the circuits as in Definition 4.4 must be in $\Delta^B_1(VL) = \mathrm{L}$. The description of the circuits by the formulas φ and τ is a notational variant of the direct connection language, hence F is in L-uniform FNC^1. □

Remark 4.17 We can describe the provably total functions of VNC^1_* exactly, but the characterization does not lead to a transparent previously studied class. Let NC^1_* be the smallest class $X \supseteq \mathrm{AC}^0$ such that X-uniform NC^1 is included in X, and let FNC^1_* denote the class of functions whose output has length polynomially bounded in the length of input, and whose bit-graph is in NC^1_*. (We could stratify this definition as follows: let $\mathrm{NC}^1_0 = \mathrm{AC}^0$, let NC^1_{k+1} consist of NC^1_k-uniform NC^1, and put $\mathrm{NC}^1_* = \bigcup_{k\in\omega} \mathrm{NC}^1_k$. Notice that U_D-uniform NC^1 is included in NC^1_1.) Then it is possible to show that the provably total computable functions of VNC^1_* (i.e., functions defined by an $L_{\overline{VNC}^1_*}$-term) are exactly the FNC^1_*-functions, and $\Delta^1_1(VNC^1_*)$-predicates (i.e., predicates definable by an open $L_{\overline{VNC}^1_*}$-formula) are exactly the NC^1_*-languages.

The theory V^i extended by the *axiom of choice*

$$\forall x < a\, \exists X \le b\, \varphi(x, X) \to \exists Z\, \forall x < a\, \varphi(x, Z^{[x]})$$

for Σ^B_{i+1}-formulas φ is $\forall\exists\Sigma^B_{i+1}$-conservative over V^i (Zambella [182]). We will prove that the axiom of choice for Σ^B_1-formulas can be similarly $\forall\exists\Sigma^B_1$-conservatively added to VNC^1_*. We will in fact show that the same holds for a version of the axiom of choice without the bound on X.

Definition 4.18 Let Γ be a set of formulas. The *unbounded axiom of choice* is the schema

$$(\Gamma\text{-}AC) \qquad \forall x < a\, \exists X\, \varphi(x, X) \to \exists Z\, \forall x < a\, \varphi(x, Z^{[x]}),$$

where $\varphi \in \Gamma$ may have other parameters, and $Z^{[x]}$ denotes $\{u \mid \langle x, u\rangle \in Z\}$, where $\langle\cdot,\cdot\rangle$ is a pairing function. A theory T is closed under the *unbounded choice rule* Γ-AC^R, if

$$T \vdash \exists X\, \varphi(x, X) \Rightarrow T \vdash \exists Z\, \forall x < a\, \varphi(x, Z^{[x]}),$$

where $\varphi \in \Gamma$ may have other parameters.

It is easy to see that Σ^B_0-AC is equivalent to $\exists\Sigma^B_1$-AC, and similarly for AC^R.

Theorem 4.19 *Let T be a $\forall\exists\forall\Pi^B_1$-axiomatized extension of V^0 closed under Σ^B_0-AC^R. Then $T + \exists\Sigma^B_1$-AC is a $\forall\exists\Sigma^B_1$-conservative extension of T.*

Proof:

Claim 4.19.1 *Let $\mathcal{M} \vDash T$, $a \in M$, and φ a Σ^B_0-formula with parameters from M. Then there exists a model $\mathcal{N} \vDash T$ such that $\mathcal{M} \preceq_{\exists\Sigma^B_1} \mathcal{N}$, and $\mathcal{N}$ satisfies*

$$\exists Z\, \forall x < a\, \varphi(x, Z^{[x]})$$

or

$$\exists x < a\, \forall X\, \neg\varphi(x, X).$$

Proof: Let $\mathcal{M}_M$ be the expansion of $\mathcal{M}$ by constants for all elements of M. If

$$T + \mathrm{Th}_{\forall\Pi^B_1}(\mathcal{M}_M) + \exists x < a\, \forall X\, \neg\varphi(x, X)$$

is consistent, then any its model $\mathcal{N}$ satisfies the conclusion. Otherwise there is a sentence $\psi = \forall X\, \vartheta(X)$, where $\vartheta \in \Sigma^B_0$ has parameters from M, such that $\mathcal{M} \vDash \psi$, and

$$T \vdash \psi \to \forall x < a\, \exists X\, \varphi(x, X).$$

We can rewrite it as

$$T \vdash \exists X\, (\vartheta(X) \wedge x < a \to \varphi(x, X)),$$

hence

$$T \vdash \exists Z\, \forall x < a\, (\vartheta(Z^{[x]}) \to \varphi(x, Z^{[x]}))$$

by Σ^B_0-AC^R, which implies

$$\mathcal{M} \vDash \exists Z\, \forall x < a\, \varphi(x, Z^{[x]}).$$

Thus we may take $\mathcal{N} = \mathcal{M}$. □ (Claim 4.19.1)

Claim 4.19.2 *Any model of T has an $\exists\Sigma^B_1$-elementary extension to a model of $T + \Sigma^B_0$-AC.*

Proof: Let $\mathcal{M}_0 \vDash T$. We enumerate all pairs of an element $a \in M_0$ and a formula $\varphi \in \Sigma^B_0$ with parameters from M_0 as $\langle a_\alpha, \varphi_\alpha\rangle$ for $\alpha < \varkappa$, where $\varkappa$ is a cardinal. We construct an $\exists\Sigma^B_1$-elementary chain of models $\mathcal{N}_\alpha \vDash T$, $\alpha \leq \varkappa$, where $\mathcal{N}_0 = \mathcal{M}_0$, $\mathcal{N}_{\alpha+1}$ is obtained from $\mathcal{N}_\alpha$ by an application of Claim 1 using $a = a_\alpha$, $\varphi = \varphi_\alpha$, and $\mathcal{N}_\lambda = \bigcup_{\alpha<\lambda} \mathcal{N}_\alpha$ for limit λ. Notice that validity of T is preserved by unions of $\exists\Sigma^B_1$-elementary chains, as T is $\forall\exists\forall\Pi^B_1$-axiomatized. Then $\mathcal{M}_1 := \mathcal{N}_\varkappa$ is an $\exists\Sigma^B_1$-elementary extension of $\mathcal{M}_0$, $\mathcal{M}_1 \vDash T$, and

$$\mathcal{M}_1 \vDash \forall x < a\, \exists X\, \varphi(x, X) \to \exists Z\, \forall x < a\, \varphi(x, Z^{[x]})$$

for all $a \in M_0$, and $\varphi \in \Sigma^B_0$ with parameters from M_0. We continue in the same way to construct a chain $\mathcal{M}_0 \preceq_{\exists\Sigma^B_1} \mathcal{M}_1 \preceq_{\exists\Sigma^B_1} \mathcal{M}_2 \preceq_{\exists\Sigma^B_1} \dots$, whose union is a model of $T + \Sigma^B_0$-AC. □ (Claim 4.19.2)

Assume that $T + \exists\Sigma^B_1$-$AC = T + \Sigma^B_0$-AC proves a $\forall\exists\Sigma^B_1$-formula α, and let $\mathcal{M}$ be any model of T. Take an $\exists\Sigma^B_1$-elementary extension $\mathcal{N} \vDash T + \Sigma^B_0$-$AC$ of $\mathcal{M}$ by Claim 2. Then $\mathcal{N} \vDash \alpha$, hence $\mathcal{M} \vDash \alpha$. □

Corollary 4.20 *$VNC^1_* + \exists\Sigma^B_1$-$AC$ is a $\forall\exists\Sigma^B_1$-conservative extension of VNC^1_*.*

Proof: In view of Theorem 4.19 and Corollary 4.13, it suffices to show that VNC^1_* is closed under Σ^B_0-AC^R. Let

$$VNC^1_* \vdash \exists X\, \varphi(x, X, \vec{a}, \vec{A}),$$

where $\varphi \in \Sigma^B_0$ with all free variables shown. By Corollary 4.12 and Theorem 4.10, there exists an $L_{\overline{VNC}^1_*}$-term F such that

$$\overline{VNC}^1_* \vdash \varphi(x, F(x, \vec{a}, \vec{A}), \vec{a}, \vec{A}).$$

By Corollary 4.8, there exists an $L_{\overline{VNC}^1_*}$-term G such that $\overline{VNC}^1_*$ proves

$$G(a, \vec{a}, \vec{A}) = \{\langle x, y\rangle \mid x < a, y \in F(x, \vec{a}, \vec{A})\}.$$

Then

$$\overline{VNC}^1_* \vdash \forall x < a\, \varphi(x, G(a, \vec{a}, \vec{A})^{[x]}, \vec{a}, \vec{A}),$$

hence

$$VNC^1_* \vdash \exists Z\, \forall x < a\, \varphi(x, Z^{[x]}, \vec{a}, \vec{A})$$

by Corollary 4.6. □

5 Propositional translation

We will define a propositional formula

$$[\![\varphi(x_1, \dots, x_r, X_1, \dots, X_s)]\!]_{n_1,\dots,n_r,m_1,\dots,m_s}(p_{1,0}, \dots, p_{1,m_1-1}, \dots, p_{s,0}, \dots, p_{s,m_s-1})$$

for each $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula $\varphi(\vec{x}, \vec{X})$, and natural numbers $\vec{n}, \vec{m}$. Let $X_1, \dots, X_s$ be sets such that $|X_i| \le m_i$, and let $\tilde{X}_i$ denote the propositional valuation which assigns the value 1 to $p_{i,k}$ iff $k \in X_i$. Then the translation is defined in such a way that

$$(5) \qquad [\![\varphi]\!]_{\vec{n},\vec{m}}(\tilde{X}_1, \dots, \tilde{X}_s) = 1 \Leftrightarrow \mathbb{N} \vDash \varphi(\vec{n}, \vec{X}).$$

If $T(\vec{x}, \vec{X})$ is a set $L_{\overline{VNC}^1_*}$-term, we define a bounding term $b_T(\vec{n}, \vec{m})$, that is a number L_0-term such that $|T(\vec{n}, \vec{X})| \le b_T(\vec{n}, \vec{m})$ whenever $|X_i| \le m_i$ for each i, and we define propositional formulas $[\![T]\!]^k_{\vec{n},\vec{m}}$ for $k < b_T(\vec{n}, \vec{m})$ so that

$$(6) \qquad [\![T]\!]^k_{\vec{n},\vec{m}}(\tilde{X}_1, \dots, \tilde{X}_s) = 1 \Leftrightarrow \mathbb{N} \vDash k \in T(\vec{n}, \vec{X}).$$

Finally, if $t(\vec{x}, \vec{X})$ is a number $L_{\overline{VNC}^1_*}$-term, we define a bounding L_0-term b_t such that $t(\vec{n}, \vec{X}) \le b_t(\vec{n}, \vec{m})$ whenever $|X_i| \le m_i$ for all i, and we introduce propositional formulas $[\![t]\!]^k_{\vec{n},\vec{m}}$ for $k \le b_t(\vec{n}, \vec{m})$ so that

$$(7) \qquad [\![t]\!]^k_{\vec{n},\vec{m}}(\tilde{X}_1, \dots, \tilde{X}_s) = 1 \Leftrightarrow \mathbb{N} \vDash t(\vec{n}, \vec{X}) = k.$$

The bounding terms are defined inductively as follows:

$$\begin{aligned}
b_{x_i}(\vec{n}, \vec{m}) &= n_i, \\
b_{X_j}(\vec{n}, \vec{m}) &= m_j, \\
b_{f(t_1,\dots,t_r)}(\vec{n}, \vec{m}) &= f(b_{t_1}(\vec{n}, \vec{m}), \dots, b_{t_r}(\vec{n}, \vec{m})), \qquad f \in \{0, \mathrm{s}, +, \cdot, |x|\}, \\
b_{|T|}(\vec{n}, \vec{m}) &= b_T(\vec{n}, \vec{m}), \\
b_{C_\varphi(s,\vec{t},\vec{T})}(\vec{n}, \vec{m}) &= b_s(\vec{n}, \vec{m}), \\
b_{Y_\varphi(\vec{t},s,u,T)}(\vec{n}, \vec{m}) &= (|b_u(\vec{n}, \vec{m})| + 1) b_s(\vec{n}, \vec{m}).
\end{aligned}$$

The translations $[\![\varphi]\!]_{\vec{n},\vec{m}}$, $[\![T]\!]^k_{\vec{n},\vec{m}}$, $[\![t]\!]^k_{\vec{n},\vec{m}}$ are defined by simultaneous induction on complexity, along with formulas $\{\!\{R\}\!\}_{\vec{n},\vec{m}}$, $\{\!\{F\}\!\}^k_{\vec{n},\vec{m}}$, $\{\!\{f\}\!\}^k_{\vec{n},\vec{m}}$ for predicates R (including equality), set function symbols F, and number function symbols f. (The formulas $\{\!\{\alpha\}\!\}$ are in a sense variants of $[\![\alpha]\!]$, cf. Lemma 5.1 (v). However, they are conceptually different: they are defined for symbols of the language, not for formulas. In particular, they are not tied to particular variables X_j, and by the same token, they are not supposed to use the same propositional variables $p_{j,k}$ as above. They are only used in the definition of $[\![\alpha(\vec{t},\vec{T})]\!]$ below where formulas are explicitly substituted for their propositional variables, and we will indicate their variables explicitly when defining them. Their purpose is to make the definition of $[\![\alpha(\vec{t},\vec{T})]\!]$ below uniform, so that we do not have to treat specially the case where $\vec{t},\vec{T}$ are simple variables, and so that we do not have to repeat the unsightly expression with wide conjunctions and disjunctions for each symbol of the language separately.) Let us denote

$$I(\varphi)=\begin{cases}\top & \text{if } \varphi \text{ holds},\\ \bot & \text{otherwise.}\end{cases}$$

If α is a predicate or function symbol, we put

$$\begin{aligned}[\![\alpha(t_1,\dots,t_r,T_1,\dots,T_s)]\!]^k_{\vec{n},\vec{m}} = \bigvee_{\substack{k_1\le b_{t_1}(\vec{n},\vec{m})\\ \dots\\ k_r\le b_{t_r}(\vec{n},\vec{m})}} \Big(&\bigwedge_{i=1}^{r}[\![t_i]\!]^{k_i}_{\vec{n},\vec{m}}\\ &\wedge \{\!\{\alpha\}\!\}^k_{\vec{k},b_{T_1}(\vec{n},\vec{m}),\dots,b_{T_s}(\vec{n},\vec{m})}\big([\![T_1]\!]^0_{\vec{n},\vec{m}},\dots,[\![T_1]\!]^{b_{T_1}-1}_{\vec{n},\vec{m}},\dots,[\![T_s]\!]^0_{\vec{n},\vec{m}},\dots,[\![T_s]\!]^{b_{T_s}-1}_{\vec{n},\vec{m}}\big)\Big),\end{aligned}$$

where the superscript k is omitted if α is a predicate. We further define

$$\begin{aligned}
[\![x_i]\!]^k_{\vec{n},\vec{m}} &= I(k=n_i),\\
[\![X_j]\!]^k_{\vec{n},\vec{m}} &= p_{j,k},\\
[\![\varphi\circ\psi]\!]_{\vec{n},\vec{m}} &= [\![\varphi]\!]_{\vec{n},\vec{m}}\circ[\![\psi]\!]_{\vec{n},\vec{m}}, \qquad \circ\in\{\wedge,\vee,\neg\},\\
[\![\exists x\le t\,\varphi]\!]_{\vec{n},\vec{m}} &= \bigvee_{k\le b_t(\vec{n},\vec{m})}[\![x\le t\wedge\varphi]\!]_{k,\vec{n},\vec{m}},\\
[\![\forall x\le t\,\varphi]\!]_{\vec{n},\vec{m}} &= \bigwedge_{k\le b_t(\vec{n},\vec{m})}[\![x\le t\to\varphi]\!]_{k,\vec{n},\vec{m}},\\
\{\!\{R\}\!\}_{n,n'} &= I(n\,R\,n'), \qquad R\in\{\le,=\},\\
\{\!\{\in\}\!\}_{n,m}(p_0,\dots,p_{m-1}) &= \begin{cases}p_n & \text{if } n<m,\\ \bot & \text{otherwise},\end{cases}\\
\{\!\{=\}\!\}_{m,m'}(p_0,\dots,p_{m-1},q_0,\dots,q_{m'-1}) &= \bigwedge_{i<\min\{m,m'\}}(p_i\leftrightarrow q_i)\wedge\bigwedge_{i=m'}^{m-1}\neg p_i\wedge\bigwedge_{i=m}^{m'-1}\neg q_i,\\
\{\!\{f\}\!\}^k_{\vec{n}} &= I(f(\vec{n})=k), \qquad f\in\{0,\mathrm{s},+,\cdot,|x|\},\\
\{\!\{|X|\}\!\}^k_m(p_0,\dots,p_{m-1}) &= \begin{cases}p_{k-1}\wedge\bigwedge_{i=k}^{m-1}\neg p_i & \text{if } k>0,\\ \bigwedge_{i<m}\neg p_i & \text{otherwise},\end{cases}\\
\{\!\{C_{\varphi(u,\vec{x},\vec{X})}\}\!\}^k_{n,\vec{n},\vec{m}}(\vec{p}) &= I(k<n)\wedge[\![\varphi]\!]_{k,\vec{n},\vec{m}}(\vec{p}).
\end{aligned}$$

It remains to define the formula $\{\!\{Y_\varphi(\vec{p},n,r,I)\}\!\}^k_{\vec{p},n,r,m}(q_0,\dots,q_{m-1})$ for an open $L_{\overline{VNC}^1_*}$-formula $\varphi(\vec{p},d,x,y)$. We fix $\vec{p},n,m,r$, and we write $\{\!\{Y_\varphi\}\!\}^{d,x}$ for $\{\!\{Y_\varphi\}\!\}^{dn+x}_{\vec{p},n,r,m}$, where $x<n$. As φ has no free set variables, $[\![\varphi]\!]_{\vec{p},d,x,y}$ is a Boolean sentence with a definite truth value. We may thus define the edge relations

$$e(d,x,y) \Leftrightarrow [\![\varphi]\!]_{\vec{p},d,x,y}=1,$$
$$e^*(d,x,y) \Leftrightarrow e(d,x,y) \wedge \Big(\bigwedge_{z<y}\neg e(d,x,z) \vee \bigwedge_{z=y+1}^{n-1}\neg e(d,x,z)\Big)$$

for $d<|m|$, $x,y<n$. By induction on $d<|m|$, we define

$$\{\!\{Y_\varphi\}\!\}^{0,x}(q_0,\dots,q_{m-1}) = \begin{cases} q_x & \text{if } x<r,\\ \bot & \text{otherwise,}\end{cases}$$

$$\{\!\{Y_\varphi\}\!\}^{d+1,x}(q_0,\dots,q_{m-1}) = \begin{cases} \displaystyle\bigvee_{e^*(d,x,y)} \{\!\{Y_\varphi\}\!\}^{d,y}(q_0,\dots,q_{m-1}) & \text{if } d \text{ is even,}\\ \displaystyle\bigwedge_{e^*(d,x,y)} \{\!\{Y_\varphi\}\!\}^{d,y}(q_0,\dots,q_{m-1}) & \text{if } d \text{ is odd.}\end{cases}$$

We also put $\{\!\{Y_\varphi\}\!\}^{d,x}=\bot$ for $d>|m|$. Notice that the definition of e^* ensures that there are at most two y such that $e^*(d,x,y)$ for any given d,x, hence the conjunctions and disjunctions in the definition of $\{\!\{Y_\varphi\}\!\}^{d+1,x}$ are at most binary. As the formulas have depth $d\le|m|$, they are of size $O(m)$. It follows by induction on complexity that the formulas $[\![\alpha]\!]^{(k)}_{\vec{n},\vec{m}}$ for any fixed formula or term α have size $\mathrm{poly}(\vec{n},\vec{m})$ and logarithmic depth.

In fact, $[\![\alpha]\!]^{(k)}_{\vec{n},\vec{m}}$ is constructible in logarithmic space given $\vec{n},\vec{m},k$ in unary (note that α is fixed, it is not given to the machine as input). This can be established by induction on the complexity of α. The only non-obvious case is $\{\!\{Y_\varphi\}\!\}$, which can be constructed in log-space as follows. Given $\vec{p},d,x,y$, we can construct the Boolean sentence $[\![\varphi]\!]_{\vec{p},d,x,y}$ by the induction hypothesis, and we can evaluate it in log-space (note that we do not have to write it down, the log-space formula evaluator will call an algorithm computing bits of $[\![\varphi]\!]_{\vec{p},d,x,y}$ as a subroutine). This means that we can compute the relation e above, from which we compute e^* easily. We can also compute in log-space the extended connection language $L_{EC}(C)$ of the circuit C whose edge relation is given by e^*: given a starting node and $p\in\{0,1\}^*$, we can trace the path determined by p in a loop, where in each step we compute the (at most two) inputs of the given node by calling the algorithm for e^* to check all possibilities, and we follow the left or right input according to the relevant bit of p. Then we can compute the description of the formula $\{\!\{Y_\varphi\}\!\}$ (which is essentially C unfolded into a tree) by recursive depth-first traversal of C; the depth of recursion is logarithmic, and for each recursive call we only need to remember one bit (namely, whether we have descended into the left or right child), as we can recover the current node in C from the recursion stack using $L_{EC}(C)$.

It is also straightforward to show (5), (6), (7) by induction on complexity.

We recall that a *Frege system* is a propositional proof system given by a finite set F of rules of the form

$$\frac{\varphi_1,\dots,\varphi_n}{\varphi}$$

which is sound and implicationally complete. An F-proof of a formula φ is a sequence of propositional formulas ending with φ such that every formula is derived from previous formulas by an instance of an F-rule. By a well-known theorem of Cook and Reckhow [69], all Frege systems are polynomially equivalent, hence the choice of the basic rules does not matter (often one takes Modus Ponens and a list of axioms). Frege systems are also polynomially equivalent to the propositional version of Gentzen's sequent calculus LK, which is easier to work with in some contexts.

Lemma 5.1

(i) *If τ,σ are terms, then $b_{\tau(\vec{x},\vec{X},\sigma(\vec{x},\vec{X}))}(\vec{n},\vec{m}) = b_\tau(\vec{n},\vec{m},b_\sigma(\vec{n},\vec{m}))$.*

(ii) *If $\alpha(\vec{x},\vec{X},Y)$ is a formula or term, and $T(\vec{x},\vec{X})$ is a set term, then*

$$[\![\alpha(\vec{x},\vec{X},T(\vec{x},\vec{X}))]\!]^{(k)}_{\vec{n},\vec{m}} = [\![\alpha]\!]^{(k)}_{\vec{n},\vec{m},b_T(\vec{n},\vec{m})}\big([\![T]\!]^0_{\vec{n},\vec{m}},\dots,[\![T]\!]^{b_T(\vec{n},\vec{m})-1}_{\vec{n},\vec{m}}\big),$$

where k is present only if α is a term, and on the right-hand side the formulas are substituted for the variables corresponding to Y.

(iii) *If $t(\vec{x},\vec{X})$ is a number term, there are size* poly$(\vec{n},\vec{m})$ *log-space constructible Frege proofs of the formulas*

$$\bigvee_{k\le b_t(\vec{n},\vec{m})} [\![t]\!]^k_{\vec{n},\vec{m}},$$

$$\bigwedge_{\substack{k\le b_t(\vec{n},\vec{m})\\ l<k}} \big([\![t]\!]^k_{\vec{n},\vec{m}} \to \neg[\![t]\!]^l_{\vec{n},\vec{m}}\big).$$

(iv) *If $\alpha(y,\vec{x},\vec{X})$ is a formula or term, and $t(\vec{x},\vec{X})$ is a number term, then there are size* poly$(\vec{n},\vec{m})$ *log-space constructible Frege proofs of the formulas*

$$[\![\alpha(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{(k)}_{\vec{n},\vec{m}} \leftrightarrow \bigvee_{r\le b_t(\vec{n},\vec{m})} \big([\![t]\!]^r_{\vec{n},\vec{m}} \wedge [\![\alpha]\!]^{(k)}_{r,\vec{n},\vec{m}}\big),$$

where k is present only if α is a term, and we put $[\![\alpha]\!]^k_{r,\vec{n},\vec{m}} = \bot$ if α is a number term and $k > b_\alpha(\vec{n},\vec{m})$, or if α is a set term and $k \ge b_\alpha(\vec{n},\vec{m})$.

(v) *If $\alpha(\vec{x},\vec{X})$ is a predicate or function symbol, there are size* poly$(\vec{n},\vec{m})$ *log-space constructible Frege proofs of*

$$\{\!\{\alpha\}\!\}^{(k)}_{\vec{n},\vec{m}}(\vec{p}) \leftrightarrow [\![\alpha(\vec{x},\vec{X})]\!]^{(k)}_{\vec{n},\vec{m}}(\vec{p}).$$

Proof: By straightforward induction on complexity. For example, we will show the proof of the step for $\alpha = \beta(\vec{t},\vec{T})$ in (iv), where β is a predicate or function symbol. Let $r \le b_t(\vec{n},\vec{m})$. By the induction hypothesis, we can construct proofs of

$$[\![t_i(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{k_i}_{\vec{n},\vec{m}} \leftrightarrow \bigvee_{s\le b_t(\vec{n},\vec{m})} \big([\![t]\!]^s_{\vec{n},\vec{m}} \wedge [\![t_i]\!]^{k_i}_{s,\vec{n},\vec{m}}\big),$$

hence we construct proofs of

$$[\![t]\!]^r_{\vec{n},\vec{m}} \to \big([\![t_i(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{k_i}_{\vec{n},\vec{m}} \leftrightarrow [\![t_i]\!]^{k_i}_{r,\vec{n},\vec{m}}\big)$$

using (iii). Similarly, we can construct proofs of

$$[\![t]\!]^r_{\vec{n},\vec{m}} \to \big([\![T_i(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{j}_{\vec{n},\vec{m}} \leftrightarrow [\![T_i]\!]^{j}_{r,\vec{n},\vec{m}}\big).$$

Using the definition of $[\![\beta(\vec{t},\vec{T})]\!]$ and (i), we infer

$$[\![t]\!]^r_{\vec{n},\vec{m}} \to \Big[[\![\alpha(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{(k)}_{\vec{n},\vec{m}} \\ \leftrightarrow \bigvee_{\substack{k_1\le b_{t_1}(b_t(\vec{n},\vec{m}),\vec{n},\vec{m})\\ \cdots}} \Big(\bigwedge_i [\![t_i]\!]^{k_i}_{r,\vec{n},\vec{m}} \wedge \{\!\{\beta\}\!\}^{(k)}_{\vec{k},b_{T_1}(b_t(\vec{n},\vec{m}),\vec{n},\vec{m}),\dots}([\![T_1]\!]^0_{r,\vec{n},\vec{m}},\dots)\Big)\Big].$$

It is easy to see that there are short proofs of

$$\{\!\{\beta\}\!\}^{(k)}_{\vec{k},\vec{v}}(\vec{p}) \leftrightarrow \{\!\{\beta\}\!\}^{(k)}_{\vec{k},\vec{u}}(\vec{p},\vec{\bot})$$

for any $\vec{u} \ge \vec{v}$. Using the fact that $b_{T_j}(r,\vec{n},\vec{m}) \le b_{T_j}(b_t(\vec{n},\vec{m}),\vec{n},\vec{m})$, and the definition of $[\![t_i]\!]^j$ or $[\![T_i]\!]^j$ as $\bot$ for too large j, we obtain a proof of

$$[\![t]\!]^r_{\vec{n},\vec{m}} \to \Big[[\![\alpha(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{(k)}_{\vec{n},\vec{m}} \\ \leftrightarrow \bigvee_{\substack{k_1\le b_{t_1}(r,\vec{n},\vec{m})\\ \cdots}} \Big(\bigwedge_i [\![t_i]\!]^{k_i}_{r,\vec{n},\vec{m}} \wedge \{\!\{\beta\}\!\}^{(k)}_{\vec{k},b_{T_1}(r,\vec{n},\vec{m}),\dots}([\![T_1]\!]^0_{r,\vec{n},\vec{m}},\dots)\Big)\Big],$$

hence

$$[\![t]\!]^r_{\vec{n},\vec{m}} \to \big([\![\alpha(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{(k)}_{\vec{n},\vec{m}} \leftrightarrow [\![\alpha]\!]^{(k)}_{r,\vec{n},\vec{m}}\big)$$

by the definition of $[\![\beta(\vec{t},\vec{T})]\!]$. We get the required

$$[\![\alpha(t(\vec{x},\vec{X}),\vec{x},\vec{X})]\!]^{(k)}_{\vec{n},\vec{m}} \leftrightarrow \bigvee_{r\le b_t(\vec{n},\vec{m})} \big([\![t]\!]^r_{\vec{n},\vec{m}} \wedge [\![\alpha]\!]^{(k)}_{r,\vec{n},\vec{m}}\big)$$

using (iii). □

Theorem 5.2 *Let $\varphi(\vec{x},\vec{X})$ be a $\Sigma^B_0(L_{\overline{VNC}^1_*})$-formula provable in $\overline{VNC}^1_*$. Then the formulas $[\![\varphi]\!]_{\vec{n},\vec{m}}$ have Frege proofs of size* $\mathrm{poly}(\vec{n},\vec{m})$ *constructible in logarithmic space.*

Proof: It will be more convenient to work with sequent calculus, which is p-equivalent to Frege systems. The sequent $\vdash \varphi$ has an LK-proof π using substitution instances of axioms of $\overline{VNC}^1_*$ and equality axioms as extra initial sequents. We may reformulate the extensionality axiom $\forall x\,(x \in X \leftrightarrow x \in Y) \to X = Y$ of $BASIC$ as

$$\forall x < |X|\,(x \in X \to x \in Y) \wedge \forall x < |Y|\,(x \in Y \to x \in X) \to X = Y,$$

hence all the initial sequents are $\Sigma^B_0(L_{\overline{VNC}^1_*})$. Using the free-cut elimination theorem [42], we may thus assume that all formulas in π are $\Sigma^B_0(L_{\overline{VNC}^1_*})$. We will show by induction on the length of the proof that for every sequent $\Gamma \vdash \Delta$ in π, the sequents $[\![\Gamma]\!]_{\vec{n},\vec{m}} \vdash [\![\Delta]\!]_{\vec{n},\vec{m}}$ have propositional LK-proofs constructible in logarithmic space, where $[\![\Gamma]\!]_{\vec{n},\vec{m}}$ denotes $\{[\![\psi]\!]_{\vec{n},\vec{m}} \mid \psi \in \Gamma\}$ for any set of formulas Γ.

The induction steps for the cut rule, propositional rules, and structural rules is trivial, we simply use the induction hypothesis and apply the same rule.

If the last rule in the proof is the $\forall$-right rule, it must have the form

$$\frac{\Gamma \vdash y \le t \to \psi(y), \Delta}{\Gamma \vdash \forall x \le t\, \psi(x), \Delta}$$

as the conclusion is $\Sigma_0^B(L_{\overline{VNC}^1_*})$. By the induction hypothesis we can construct proofs of

$$[\![\Gamma]\!]_{\vec{n},\vec{m}} \vdash [\![y \le t \to \psi(y)]\!]_{r,\vec{n},\vec{m}}, [\![\Delta]\!]_{\vec{n},\vec{m}}$$

for every $r \le b_t(\vec{n},\vec{m})$, from which we derive

$$[\![\Gamma]\!]_{\vec{n},\vec{m}} \vdash \bigwedge_{r \le b_t(\vec{n},\vec{m})} [\![y \le t \to \psi(y)]\!]_{r,\vec{n},\vec{m}}, [\![\Delta]\!]_{\vec{n},\vec{m}}$$

using the $\wedge$-right rule. The case of $\exists$-left is similar.

If the last rule in the proof is the $\exists$-right rule, it must have the form

$$\frac{\Gamma \vdash s \le t \wedge \psi(s), \Delta}{\Gamma \vdash \exists x \le t\, \psi(x), \Delta}$$

where s is a term. By the induction hypothesis we can construct a proof of

$$[\![\Gamma]\!]_{\vec{n},\vec{m}} \vdash [\![s \le t \wedge \psi(s)]\!]_{\vec{n},\vec{m}}, [\![\Delta]\!]_{\vec{n},\vec{m}}.$$

By Lemma 5.1 (iv), there are short Frege proofs of

$$[\![s \le t \wedge \psi(s)]\!]_{\vec{n},\vec{m}} \leftrightarrow \bigvee_{r \le b_s(\vec{n},\vec{m})} ([\![s]\!]^r_{\vec{n},\vec{m}} \wedge [\![x \le t \wedge \psi(x)]\!]_{r,\vec{n},\vec{m}}).$$

Moreover, we can construct Frege proofs of $\neg[\![x \le t \wedge \psi(x)]\!]_{r,\vec{n},\vec{m}}$ for all $b_t(\vec{n},\vec{m}) < r \le b_s(\vec{n},\vec{m})$, hence we can construct a proof of the sequent

$$[\![s \le t \wedge \psi(s)]\!]_{\vec{n},\vec{m}} \vdash \bigvee_{r \le b_t(\vec{n},\vec{m})} [\![x \le t \wedge \psi(x)]\!]_{\vec{r},\vec{n},\vec{m}}.$$

We derive

$$[\![\Gamma]\!]_{\vec{n},\vec{m}} \vdash \bigvee_{r \le b_t(\vec{n},\vec{m})} [\![x \le t \wedge \psi(x)]\!]_{\vec{r},\vec{n},\vec{m}}, [\![\Delta]\!]_{\vec{n},\vec{m}}$$

by a cut. The case of the $\forall$-left rule is analogous.

It remains to construct proofs of propositional translations of substitution instances of axioms of $\overline{VNC}^1_*$ and equality axioms. If $\psi' = \psi(\vec{t},\vec{T})$ is an instance of an axiom ψ, then there are short Frege proofs of

$$(8) \qquad [\![\psi']\!]_{\vec{n},\vec{m}} \leftrightarrow \bigvee_{\substack{k_1 \le b_{t_1}(\vec{n},\vec{m}) \\ \dots}} \Big(\bigwedge_i [\![t_i]\!]^{k_i}_{\vec{n},\vec{m}} \wedge [\![\psi]\!]_{\vec{k},b_{T_1}(\vec{n},\vec{m}),\dots}([\![T_1]\!]^0_{\vec{n},\vec{m}},\dots)\Big)$$

by Lemma 5.1 (ii), (iv). If we can construct short proofs of $[\![\psi]\!]$, we can substitute the formulas $[\![T_i]\!]^j_{\vec{n},\vec{m}}$ in the proof (incurring a polynomial blow-up) and combine it with Lemma 5.1 (iii) to obtain the right-hand side of (8). It thus suffices to construct translations of the base form of the axioms.

Axioms of *BASIC* and equality axioms for L_0 are provable in V^0, hence their translations have log-space constructible proofs already in bounded-depth Frege [68].

The Σ^B_0-$\overline{COMP}$ axiom translates to

$$[\![u \in C_\psi(v, \vec{x}, \vec{X})]\!]_{k,l,\vec{n},\vec{m}} \leftrightarrow [\![u < v]\!]_{k,l} \wedge [\![\psi(u, \vec{x}, \vec{X})]\!]_{k,\vec{n},\vec{m}},$$

which can be proven equivalent to the tautology

$$I(k < l) \wedge [\![\psi]\!]_{k,\vec{n},\vec{m}} \leftrightarrow I(k < l) \wedge [\![\psi]\!]_{k,\vec{n},\vec{m}}$$

by Lemma 5.1 (v) and the definition of $\{\!\{C_\psi\}\!\}$.

Consider an instance

$$|Y_\psi(\vec{p}, n, r, I)| \le (|r| + 1)n \wedge \mathrm{eval}(n, |r|, \psi, I, Y_\psi(\vec{p}, n, r, I))$$

of *Open*-$\overline{SCV}$. We can prove

$$[\![|Y_\psi(\vec{p}, n, r, I)| \le (|r| + 1)n]\!]_{\vec{p},n,r,m}$$

easily using Lemma 5.1 (iii) and $b_{Y_\psi} = (|r| + 1)n$. Using the notation from the definition of $\{\!\{Y_\psi\}\!\}$, we can construct short proofs of

$$[\![dn + x \in Y_\psi(\vec{p}, n, r, I)]\!]_{d,x,\vec{p},n,r,m} \leftrightarrow \{\!\{Y_\psi\}\!\}^{d,x}$$

using Lemma 5.1 (v). As there are short proofs evaluating the Boolean sentences $[\![2 \mid d]\!]_d$ and $[\![\psi^*(\vec{p}, d, x, y)]\!]_{\vec{p},d,x,y}$ to $I(2 \mid d)$ and $I(e^*(d, x, y))$, we can construct short proofs of

$$\begin{aligned}\{\!\{Y_\psi\}\!\}^{d+1,x} \leftrightarrow \Big[\Big([\![2 \mid d]\!]_d \wedge \bigvee_{y<n} ([\![\psi^*]\!]_{\vec{p},d,x,y} \wedge \{\!\{Y_\psi\}\!\}^{d,y})\Big)& \\ \vee \Big([\![2 \nmid d]\!]_d \wedge \bigwedge_{y<n} ([\![\psi^*]\!]_{\vec{p},d,x,y} \to \{\!\{Y_\psi\}\!\}^{d,y})\Big)\Big]&\end{aligned}$$

for $d < |r|$ and $x < n$, using the definition of $\{\!\{Y_\psi\}\!\}^{d+1,x}$. Similarly, we construct proofs of

$$\{\!\{Y_\psi\}\!\}^{0,x} \leftrightarrow [\![x \in I]\!]_{x,m}.$$

Putting it all together, we obtain a proof of

$$[\![\mathrm{eval}(n, |r|, \psi, I, Y_\psi(\vec{p}, n, r, I))]\!]_{\vec{p},n,r,m}.$$

Translation of the equality axioms for C_ψ and Y_ψ is easy and left to the reader. (As a matter of fact, one can show that these axioms are redundant in $\overline{VNC}^1_*$.) □

Acknowledgements

I am grateful to Phuong Nguyen for enlightening discussions on VNC^1, and to David Mix Barrington for bringing FOLL to my attention.

The research was supported by grant IAA1019401 of GA AV ČR, grant 1M0545 of MŠMT ČR, and a grant from the John Templeton Foundation.

Chapter VI

A sorting network in bounded arithmetic

Abstract

We formalize the construction of Paterson's variant of the Ajtai–Komlós–Szemerédi sorting network of logarithmic depth in the bounded arithmetical theory VNC^1_* (an extension of VNC^1), under the assumption of existence of suitable expander graphs. We derive a conditional p-simulation of the propositional sequent calculus in the monotone sequent calculus MLK.

1 Introduction

Sorting is one of the most fundamental algorithmic operations, thus it is not surprising that much effort in theoretical computer science was invested in investigation of its computational complexity in various contexts. In particular, its exact parallel complexity was open for a long time. It has been known since the 1960s that it is fairly easy to construct parallel sorting algorithms using $O(\log^2 n)$ steps (Batcher [22]), but it proved quite difficult to further improve on this upper bound. It was only in 1983 when Ajtai, Komlós, and Szemerédi [4, 5] devised an ingenious algorithm achieving $O(\log n)$ parallel operations. The algorithm and its analysis were subsequently simplified by Paterson [148]. An important feature of the AKS algorithm is that the pattern of comparisons and swaps is fixed in advance independent of the data, hence the construction in fact gives a *sorting network* of depth $O(\log n)$. (This result is asymptotically optimal, as there is an obvious $\Omega(\log n)$ depth lower bound.) A sorting network is a structure consisting of comparators connected by wires, where a comparator is a device which takes two inputs and outputs them in sorted order.

In the present paper we are going to formalize the core of the AKS sorting network (or rather its version by Paterson) in the theory VNC^1_* of bounded arithmetic. More precisely, the basic building blocks of the AKS network, the so-called ε-halvers, are constructed using a certain kind of expander graphs. Construction of the expanders is a separate issue rather tangential to analysis of the main part of the network, we thus leave it out completely: we simply assume that VNC^1_* proves the existence of appropriate expanders, and all our results are conditional on this assumption. We note that some research towards formalization of expanders in bounded arithmetic is in progress [114].

There are several reasons why such a formalization is desirable. It is a basic problem in the development of bounded arithmetic to find what results in mathematics or computer science are provable in a given theory. In the other direction, the program of reverse mathematics seeks to find the minimal theory capable of proving a given statement. In particular, it is a natural foundational problem whether various properties of a given complexity class are provable using only concepts from the same class. Since the AKS network is a kind of a circuit of logarithmic depth, the natural class it fits into is (nonuniform) NC^1; it is thus reassuring to have a proof of its correctness in an NC^1-theory such as VNC^1_*.

The formalization has applications in propositional proof complexity. The monotone sequent calculus MLK is the fragment of the usual propositional sequent calculus LK using only sequents consisting of monotone formulas. Atserias, Galesi, and Pudlák [14] have shown that MLK quasipolynomially simulates LK (with respect to monotone sequents), but it is an open problem whether one can give a polynomial simulation. It was also shown in [14] that it is sufficient for an affirmative answer to construct monotone formulas for threshold functions such that their basic properties have polynomial-size proofs in LK. Such monotone formulas can be obtained by evaluation of the AKS network on 0–1 inputs. Since VNC^1_* proves soundness of the network, and translates into polynomial LK-proofs, the properties of these formulas required by [14] indeed have polynomial LK-proofs. We thus obtain a p-simulation of LK by MLK under our basic assumption on formalizability of expanders in VNC^1_*.

There are other potential applications of the AKS network in bounded arithmetic. As shown in [66], the closure of the class NL under complement is provable in the bounded arithmetic for NL. However, it is not known whether we can formalize the closure of the related class SL under complement in an SL-theory. Formalization of the AKS network is the first step, as the network is involved in the proof of SL = coSL from [134].

Our formalization is carried out in a not quite standard theory VNC^1_* introduced for this very purpose in Chapter V. This theory was chosen to satisfy two conflicting goals. On the one hand, the application to monotone sequent calculus described above requires that propositional translations of Π^B_1-formulas provable in the theory have polynomial-size proofs in LK, or equivalently, in Frege systems, hence we need some kind of an NC^1-theory. On the other hand, successful formalization of the AKS network requires at the very least that the theory proves that the network can be evaluated. We thus need the ability to evaluate (sufficiently uniformly described) circuits of logarithmic depth. The standard NC^1-theory VNC^1 is too weak for this purpose, as evaluation of log-depth circuits is not known to be possible in uniform NC^1 (i.e., ALOGTIME). VNC^1 can only evaluate log-depth circuits described by their extended connection language (ecl, see Ruzzo [162]), which is however not available for the AKS network (see also Section 6). The network is defined as a sequence of steps, each of which is described locally: the n elements are organized in a tree-like structure (varying in each step), and constant-depth subnetworks are applied to parts of this structure. A longer sequence of steps can move an element quite far in the structure in a hard to predict way (this is, after all, one of the reasons why the network can sort), and there does not seem to be any way of globally describing the ecl of the network other than to follow the path through the circuit step by step. (Note that this is unrelated to the complexity of description of expanders used in the construction: in fact, the expander-based gadgets have constant depth, hence their ecl is no harder than their direct connection language.)

The paper is organized as follows. Section 2 gives definitions of VNC^1_* and basic notions like comparator networks, as well as their elementary properties. In Section 3 we formally describe the AKS network, and in Section 4 we carry out the analysis of the network in VNC^1_*. In Section 5 we give a p-simulation of LK by MLK as an application, and Section 6 mentions some open problems.

2 Preliminaries

We refer the reader to §V.3 for definitions of the theories VNC^1_* and $\overline{\mathit{VNC}}^1_*$ which we will work in. As shown in §V.4, $\overline{\mathit{VNC}}^1_*$ is an open conservative extension of VNC^1_*, and $L_{\overline{\mathit{VNC}}^1_*}$-functions have Σ^B_1-definitions in VNC^1_*. For this reason, we will not distinguish the two theories, and we will work freely with $L_{\overline{\mathit{VNC}}^1_*}$-functions in VNC^1_*. Every $\Sigma^B_0(L_{\overline{\mathit{VNC}}^1_*})$-formula (or indeed, $\Delta^B_1(\mathit{VNC}^1_*)$-formula) is equivalent to an open $L_{\overline{\mathit{VNC}}^1_*}$-formula in $\overline{\mathit{VNC}}^1_*$. We will denote these formulas simply as NC^1_*-formulas, and likewise, we will refer to functions given by $L_{\overline{\mathit{VNC}}^1_*}$-terms as NC^1_*-functions. VNC^1_* proves NC^1_*-COMP and NC^1_*-IND. VNC^1_* contains VNC^1, and is contained in VL. The provably total computable functions of VNC^1_* are those definable by NC^1_*-functions in the standard model of arithmetic; this class fits between uniform NC^1 and L-uniform NC^1.

As $\mathit{VNC}^1_* \supseteq \mathit{VNC}^1 \supseteq \mathit{VTC}^0$, there is a well-behaved NC^1_*-function computing cardinality of sets. We will denote it $\operatorname{card} X$ in order to distinguish it from the basic symbol $|X|$ of L_0.

The main property of VNC^1_* we will use is that it can evaluate sufficiently uniform log-depth circuits. We can describe circuits using the following data:

- Numbers k, m, and s, where k is the number of inputs, m is the number of layers, and s is the size of each layer (we assume all layers have been padded with unused gates to have the same size).
- A function $T\colon m \times s \to \{\ulcorner\vee\urcorner, \ulcorner\wedge\urcorner, \ulcorner\neg\urcorner\} \cup \{\ulcorner x_i \urcorner \mid i < k\}$ indicating the type of each node, where we put e.g. $\ulcorner\vee\urcorner = 0$, $\ulcorner\wedge\urcorner = 1$, $\ulcorner\neg\urcorner = 2$, and $\ulcorner x_i\urcorner = i + 3$, and we represent T by its graph (a set $T \le ms(k + 3)$): i.e., $T(d, x, p)$ iff xth node on layer d has type p.
- A formula $\varphi(d, x, d', x')$ (possibly with other parameters) which states that node x' on layer d' is an input of gate x on layer d.

In order for a circuit to be well-formed, we demand that any gate uses only nodes on lower layers as inputs (but not necessarily from the adjacent layer), and all nodes have the correct number of inputs: 1 for negation nodes, 0 for input nodes, and at most 2 for conjunction and disjunction gates.

Lemma 2.1 (Thm. V.4.2) (*in VNC^1_**) *A circuit described as above with φ an NC^1_*-formula without set parameters, and m bounded by some $|a|$, can be evaluated on any input string. Moreover, the evaluation is computable by an NC^1_*-function.* □

Definition 2.2 A *comparator network* on n inputs is a directed acyclic graph without duplicate edges with three types of vertices: *input nodes* with fan-in 0 and fan-out 1, *comparators* with fan-in 2 and fan-out 2, and *output nodes* with fan-in 1 and fan-out 0. Input and output nodes are labelled by numbers $k < n$, and there exists exactly one input node and one output node labelled k for every k. The edges of the graph are called *wires*. For each comparator, one of its outgoing wires is labelled h (higher) and the other one is labelled l (lower). The *size* of a network is the number of its comparators. We represent a comparator network N by a sequence $N = \{w_i \mid i < s\}$, where w_i describes the ith node of N: its type, adjacent nodes, and labels. We require the sequence to start with the input nodes and end with the output nodes, both ordered according to their labels. If there is a wire going from node i to node j, we further require $i < j$. The network has *depth* at most d, if we can partition the comparators of N into at most d blocks (called *layers*), such that each layer is contiguous in the sequence ordering, and there are no wires going between two nodes of the same layer.

Let $\vec{X} = \{X_k \mid k < n\}$ be a sequence of sets, and $\leq$ a total ordering whose domain includes every X_k. An *evaluation* of a network N with respect to $\leq$ on input $\vec{X}$ is a sequence of sets E_e indexed by wires e of N such that $E_e = X_k$ if e is the outgoing wire of an input node with label k, and if l and h are the lower and higher outgoing wires of a comparator with incoming wires e, f, then $E_l = \min_\leq\{E_e, E_f\}$, and $E_h = \max_\leq\{E_e, E_f\}$. The *result* of an evaluation E is the sequence of sets $\vec{Y} = \{Y_k \mid k < n\}$ such that $Y_k = E_e$, where e is the incoming wire of the output node with label k. We write $\vec{Y} = \mathrm{eval}(N, \leq, \vec{X})$ (the context should suffice to disambiguate between this notation and the eval-formula from the definition of VNC^1_*).

Since comparators have the same number of incoming and outgoing wires, there are exactly n wires at any section of a network with n inputs. That is, if $N = \{w_i \mid i < s\}$ is a network with n inputs, and $i < s$ is a comparator node, then we can show by straightforward induction on i that there are n wires going from nodes $j \leq i$ to nodes $j > i$. Consequently, each layer has size at most $n/2$, and a network of depth d has size at most $nd/2$.

A comparator network of logarithmic depth resembles an NC^1-circuit. Indeed, if we want to evaluate a uniformly described network on a 0–1 input, we can replace each comparator by a pair of $\wedge$ and $\vee$ gates (i.e., min and max in the Boolean domain), turning it into a logarithmic depth bounded fan-in circuit, which can be evaluated in VNC^1_*. This argument does not work for nonconstant domains, as we then cannot compute the required comparisons by bounded depth bounded fan-in circuits. Nevertheless, we will show that we can evaluate a log-depth network on arbitrary inputs in VNC^1_* using a simple trick based on a variant of the 0–1 principle.

Lemma 2.3 (*in VNC^1_**) *Let N be a comparator network on n inputs of depth $d \leq \log m$ for some m defined by an NC^1_*-formula without set parameters, $\leq$ a total ordering defined by an NC^1_*-formula, and $\{X_k \mid k < n\}$ a sequence of sets in the domain of $\leq$. Then there exists a unique evaluation of N on input $\vec{X}$ with respect to $\leq$.*

Proof: Uniqueness: if E and E' are two evaluations of $N = \{w_i \mid i < s\}$, we prove by straightforward induction on $i < s$ that $E_e = E'_e$ for all wires e incident with a node $j \leq i$.

Existence: the basic idea is to represent the input value X_i by the set $B_i \subseteq n$ such that $j \in B_i$ iff $X_i \geq X_j$. Then $X_i \leq X_j$ iff $B_i \subseteq B_j$, hence $\max_\leq\{X_i, X_j\}$, $\min_\leq\{X_i, X_j\}$ are represented by $B_i \cup B_j$, $B_i \cap B_j$, respectively. In other words, we can compute min or max by n parallel binary conjunctions or disjunctions in this representation, as in the 0–1 case. In more detail, we construct a circuit C as follows. For each wire e in N, we put in C nodes e_i for all $i < n$. If w is a comparator in N with incoming wires e, f and outgoing wires l, h, we include in C the gates $l_i = e_i \wedge f_i$, $h_i = e_i \vee f_i$. If e^k is the outgoing wire of the kth input node, we make e^k_i an input node of the circuit and initialize it to 1 iff $X_i \leq X_k$ using NC^1_*-comprehension. Since C is a circuit of logarithmic depth defined by an NC^1_*-formula without set parameters, we can evaluate it by Lemma 2.1. Let V be its valuation, let $\varphi(e)$ denote the NC^1_*-formula $\exists k < n\, \forall i < n\, V(e_i) = V(e^k_i)$, and for each wire e, define the set

$$E_e = \{j \mid \exists k < n\, (\forall i < n\, V(e_i) = V(e^k_i) \wedge j \in X_k)\}$$

by NC^1_*-comprehension. If w is a comparator with incoming wires e, f and outgoing wires l, h, and if we assume $\varphi(e) \wedge \varphi(f)$, then it is easy to see that $\varphi(l) \wedge \varphi(h)$, and $E_l = \min_\leq\{E_e, E_f\}$, $E_h = \max_\leq\{E_e, E_f\}$. We can thus prove by induction $\varphi(e)$ for all e, which implies that E is a correct evaluation of N. □

Lemma 2.4 (*in VNC^1_**)
Let N, $\vec{X}$, and $\leq$ be as in Lemma 2.3. Let $\preceq$ be an NC^1_*-defined total order, and F an NC^1_*-function such that $X \leq X'$ implies $F(X) \preceq F(X')$. Then $\mathrm{eval}(N, \preceq, F(\vec{X})) = F(\mathrm{eval}(N, \leq, \vec{X}))$.

Proof: Let E be the evaluation of N on $\vec{X}$ wrt $\leq$, and put $E'_e = F(E_e)$. Then E' is an evaluation of N on $F(\vec{X})$ wrt $\preceq$. □

Lemma 2.5 (*in VNC^1_**) *Let N, $\vec{X}$, and $\leq$ be as in Lemma 2.3, and $\vec{Y} = \mathrm{eval}(N, \leq, \vec{X})$. Then there exists a permutation π of n such that $Y_i = X_{\pi(i)}$ for all $i < n$.*

Proof: The proof of Lemma 2.3 shows that

$$(1) \qquad \forall i < n\, \exists j < n\, Y_i = X_j.$$

On the other hand, if $j < n$, $N = \langle w_k \mid k < s\rangle$, and $\vec{E}$ is the evaluation of N on $\vec{X}$ wrt $\leq$, we can show by induction on k the following property: if w_k is a comparator node, there exists a wire e going from a node $\leq k$ to a node $> k$ such that $E_e = X_j$. Therefore,

$$(2) \qquad \forall j < n\, \exists i < n\, X_j = Y_i.$$

Assume first that the X_i's are pairwise distinct. Then for each i there is a *unique* j such that $Y_i = X_j$ by (1). We put $\pi(i) = j$. Then (2) implies that π is surjective, hence it is a bijection by *PHP* (provable in $VTC^0 \subseteq VNC^1_*$), and $Y_i = X_{\pi(i)}$ by the definition.

In the general case, we define

$$i \preceq j \iff X_i \leq X_j \wedge (X_j = X_i \to i \leq j).$$

It is easy to see that $\preceq$ is a total order on n, hence by the previous part of the proof, there exists a permutation π such that $\mathrm{eval}(N, \preceq, \langle 0, \dots, n-1 \rangle)_i = \pi(i)$. Using Lemma 2.4 for $F(i) = X_i$, we obtain

$$Y_i = \mathrm{eval}(N, \leq, \langle F(0), \dots, F(n-1) \rangle)_i = F(\mathrm{eval}(N, \preceq, \langle 0, \dots, n-1 \rangle)_i) = X_{\pi(i)}. \qquad \square$$

Lemma 2.6 (*in VNC^1_**) *If $\leq$ is a total ordering defined by an NC^1_*-formula, and $\langle X_i \mid i < n \rangle$ a sequence of sets in the domain of $\leq$, then there exists a permutation π of n such that $X_{\pi(i)} \leq X_{\pi(j)}$ for each $i \leq j < n$.*

Proof: Define

$$i \preceq j \iff X_i < X_j \vee (X_i = X_j \wedge i \leq j).$$

It is easy to see that $\preceq$ is a total order on n. Put

$$\sigma(i) := \mathrm{card}\{k < n \mid k \prec i\}.$$

Clearly $i \prec j$ implies $\sigma(i) < \sigma(j)$. In particular, σ is injective, hence it is a permutation by *PHP*. We can thus define $\pi = \sigma^{-1}$, and then $i \leq j$ implies $\pi(i) \preceq \pi(j)$, which gives $X_{\pi(i)} \leq X_{\pi(j)}$. $\square$

3 Ajtai–Komlós–Szemerédi–Paterson network

In this section we will define in detail Paterson's variant of the Ajtai–Komlós–Szemerédi network. We generally follow Paterson's construction, but we had to disentangle the gradual way in which he describes it: first, we learn the basic tree-like structure with idealized rational sizes; then it is modified so that the bottom and top parts work out correctly; then it turns out that one tree is not enough, and it is going to be split in many trees after some point; and finally, changes throughout the whole construction are proposed to make all sizes integer rather than rational. In contrast, we have to formalize (and therefore explicitly describe) the final network. We made some inessential changes to facilitate the formalization.[1]

Before describing the sorting network proper, let us start with a few auxiliary structures.

Definition 3.1 Let D and $0 < \varepsilon < 1$ be constants. An *$\langle \varepsilon, D \rangle$-expander on $m + m$ vertices* is a bipartite graph $G \subseteq m \times m$ such that every vertex (in either partition) has degree at most D, and for every $k \leq m$, every subset of one partition with more than εk vertices has at least $(1 - \varepsilon)k$ neighbours in the other partition.

[1]For example, the last step (making sizes integer) of Paterson's construction is actually incompatible with his choice of the parameters of the network. He solves it by modification of what we denote Case 3 below so that the splitting is applied not only to the root bag and cold storage, but to more levels on top of the tree. Since this introduces an undesirable extra complication to the overall structure, we chose to solve it in another way, namely by picking a different set of parameters. In general, we made no effort to optimize parameters and the resulting constant in the size bound for the network, since we need the network for strictly theoretical purposes where the values of these constants are irrelevant.

From now on, we fix the first parameter ε_0 of our sorting network, say $\varepsilon_0 = 1/600$.

Assumption 3.2 *There exists a constant D and a (parameter-free) NC^1_*-function $G(m)$ such that VNC^1_* proves: for all numbers m, $G(m)$ is an $\langle \varepsilon_0, D\rangle$-expander on $m + m$ vertices.*

We fix D from Assumption 3.2 as our next parameter.

Definition 3.3 An *ε_0-halver on m elements,* where m is even, is a comparator network with m inputs whose output is partitioned into two blocks (left and right) of size $m/2$ with the following property: for each $k \leq m/2$, if a 0–1 input contains k zeros, then at most $\varepsilon_0 k$ zeros get in the right output block, and if the input contains k ones, then at most $\varepsilon_0 k$ ones get in the left output block.

Lemma 3.4 *There is an NC^1_*-function which, provably in VNC^1_*, computes for any even m an ε_0-halver on m inputs of depth D^2.*

Proof: Let G be an $\langle \varepsilon_0, D\rangle$-expander on $m/2 + m/2$ vertices given by Assumption 3.2. For each partition and each its vertex, we enumerate its outgoing edges by numbers $i < D$. In this way, every edge is labelled by a pair of numbers $\langle i, j\rangle \in D \times D$. As different edges with the same label are disjoint, the labelling defines a partition of the edges of G into D^2 partial matchings, which we denote by $\{G_k \mid k < D^2\}$. We construct a comparator network on m inputs as follows. We split the wires between any two adjacent layers into a left and right block as in Definition 3.3, and we identify each block with the vertices of one partition of G. For each $k < D^2$, we include a layer of comparators corresponding to the edges in G_k (with the higher output of the comparator landing in the right block).

Consider an evaluation of the network on a 0–1 input, and a wire a from the left block. In each layer of the network, the value of a is either unchanged, or it is replaced with the minimum of the value of a and of a value of some wire in the right block, hence the value of a never increases during the computation. Symmetrically, the value of a wire in the right block never decreases. Let $\langle a, b\rangle \in G$. We have $\langle a, b\rangle \in G_k$ for some k. After the kth layer, the value of wire a is less that or equal to the value of wire b, and then the former can only decrease, and the latter only increase, hence the relation is preserved. It follows that the output of the network is compatible with G in the following sense: the output value of a wire a in the left block is less that or equal to the value of a wire b in the right block whenever they are joined by an edge in G.

Let there be $k \leq m/2$ zeros and $m - k$ ones in the input (or in the output, for that matter), and assume for contradiction that the right output block contains strictly more than $\varepsilon_0 k$ zeros. As G is an expander, the positions of these zeros are connected by an edge to at least $(1 - \varepsilon_0)k$ positions in the left block. By the compatibility property, the value of each of them is also zero, hence the total number of zeros in the output is more than $\varepsilon_0 k + (1 - \varepsilon_0)k = k$, a contradiction. The case of k ones in the input is symmetric. □

Definition 3.5 Let N be a comparator network with $m+k$ inputs. A network N' on m inputs is constructed from N by *chopping from left* as follows. We pick k input wires (say, the wires with the smallest index), and mark them for deletion. If both inputs of a comparator are marked, we mark both outputs as well. If only one input of a comparator is marked, we mark its lower output. When we finish the marking, we delete all marked wires and comparators with both inputs marked, and we replace each comparator with one marked input with a wire connecting its unmarked input and output. (This is equivalent to the following operation: we expand the m inputs with k virtual elements, we apply N while considering the virtual elements to order below the real elements, and then we delete the virtual elements from output.) *Chopping from right* is defined symmetrically.

Definition 3.6 Let $\varepsilon \in [\varepsilon_0, 1)$ be a rational constant, and $m \geq l > 0$ be even integers. An *$\langle l, \varepsilon, \varepsilon_0\rangle$-separator on m elements* is a comparator network N whose m outputs are partitioned into four blocks, FL, CL, CR, FR (here L, R, C, and F stand for left, right, centre, and far, respectively), of sizes $\operatorname{card} FL = \operatorname{card} FR = l/2$, $\operatorname{card} CL = \operatorname{card} CR = (m-l)/2$, such that N is an ε_0-halver with respect to the blocks $L = FL \cup CL$ and $R = FR \cup CR$, and satisfies the following additional property: for any $k \leq l/2$, if a 0–1 input contains k zeros, then at most εk zeros are output outside FL, and if the input contains k ones, then at most εk ones are output outside FR.

Lemma 3.7 *Let $p \geq 0$ be an integer constant. There exists an NC^1_*-function which, provably in VNC^1_*, computes an $\langle l, (p+1)\varepsilon_0, \varepsilon_0\rangle$-separator on m elements of depth $(p+1)D^2$ for any given even m and even $l \leq m$ such that $l \geq m2^{-p}$.*

Proof: We proceed by induction on p. (Notice that the induction is external, as p is standard.) If $p = 0$, it suffices to take an ε_0-halver on m elements from Lemma 3.4. Let $p > 0$, and assume that the statement is true for $p - 1$. We are given even m, l such that $2^{-p}m \leq l \leq m$. If $2^{1-p}m \leq l$, we may simply use the induction hypothesis, hence we assume $l \leq 2^{1-p}m$. We distinguish two cases.

First, assume that $p > 1$, so that $2l \leq m$. By the induction hypothesis, we obtain a $\langle 2l, p\varepsilon_0, \varepsilon_0\rangle$-separator on m inputs. We denote its output blocks by FL', CL', CR', FR'. We take an ε_0-halver H on l elements. We apply H to FL', denoting its output blocks as FL (the left one) and CL'' (the right one), and symmetrically we apply a copy of H in parallel to FR' obtaining CR'' and FR. We put $CL = CL' \cup CL''$ and $CR = CR' \cup CR''$. Clearly the resulting network is an ε_0-halver. Consider an input with k zeros, where $k \leq l/2$. Then $k \leq 2l/2$, hence at most $p\varepsilon_0 k$ zeros land outside FL' by the induction hypothesis. There remain at most $k \leq l/2$ zeros in FL', and H is a halver, thus at most $\varepsilon_0 k$ zeros end up in CL''. In total, at most $(p+1)\varepsilon_0 k$ zeros end up outside FL. The case of an input with k ones is symmetric.

Finally, let $p = 1$, thus $m/2 \leq l \leq m$. We construct our network as follows. First, we apply an ε_0-halver on m elements, obtaining the blocks L and R. We fix an ε_0-halver H on l elements. We chop H from right to $m/2$ inputs, and apply it to L, denoting its left output block with $l/2$ elements as FL, and its chopped right block as CL. Symmetrically, we chop a copy of H from left to $m/2$ inputs, and apply it in parallel to R, obtaining FR and CR. Again, it is clear that the network is an ε_0-halver. Consider an input with k zeros, where $k \leq l/2$. At most $k\varepsilon_0$ zeros end up in R. We can simulate the effect of chopped H on L as follows: we extend the partial result in L with ones to l elements, apply H, and discard the excessive ones from the right block CL. The number of zeros in the extended input is thus still at most $k \le l/2$, hence at most $\varepsilon_0 k$ zeros land in CL. In total, at most $2\varepsilon_0 k$ zeros end up outside FL, as required. The case of an input with k ones is symmetric. □

We now proceed to the description of the sorting network. Let n be the number of inputs. We fix the parameters $p = 4$, $\lambda_0 = 2^{-p} = 1/16$, $\varepsilon = (p+1)\varepsilon_0 = 1/120$, $\lambda = 1/8$, $A = 3$, $C = 150$, $\nu = 2\lambda A + (1-\lambda)/2A = 43/48$, $c_m = \lceil -\log 2A/\log\nu\rceil = 17$. Without loss of generality, we assume

$$n \ge C/\nu.$$

The sorting network consists of $O(\log n)$ *stages*, where the transition from one stage to the next one is computed by a constant depth comparator network. In each stage, the n elements are divided into a number of *bags*. Each bag is capable of accommodating a certain number of elements, called its *capacity*, but some of the bags may actually hold fewer elements than its capacity. The bags are organized in a subset of an ambient binary tree. All bags on the same level of the tree have the same capacity. In stage t, bags with nonempty capacity only appear at levels d such that $d \equiv t \pmod 2$ and $d_0(t) \le d \le d_1(t)$. (Note that we number stages and levels of the tree starting from 0.) We will write just d_0, d_1 if t is understood from the context. We label bags on level d by numbers $i < 2^d$ in the natural order from left to right (i.e., the children of the ith bag on level d are the $2i$th and $(2i+1)$th bags on level $d+1$).

The level d_1 is called the *bottom level*, and it is the only one which may contain bags not filled up to their full capacity. The level d_0 is the *root level*. The condition $d \ge d_0$ effectively means that the structure consists of 2^{d_0} disjoint trees with roots at the root level. Each of these 2^{d_0} trees also has a *cold storage* attached to it, which is a special bag sitting outside the ambient tree structure. Note that the roots of the trees may be empty, if $d_0(t) \not\equiv t \pmod 2$. We label the trees by numbers $i < 2^{d_0}$ in the left-to-right order, the same as their roots.

The parameters and sizes of various parts of the structure are as follows. For any $t \le c_m\lceil\log n\rceil$ and $d \le 2\lceil\log n\rceil$, put

$$s'(t,d) = \frac{n}{2^d}\left(1 - (2A)^{d-2}\nu^t\right).$$

(Notice that here and below, the exponentiation has a fixed base, and the exponent is bounded by $O(\log n)$, hence the expression is definable by a well-behaved bounded formula in $I\Delta_0 \subseteq V^0$.) We define

$$\begin{aligned}
d_1'(t) &= \max\{d \mid (2A)^{d-2} < \nu^{-t}\} = \max\{d \mid s'(t,d) > 0\},\\
d_1(t) &= d_1'(t) - ((d_1'(t) - t) \bmod 2),\\
d_0'(t) &= \min\{d \mid nA^d\nu^t \ge C\},\\
d_0(t) &= \begin{cases} d_0'(t) - 1 & \text{if } t > 0, d_0'(t) > d_0'(t-1), d_0'(t-1) \equiv t \pmod 2,\\ d_0'(t) & \text{otherwise.}\end{cases}
\end{aligned}$$

As $A > 1 > \nu$, $d_\alpha(t)$ are well-defined by a bounded formula, and $d_\alpha(t) = O(t)$.

There are $n \bmod 2^{d_0}$ trees of size (i.e., the number of elements it holds) $\lceil n2^{-d_0}\rceil$, and $2^{d_0} - (n \bmod 2^{d_0})$ trees of size $\lfloor n2^{-d_0}\rfloor$. These sizes are distributed so that the leftmost i trees have total size $\lfloor in2^{-d_0}\rfloor$, thus the tree with label i has size

$$T(t,i) = \lfloor (i+1)n2^{-d_0}\rfloor - \lfloor in2^{-d_0}\rfloor = \begin{cases} \lceil n2^{-d_0}\rceil & \text{if } (in \bmod 2^{d_0}) > ((i+1)n \bmod 2^{d_0}), \\ \lfloor n2^{-d_0}\rfloor & \text{otherwise.} \end{cases}$$

If $d \geq d_0$ and $d \equiv t \pmod 2$, each subtree rooted at level d has nominal capacity

$$s(t,d) = 2\lceil s'(t,d)/2\rceil,$$

and actually holds $\max\{0, s(t,d)\}$ elements. This means that the capacity of any bag at level d is

$$b(t,d) = \begin{cases} s(t,d) - 4s(t,d+2) & \text{if } d_0 \leq d \leq d_1, d \equiv t \pmod 2, \\ 0 & \text{otherwise,} \end{cases}$$

and the number of elements it holds is

$$h(t,d) = \begin{cases} s(t,d) & \text{if } d = d_1, \\ b(t,d) & \text{otherwise.} \end{cases}$$

Note that the capacity and actual content of each bag is even. The capacity (and content) of cold storage is accordingly

$$c(t,i) = \begin{cases} T(t,i) - s(t,d_0) & \text{if } d_0 \equiv t \pmod 2, \\ T(t,i) - 2s(t,d_0+1) & \text{otherwise,} \end{cases}$$

where $i < 2^{d_0}$ is the label of the tree.

We also define "ideal sizes" of the various parameters, which are rational numbers approximated by the real sizes. The ideal size of each tree is $T'(t) = n2^{-d_0}$. We already know the ideal subtree capacity $s'(t,d)$. The ideal bag capacity is defined by

$$b'(t,d) = \begin{cases} s'(t,d) - 4s'(t,d+2) = \left(1 - \dfrac{1}{4A^2}\right) nA^d\nu^t & \text{if } d_0 \leq d \leq d_1, d \equiv t \pmod 2, \\ 0 & \text{otherwise,} \end{cases}$$

and the ideal cold storage capacity is

$$c'(t) = \begin{cases} T'(t) - s'(t,d_0) = \dfrac{1}{4A^2} nA^{d_0}\nu^t & \text{if } d_0 \equiv t \pmod 2, \\ T'(t) - 2s'(t,d_0+1) = \dfrac{1}{2A} nA^{d_0}\nu^t & \text{otherwise.} \end{cases}$$

Notice that $d_0(0) = d_1(0) = 0$, thus the structure at stage 0 consists of a single root bag and the associated cold storage. We initialize the network by putting arbitrary $s(0,0)$ elements to the root bag, and the rest to the cold storage.

Let t_m be the least $t > 0$ such that $d_0(t) = d_1(t)$. We will see below (Lemma 4.4) that t_m exists, $t_m \leq c_m\lceil\log n\rceil$, and $T(t_m, i)$ is bounded by a constant. The stage t_m will be the last regular stage of our network. After this stage, we sort each of the 2^{d_0} constant-size trees using a suitable constant-size sorting network, and stop.

We have to define the constant-depth network which makes the transition from stage $t < t_m$ to stage $t + 1$. A general overview is that we will apply a suitable constant-depth subnetwork to each nonempty bag to split its content into a few parts, which we send to its parent and children bags. Root bags will exchange elements with their cold storage instead of a parent. Notice that when a bag is nonempty at stage t, then its children and parent are empty (except for the cold storage), whereas the opposite holds at stage $t + 1$. Now we describe the actual network fragments. We have to distinguish several cases.

Case 1: we consider a nonempty bag B on level d such that $d_0 < d \leq d_1$. If $d = d_1(t) > d_1(t + 1)$, we send all of B to its parent. Otherwise, we use an $\langle l, \varepsilon, \varepsilon_0\rangle$-separator of depth $(p+1)D^2$ from Lemma 3.7 to split B into FL, CL, CR, and FR, where $l = s(t, d)-2s(t+1, d+1)$. We send CL to the left child, CR to the right child, and $FL \cup FR$ to the parent.

Case 2: a nonempty root bag B, assuming $d_0(t) = d_0(t + 1)$. We apply a separator just like in Case 1, except that we send $FL \cup FR$ to the cold storage instead of B's parent.

Case 3: a root bag B of the ith tree, assuming $d_0(t) \neq d_0(t+1)$. We will see in Lemma 4.2 that $d_0(t+1) = d_0(t)+1$, and $b(t, d_0)+c(t, i)$ is bounded by a constant. Note that $d_1(t) \geq d_0(t)+2$. We merge the bag with its cold storage, and apply a constant-size sorting network to split it to two pieces, L of size $T(t+1, 2i)-2s(t, d_0(t)+2)$, and R of size $T(t+1, 2i+1)-2s(t, d_0(t)+2)$, so that each element of L is less than or equal to each element of R. We put arbitrary $c(t+1, 2i)$ elements from L to the newly created cold storage of the left child of B, and send the rest of L to the left child itself. We do the same with R and the right child.

Case 4: a cold storage. If $d_0 \equiv t \pmod 2$, we expand the storage with some elements sent from its root bag, as described in Case 2, or merge it with the root bag and split it to children, as described in Case 3. If $d_0 \not\equiv t \pmod 2$ (which implies $d_0(t) = d_0(t + 1)$, as we will see), we send arbitrary $s(t + 1, d_0) - 2s(t, d_0 + 1)$ elements to the root bag.

We observe that the network is defined by an NC^1_*-function $F(n)$.

4 Analysis of the network

We first check that our definition of the various parameters of the network are sensible, and that all sizes work out correctly when shuffling elements around.

We have already seen why d_0 and d_1 are well-defined.

Lemma 4.1 *(in VNC^1_*)*

(i) $d'_1(t) \leq d'_1(t + 1) \leq d'_1(t) + 1$.

(ii) $d_1(t + 1) - d_1(t) = \pm 1$.

(iii) *If* $t > 0$, *then* $d_1(t) > 0$.

Proof:

(i): $d'_1(t) \leq d'_1(t+1)$ is clear as $\nu < 1$. We have $(2A)^{d_1(t)-1} \geq \nu^{-t}$, hence $(2A)^{d_1(t)} \geq \nu^{-(t+1)}$ as $A \geq \nu^{-1}$, which implies $d_1(t+1) < d_1(t) + 2$.

(ii): Since $d'_1(t) - 1 \leq d_1(t) \leq d'_1(t)$, we obtain $|d_1(t+1) - d_1(t)| \leq 2$ from (i). However, $d_1(t) \equiv t \not\equiv t+1 \equiv d_1(t+1) \pmod 2$, hence $|d_1(t+1) - d_1(t)| = 1$.

(iii): Since $(2A)^0 = 1 < \nu^{-t}$, we have $d'_1(t) \geq 2$ and $d_1(t) \geq 1$. □

Lemma 4.2 (*in* VNC^1_*)

(i) $d'_0(t) \leq d'_0(t+1) \leq d'_0(t) + 1$.

(ii) $d'_0(t-1) \leq d_0(t) \leq d'_0(t)$.

(iii) $d_0(t) \leq d_0(t+1) \leq d_0(t) + 1$.

(iv) *If* $d_0(t) < d_0(t+1)$, *then* $d_0(t) \equiv t \pmod 2$, *and* $b(t, d_0(t)) + c(t, i) \leq \lceil C/\nu \rceil$.

Proof:

(i) follows from $\nu < 1 < A\nu$ as in the proof of Lemma 4.1.

(ii): If $d_0(t) \neq d'_0(t)$, then $d'_0(t) > d'_0(t-1)$ and $d_0(t) = d'_0(t) - 1$ by the definition.

(iii): We have $d_0(t) \leq d'_0(t) \leq d_0(t+1)$ from (ii). $d_0(t+1) \geq d_0(t)$ could only happen if $d_0(t) < d'_0(t) < d'_0(t+1) = d_0(t+1)$. The former inequality implies $d_0(t) = d'_0(t-1) \equiv t \pmod 2$, hence $d'_0(t) = d_0(t)+1 \equiv t+1 \pmod 2$, thus by the definition $d_0(t+1) = d'_0(t+1)-1$, a contradiction.

(iv): If $d_0(t) < d'_0(t)$, then $d_0(t) = d'(t-1) \equiv t \pmod 2$. If $d_0(t) = d'_0(t)$, we must have $d_0(t+1) = d'_0(t+1) > d'_0(t)$, hence $d_0(t) = d'_0(t) \equiv t \pmod 2$.

Since $d_0(t) < d'_0(t+1)$, we have $nA^{d_0(t)}\nu^{t+1} < C$, hence

$$\begin{aligned} b(t, d_0(t)) + c(t, i) &= T(t, i) - 4s(t, d_0(t) + 2) < T'(t) - 4s'(t, d_0(t) + 2) + 1 \\ &= b'(t, d_0(t)) + c'(t) + 1 = nA^{d_0(t)}\nu^t + 1 < 1 + C\nu^{-1}. \end{aligned}$$

□

Lemma 4.3 (*in* VNC^1_*) *If* $d_0 \leq d \leq d_1$, $d \equiv t \pmod 2$, *and* $i < 2^{d_0}$, *then* $b(t, d) > 0$ *and* $c(t, i) > 0$.

Proof: Since $d_0(t) \geq d'_0(t-1)$, we have $nA^{d_0}\nu^{t-1} \geq C$. As $s(t, d) < s'(t, d) + 2$, we obtain

$$\begin{aligned} b(t, d) &= s(t, d) - 4s(t, d+2) > s'(t, d) - 4s'(t, d+2) - 8 = b'(t, d) - 8 \\ &= \left(1 - \frac{1}{4A^2}\right) nA^d\nu^t - 8 \geq \left(1 - \frac{1}{4A^2}\right) nA^{d_0}\nu^t - 8 \geq \left(1 - \frac{1}{4A^2}\right) C\nu - 8 \geq 0. \end{aligned}$$

If $d_0 \equiv t \pmod 2$, we have

$$c(t, i) = T(t, i) - s(t, d_0) > c'(t) - 3 = \frac{n}{4A^2} A^{d_0}\nu^t - 3 \geq \frac{C\nu}{4A^2} - 3 \geq 0.$$

Similarly, if $d_0 \not\equiv t \pmod 2$, then

$$c(t, i) > \frac{C\nu}{2A} - 5 \geq 0.$$

□

Lemma 4.4 *(in VNC^1_*) There exists*

$$t_m = \min\{t > 0 \mid d_0(t) = d_1(t)\},$$

which satisfies $t_m \le c_m \lceil \log n \rceil$. We have $d_0(t) < d_1(t)$ for all $0 < t < t_m$. Moreover, $d'_1(t) \le \lfloor \log n \rfloor$ for all $t \le t_m$, and $T(t_m, i) \le \lceil C/\nu \rceil$.

Proof: Put $t = c_m \lceil \log n \rceil$ and $d = d'_0(t)$. As $\nu^{-c_m} \ge 2A$, we have

$$C \le nA^d \nu^t \le nA^d (2A)^{-\lceil \log n \rceil} \le A^{d - \lceil \log n \rceil},$$

thus $d \ge \lceil \log n \rceil$ and $2^d \ge n$. This implies

$$(2A)^{d-2} \nu^t \ge \frac{1}{4A^2} nA^d \nu^t \ge \frac{C}{4A^2} \ge 1,$$

hence $d'_1(t) < d'_0(t)$, and $d_1(t) \le d_0(t)$.

On the other hand, $d_1(0) = 1 > d_0(0) = 0$ from Lemma 4.1 and $n\nu \ge C$, hence there exists

$$t_m := \min\{t > 0 \mid d_1(t) \le d_0(t)\} \le c_m \lceil \log n \rceil.$$

We have $d_0(t_m - 1) < d_1(t_m - 1)$. By Lemmas 4.1 and 4.2 we obtain $d_0(t_m) = d_1(t_m)$ unless $d_0(t_m - 1) = d_1(t_m)$, $d_1(t_m - 1) = d_0(t_m) = d_1(t_m) + 1$. But then $t_m - 1 \equiv d_0(t_m - 1) = d_1(t_m) \equiv t_m \pmod 2$, a contradiction.

As $d_0(t_m) = d_1(t_m)$, we have

$$T(t_m, i) = c(t_m, i) + s(t_m, d_0) \le c(t_m, i) + b(t_m, d_0) \le \lceil C/\nu \rceil$$

by Lemma 4.2. Finally, $\lfloor n 2^{-d_0(t_m)} \rfloor = T(t_m, 0) = s(t_m, d_0(t_m)) + c(t_m, 0) \ge 2$ by Lemma 4.3, hence $d'_1(t) \le d_1(t_m) + 1 = d_0(t_m) + 1 \le \lfloor \log n \rfloor$ for any $t \le t_m$. □

The Lemma below implies, among others, that Case 4 makes sense.

Lemma 4.5 *(in VNC^1_*) If $d_0(t+1) \le d < d_1(t)$ and $d \not\equiv t \pmod 2$, then $s(t+1, d) > 2s(t, d+1)$.*

Proof: We have

$$\begin{aligned} s(t+1,d) - 2s(t,d+1) &\ge s'(t+1,d) - 2s'(t,d+1) - 4 \\ &= \frac{n}{2^d}\left(1 - (2A)^{d-2}\nu^{t+1} - 1 + (2A)^{d-1}\nu^t\right) - 4 \\ &= \frac{n}{4A^2} A^d \nu^t (2A - \nu) - 4 \ge \frac{C}{4A^2}(2A - \nu) - 4 > 0. \end{aligned}$$

□

The following Lemma ensures that the $\langle l, \varepsilon, \varepsilon_0 \rangle$-separator in Cases 1 and 2 is used correctly.

Lemma 4.6 *(in VNC^1_*) Let $d_0(t) \le d < d_1(t+1)$, $d \equiv t \pmod 2$, $m = h(t,d)$, and $l = s(t,d) - 2s(t+1, d+1)$. Then $m \ge l \ge m\lambda_0$.*

Proof: Recall that $b'(t,d) = \bigl(1-(4A^2)^{-1}\bigr)nA^d\nu^t$. Put $l' = s'(t,s) - 2s'(t+1,d+1)$. We have

$$\begin{aligned} l' &= \frac{n}{2^d}\left(1-(2A)^{d-2}\nu^t\right) - \frac{n}{2^d}\left(1-(2A)^{d-1}\nu^{t+1}\right) = \frac{n}{2^d}(2A)^{d-2}\nu^t(2A\nu-1) \\ &= nA^d\nu^t\frac{1}{4A^2}(4A^2\lambda+1-\lambda-1) = \left(1-\frac{1}{4A^2}\right)nA^d\nu^t\lambda = \lambda b'(t,d). \end{aligned}$$

If $d = d_1(t)$, then $l \le s(t,d) = m$. Otherwise

$$m - l = 2s(t+1,d+1) - 4s(t,d+2) > 0$$

by Lemma 4.5.

For the other inequality, we have

$$\begin{aligned} \frac{l}{m} \ge \frac{l}{b(t,d)} &= 1 - \frac{2s(t+1,d+1)-4s(t,d+2)}{s(t,d)-4s(t,d+2)} \ge 1 - \frac{2s(t+1,d+1)-4s(t,d+2)}{s'(t,d)-4s(t,d+2)} \\ &= \frac{s'(t,d)-2s(t+1,d+1)}{s'(t,d)-4s(t,d+2)} \ge \frac{s'(t,d)-2s'(t+1,d+1)-4}{s'(t,d)-4s'(t,d+2)} \\ &= \frac{l'-4}{b'(t,d)} \ge \lambda - \frac{4}{\bigl(1-(4A^2)^{-1}\bigr)C\nu} \ge \lambda_0. \end{aligned}$$

□

The next Lemma shows that the splitting in Case 3 make sense.

Lemma 4.7 (*in* VNC^1_*) *Let* $t < t_m$ *be such that* $d_0(t) < d_0(t+1)$, *and* $i < 2^{d_0(t)}$. *Put* $x_\alpha = T(t+1, 2i+\alpha) - 2s(t, d_0(t)+2)$ *for* $\alpha = 0,1$. *Then* $x_0 + x_1 = b(t, d_0(t)) + c(t,i)$ *and* $x_\alpha \ge c(t+1, 2i+\alpha)$.

Proof: As $d_0(t+1) = d_0(t)+1$, we have

$$\begin{aligned} T(t+1,2i) + T(t+1,2i+1) &= \lfloor (2i+2)n2^{-d_0(t+1)} \rfloor - \lfloor 2in2^{-d_0(t+1)} \rfloor \\ &= \lfloor (i+1)n2^{-d_0(t)} \rfloor - \lfloor in2^{-d_0(t)} \rfloor = T(t,i). \end{aligned}$$

Then clearly

$$b(t,d_0(t)) + c(t,i) = T(t,i) - 4s(t,d_0(t)+2) = x_0 + x_1.$$

Since $d_0(t+1) \equiv t+1 \pmod 2$, we have

$$x_\alpha - c(t+1,2i+\alpha) = s(t+1,d_0(t)+1) - 2s(t,d_0(t)+2) > 0$$

by Lemma 4.5. □

Lemma 4.8 (*in* VNC^1_*) *Let* $t < t_m$, $d_0(t+1) \le d \le d_1(t+1)$, $d \equiv t+1 \pmod 2$. *Then the total number of elements sent from stage* t *to any bag of level* d *is* $h(t+1,d)$. *If* $i < 2^{d_0(t+1)}$, *the number of elements sent to the ith cold storage is* $c(t+1,i)$.

Proof: We start with the cold storage. If $d_0(t) < d_0(t+1)$, the cold storage gets $c(t+1,i)$ elements by Case 3. Let thus $d_0(t) = d_0(t+1)$. If $d_0 \not\equiv t \pmod 2$, there remain

$$c(t,i) - (s(t+1,d_0) - 2s(t,d_0+1)) = T(t,i) - s(t+1,d_0) = c(t+1,i)$$

elements in the cold storage by Case 4. Otherwise, we get

$$c(t,i) + (s(t,d_0) - 2s(t+1,d_0+1)) = T(t,i) - 2s(t+1,d_0+1) = c(t+1,i)$$

elements by Case 2.

Now we turn to regular bags. First assume $d > d_1(t)$, hence $d = d_1(t+1)$. We get

$$\tfrac{1}{2}\big(s(t,d-1) - (s(t,d-1) - 2s(t+1,d))\big) = s(t+1,d) = h(t+1,d)$$

elements from the parent by Case 1 or 2. Let thus assume $d < d_1(t)$.

If $d > d_0(t+1)$, we obtain

$$\tfrac{1}{2}\big(b(t,d-1) - (s(t,d-1) - 2s(t+1,d))\big) = s(t+1,d) - 2s(t,d+1)$$

elements from the parent by Case 1 or 2. If $d = d_0(t+1) = d_0(t)$, we get $s(t+1,d) - 2s(t,d+1)$ elements from cold storage by Case 4. If $d = d_0(t+1) > d_0(t)$, we get

$$T(t+1,i) - 2s(t,d+1) - c(t+1,i) = s(t+1,d) - 2s(t,d+1)$$

elements from splitting of the parent by Case 3. Thus, in all cases, the bag obtains

$$s(t+1,d) - 2s(t,d+1)$$

elements “from above”.

If $d = d_1(t+1)$, then we obtain $s(t,d+1)$ elements from each child by Case 1, hence we get $s(t+1,d) = h(t+1,d)$ elements in total.

If $d < d_1(t+1)$, then we cannot have $d+1 = d_1(t) > d_1(t+1)$. We thus obtain $s(t,d+1) - 2s(t+1,d+2)$ elements from each child by Case 1. We have

$$\begin{aligned} s(t+1,d) - 2s(t,d+1) + 2(s(t,d+1) - 2s(t+1,d+2)) \\ = s(t+1,d) - 4s(t+1,d+2) = b(t+1,d) = h(t+1,d) \end{aligned}$$

elements in total. □

Having checked that the network is coherently defined, we turn our attention to its behaviour when evaluated. In order to simplify the analysis, we first consider the special case when the input is a permutation of the sequence $0,\dots,n-1$ (the most important point being that the inputs are pairwise distinct), and $\le$ is the usual ordering. We fix an evaluation of the network on such input. (Strictly speaking, we only defined evaluation of a network on set inputs, not number inputs. We can encode numbers $k < n$ by sets in a straightforward way, e.g., by $\{k\}$.)

We associate with each bag B its *natural interval* in $[0,n)$: the ith bag on level d in the left-to-right order corresponds to the interval

$$I(d,i) = \big[\lfloor in2^{-d}\rfloor, \lfloor (i+1)n2^{-d}\rfloor\big).$$

An element x of the bag B whose value is outside $I(d,i)$ is called a *stranger*, and its *strangeness* is defined as the smallest number j such that x belongs to the natural interval of B's ancestor on level $d-j$, i.e., $I(d-j,\lfloor i2^{-j}\rfloor)$. We let $S_j(t,d,i)$ denote the number of elements of B at stage t of strangeness at least j. Let $\xi(t)$ denote the NC^1_*-formula which is the conjunction of the following conditions:

(i) For every $i<2^{d_0}$, the values of all elements of the ith tree (including its cold storage) at stage t belong to $I(d_0,i)$.

(ii) For every d,i,j such that $d_0\le d\le d_1$, $i<2^d$, and $0<j\le d-d_0$, we have

$$S_j(t,d,i)\le \mu\delta^j b'(t,d),$$

where we put $\mu=10$, $\delta=1/270$.

Lemma 4.9 (*in VNC^1_**) *If $\xi(t)$ holds, then condition* (i) *of $\xi(t+1)$ also holds.*

Proof: If $d_0(t)=d_0(t+1)$, the conclusion is trivial, as element movements respect tree boundaries. Let us thus assume $d_0(t+1)=d_0(t)+1$, and denote $d_0=d_0(t)$ for short. Fix $i<2^{d_0}$. We know by $\xi(t)$ that all elements of the $2i$th and the $(2i+1)$th tree at stage $t+1$, which come from the ith tree at stage t, belong to $I(d_0,i)=I(d_0+1,2i)\mathbin{\dot\cup} I(d_0+1,2i+1)$.

Consider $d>d_0$ such that $d\equiv t \pmod 2$, and $i'<2^d$ such that $\lfloor i'2^{d_0-d}\rfloor=i$. Note that $d\ge d_0+2$. Since $A\delta\le 1$, we have

$$\begin{aligned}S_{d-d_0}(t,d,i')&\le \mu\delta^{d-d_0}b'(t,d)=\mu\delta^{d-d_0}A^{d-d_0}b'(t,d_0)\le\mu(A\delta)^2b'(t,d_0)\\&=\mu(A\delta)^2\left(1-\frac{1}{4A^2}\right)nA^{d_0}\nu^t<\mu(A\delta)^2\left(1-\frac{1}{4A^2}\right)C\nu^{-1}\le 1,\end{aligned}$$

using $\xi(t)$, and $nA^{d_0}\nu^{t+1}<C$, which follows from $d_0<d_0'(t+1)$. This means that every element of the i'th bag on level d at stage t has strangeness less than $d-d_0$, i.e., it belongs to the interval $I(d_0+1,\lfloor i'2^{d_0+1-d}\rfloor)$. On the other hand, these elements end up in the $\lfloor i'2^{d_0+1-d}\rfloor$th tree at stage $t+1$, as required.

The remaining elements of the $2i$th and $(2i+1)$th tree at stage $t+1$ come from the root and cold storage of the ith tree at stage t. We know from above that there are exactly $2s(t,d_0+2)$ elements of $I(d_0+1,2i)$ and $2s(t,d_0+2)$ elements of $I(d_0+1,2i+1)$ in the rest of the ith tree at stage t. Since $I(d_0+1,2i)$ and $I(d_0+1,2i+1)$ have $T(t+1,2i)$ and $T(t+1,2i+1)$ elements in total, respectively, the root and cold storage of the ith tree at stage t contain $T(t+1,2i)-2s(t,d_0+2)$ elements of $I(t+1,2i)$, and $T(t+1,2i+1)-2s(t,d_0+2)$ elements of $I(t+1,2i+1)$. By Case 3 of the definition of the network, we send the smallest $T(t+1,2i)-2s(t,d_0+2)$ of these elements to the $2i$th tree at stage $t+1$, and the largest $T(t+1,2i+1)-2s(t,d_0+2)$ elements to the $(2i+1)$th tree. As elements of $I(t+1,2i)$ are smaller than elements of $I(t+1,2i+1)$, all these elements end up in the correct tree. □

Lemma 4.10 (*in VNC^1_**) *If $\xi(t)$ holds, $d_0(t+1)\le d\le d_1(t+1)$, $d\equiv t+1 \pmod 2$, $i<2^d$, and $2\le j\le d-d_0(t+1)$, then $S_j(t+1,d,i)\le\mu\delta^j b'(t+1,d)$.*

Proof: Note that $d_0(t+1) < d-1$. Denote by B the ith bag on level d. Elements of B of strangeness j or more at stage $t+1$ come from two sources: elements of B's children at stage t of strangeness at least $j+1$, and elements of B's parent at stage t of strangeness at least $j-1$, both using Case 1 of the definition of the network.

Using $\xi(t)$, the number of elements of B's children with strangeness $j+1$ or more is at most

$$2\mu\delta^{j+1}b'(t,d+1). \tag{3}$$

Let P be B's parent. The number of elements of P of strangeness $j-1$ or more at stage t is

$$k := S_{j-1}(t,d-1,\lfloor i/2\rfloor) \le \mu\delta^{j-1}b'(t,d-1).$$

Let a be the number of elements of P whose value is smaller than x, and b the number of elements of P whose value is at least y, where $I(d-j,\lfloor i2^{-j}\rfloor) = [x,y)$, so that $a+b=k$. Put

$$l = s(t,d-1) - 2s(t+1,d).$$

By Case 1, we apply to P's content an $\langle l,\varepsilon,\varepsilon_0\rangle$-separator S, and send the part $CL \cup CR$ of its output to B.

Notice that $b'(t,d-1) \ge \bigl(1-(4A^2)^{-1}\bigr)C\nu$ by the proof of Lemma 4.3. Using the proof of Lemma 4.6, we obtain

$$\begin{aligned} l &\ge s'(t,d-1) - 2s'(t+1,d) - 4 = \lambda b'(t,d-1) - 4 \\ &\ge \left(\lambda - \frac{4}{\bigl(1-(4A^2)^{-1}\bigr)C\nu}\right) b'(t,d-1) \ge 2\mu\delta\, b'(t,d-1) \ge 2k, \end{aligned}$$

hence $a,b \le l/2$. Let $F(u) \in \{0,1\}$ be defined by $F(u)=1$ iff $u \ge x$. The application of F to the elements of P gives a 0–1 sequence with a zeros. If we evaluate S on this input, at most εa zeros end up outside FL by Definition 3.6. Using Lemma 2.4, the application of S to P sends at most εa elements smaller than x to $CL \cup CR \cup FR$. By a similar argument, at most εb elements greater than or equal to y end up in $CL \cup CR \cup FL$. In total, the number of elements outside $I(d-j,\lfloor i2^{-j}\rfloor)$ sent from P to B is at most

$$\varepsilon a + \varepsilon b = \varepsilon k \le \varepsilon\mu\delta^{j-1}b'(t,d-1). \tag{4}$$

Putting (3) and (4) together, we see that at stage $t+1$, B contains at most

$$2\mu\delta^{j+1}b'(t,d+1) + \varepsilon\mu\delta^{j-1}b'(t,d-1) = \left(\frac{2A\delta}{\nu} + \frac{\varepsilon}{A\delta\nu}\right)\mu\delta^j b'(t+1,d) \le \mu\delta^j b'(t+1,d)$$

elements of strangeness j or more. □

Lemma 4.11 (*in* VNC^1_*) *If* $\xi(t)$ *holds,* $d_0(t+1) < d \le d_1(t+1)$*,* $d \equiv t+1 \pmod 2$*, and* $i < 2^d$*, then* $S_1(t+1,d,i) \le \mu\delta\, b'(t+1,d)$*.*

Proof: Let B be the ith bag on level d, P its parent, and B' its sibling. Let $i' = i + (-1)^i$ be the label of B', and put $I = I(d,i)$, $I' = I(d,i')$.

Strangers in B at stage $t+1$ come from two sources: elements of strangeness at least 2 in B's children at stage t, of whom there are at most

$$2\mu\delta^2 b'(t,d+1) = 2A\mu\delta^2 b'(t,d), \tag{5}$$

and elements of P at stage t sent downwards to B which are either strangers in P, or belong to I'.

The number of elements of the subtree below B' at stage t which do not belong to I' is

$$\begin{aligned}\sum_{\substack{j=1\\ j\text{ odd}}}^{d_1-d}\ \sum_{\hat\imath=2^j i'}^{2^j(i'+1)-1} S_{j+1}(t,d+j,\hat\imath) &\le \sum_{\substack{j=1\\ j\text{ odd}}}^{d_1-d} 2^j\mu\delta^{j+1}A^j b'(t,d)\\ &= 2A\mu\delta^2 b'(t,d)\sum_{k=0}^{(d_1-d-1)/2}(2A\delta)^k\\ &\le \frac{2A\mu\delta^2}{1-4A^2\delta^2}b'(t,d) =: \alpha b'(t,d).\end{aligned}$$

This subtree thus contains at least $2s(t,d+1) - \alpha b'(t,d)$ elements of I', hence P contains

$$\begin{aligned}x &\le \operatorname{card} I' - 2s(t,d+1) + \alpha b'(t,d) \le 1 + \alpha b'(t,d) + \frac{n}{2^d} - 2s'(t,d+1)\\ &= 1 + \alpha b'(t,d) + \frac{n}{2^d}(2A)^{d-1}\nu^t = 1 + \left(\alpha + \frac{2A}{4A^2-1}\right) b'(t,d)\end{aligned}$$

elements of I'. P also contains

$$y + z \le \mu\delta\, b'(t,d-1) = \frac{\mu\delta}{A} b'(t,d)$$

strangers, where y is the number of elements below $\min(I\cup I')$, and z the number of elements above $\max(I\cup I')$.

Assume that i is even (i.e., $I < I'$); the other case is symmetric. Remember that we apply a $\langle l,\varepsilon,\varepsilon_0\rangle$-separator (hence an ε_0-halver) to P, and send the content of CL to B. Let c be the element of P which splits it in half, i.e., there are $\frac{1}{2}b(t,d-1)$ elements of P greater than c. Define $F(u)\in\{0,1\}$ by $F(u)=1$ iff $u>c$. By Definition 3.3 and Lemma 2.4, there are at most $(\varepsilon_0/2)b(t,d-1)$ elements of P greater than c which end up in $FL\cup CL$. Furthermore, there are at most $\max\bigl\{0, x+z-\frac{1}{2}b(t,d-1)\bigr\}$ elements greater than $\max I$ below c, and y elements smaller than $\min I$. The total number of elements outside I in CL is thus bounded by

$$\begin{aligned}
y + \max\bigl\{0, x + z - \tfrac{1}{2}b(t,d-1)\bigr\} + \frac{\varepsilon_0}{2}b(t,d-1) \\
&\le 1 + \left(\alpha + \frac{2A}{4A^2-1} + \frac{\mu\delta}{A}\right) b'(t,d) - \frac{1-\varepsilon_0}{2}b(t,d-1) \\
&\le 5 + \left(\alpha + \frac{2A}{4A^2-1} + \frac{\mu\delta}{A}\right) b'(t,d) - \frac{1-\varepsilon_0}{2}b'(t,d-1) \\
&= 5 + \left(\alpha + \frac{2A}{4A^2-1} + \frac{\mu\delta}{A} - \frac{1-\varepsilon_0}{2A}\right) b'(t,d) \\
&\le \left(\alpha + \frac{1}{2A(4A^2-1)} + \frac{\mu\delta}{A} + \frac{\varepsilon_0}{2A} + \frac{5}{\bigl(1-(4A^2)^{-1}\bigr)AC\nu}\right) b'(t,d),
\end{aligned}$$

as $b'(t,d-1) \ge \bigl(1-(4A^2)^{-1}\bigr)C\nu$ by the proof of Lemma 4.3. Combining this with (5), we see that the number of strangers in B at stage $t+1$ is at most

$$\begin{aligned}
\left(\frac{2A\mu\delta^2}{1-4A^2\delta^2} + \frac{1}{2A(4A^2-1)} + \frac{\mu\delta}{A} + \frac{\varepsilon_0}{2A} + \frac{5}{\bigl(1-(4A^2)^{-1}\bigr)AC\nu} + 2A\mu\delta^2\right) b'(t,d) \\
\le \mu\delta\nu\, b'(t,d) = \mu\delta\, b'(t+1,d).
\end{aligned}$$

□

Theorem 4.12 *Under Assumption 3.2, there exists an* NC^1_**-function* $N(n)$*, and a constant* c *such that* VNC^1_* *proves the following:*

For every $n > 0$*,* $N(n)$ *is a comparator network on* n *inputs of depth at most* $c \log n$*. If* $\le$ *is a total ordering defined by an* NC^1_**-formula, and* $\langle X_i \mid i < n\rangle$ *a sequence of sets in the domain of* $\le$*, then there exists a permutation* π *of* n *such that* $\mathrm{eval}(N(n), \le, \vec{X}) = \langle X_{\pi(i)} \mid i < n\rangle$*, and* $X_{\pi(i)} \le X_{\pi(j)}$ *for every* $i \le j < n$*.*

Proof: If $n \le C/\nu$, we let $N(n)$ be any sorting network on n inputs, otherwise we define $N(n)$ as the network described in Section 3. Clearly, $N(n)$ is a comparator network on n inputs of depth at most $c_m(p+1)D^2\lceil \log n\rceil + O(1)$.

First, let $\vec{X}$ be a permutation of $\langle 0, \dots, n-1\rangle$, and $\le$ the usual ordering. For every $t < t_m$, $\xi(t)$ implies $\xi(t+1)$ by Lemmas 4.9, 4.10, and 4.11, and $\xi(0)$ holds trivially. Using induction, we obtain $\xi(t_m)$. By condition (i), each of the 2^{d_0} constant-size trees at stage t_m contains elements of its corresponding subinterval of $[0,n)$, hence after the final application of sorting subnetworks on the trees, the result is fully sorted.

In the general case, we pick a permutation π on n such that $X_{\pi(i)} \le X_{\pi(j)}$ for each $i \le j$ by Lemma 2.6. Put $x_i = \pi^{-1}(i)$, and $F(i) = X_{\pi(i)}$. Clearly $F(\vec{x}) = \vec{X}$, and $\mathrm{eval}(N(n), \le, \vec{x}) = \langle 0, \dots, n-1\rangle$ by the first part of the proof, hence $\mathrm{eval}(N(n), \le, \vec{X}) = \langle F(0), \dots, F(n-1)\rangle = \langle X_{\pi(i)} \mid i < n\rangle$ by Lemma 2.4. □

5 Monotone sequent calculus

The monotone sequent calculus *MLK* is the fragment of the usual Gentzen propositional sequent calculus *LK* where we allow only sequents consisting of monotone formulas, i.e., propositional formulas built using the connectives $\{\wedge, \vee, \bot, \top\}$. The calculus thus uses structural rules, the initial rule (axiom), the cut rule, and left and right introduction rules for $\wedge$, $\vee$, $\bot$, and $\top$. Its introduction was originally motivated by results in circuit complexity [156, 9] showing exponential lower bounds on the size of monotone circuits; the hope was that these can be transformed to an exponential separation between *MLK* and *LK*. Atserias, Galesi, and Pudlák [14] proved that this is not the case, as *MLK* quasipolynomially simulates *LK*:

Theorem 5.1 ([14]) *A monotone sequent in n variables which has an LK-proof of size s has also an MLK-proof of size $s^{O(1)}n^{O(\log n)}$ with $s^{O(1)}$ lines.* □

It remains an open problem (called the *Think Positively Conjecture* by Atserias [12]) whether we can improve this quasipolynomial simulation to a p-simulation, i.e., whether there exists a polynomial-time algorithm transforming an *LK*-proof of a monotone sequent to an *MLK*-proof of the same sequent. Atserias, Galesi, and Pudlák [14] suggested the following approach to attack the problem, relying on a construction of suitable monotone formulas for the threshold functions

$$\theta^n_m(x_0,\dots,x_{n-1}) = 1 \Leftrightarrow \operatorname{card}\{i \mid x_i = 1\} \geq m.$$

Theorem 5.2 ([14]) *Assume that there are monotone formulas $T^n_m(p_0,\dots,p_{n-1})$ for $m \leq n+1$ such that the formulas*

$$T^n_0(p_0,\dots,p_{n-1}) \tag{6}$$

$$\neg T^n_{n+1}(p_0,\dots,p_{n-1}) \tag{7}$$

$$T^n_m(p_0,\dots,p_{k-1},\bot,p_{k+1},\dots,p_{n-1}) \to T^n_{m+1}(p_0,\dots,p_{k-1},\top,p_{k+1},\dots,p_{n-1}) \tag{8}$$

for $m \leq n$, $k < n$ have LK-proofs constructible in time $n^{O(1)}$. Then MLK p-simulates LK-proofs of monotone sequents. □

A remarkable feature of Theorem 5.2 is that in the conclusion we construct *MLK*-proofs from *LK*-proofs, nevertheless in the assumption we only require the existence of *LK*-proofs. This significantly broadens the range of methods admissible for proving (6)–(8), and in particular, we can use propositional translations of proofs in bounded arithmetic.

Recall that the sequent calculus *LK* is p-equivalent to Frege systems: these are proof systems given by a sound and implicationally complete finite set of rules of the form $\varphi_1,\dots,\varphi_n/\varphi$, such that a Frege proof of φ is a sequence of formulas ending with φ where each formula is derived from previous formulas by a substitution instance of a basic rule. As shown in §V.5, NC^1_*-formulas provable in VNC^1_* translate to families of propositional tautologies with polynomial-time Frege proofs. The translation works as follows. For each NC^1_*-formula $\varphi(x_1,\dots,x_r,X_1,\dots,X_s)$ (i.e., $\varphi \in \Sigma^B_0(L_{\overline{VNC^1_*}})$) and natural numbers $n_1,\dots,n_r,m_1,\dots,m_s$, we define a propositional formula

$$[\![\varphi(\vec{x},\vec{X})]\!]_{\vec{n},\vec{m}}(p_{1,0},\dots,p_{1,m_1-1},\dots,p_{s,0},\dots,p_{s,m_s-1}).$$

Let $X_1,\dots,X_s$ be sets such that $|X_i|\le m_i$, and let $\tilde X_i$ denote the propositional assignment which gives the value 1 to the variable $p_{i,k}$ iff $k\in X_i$. Then the translation satisfies

$$(*)\qquad [\![\varphi]\!]_{\vec n,\vec m}(\tilde X_1,\dots,\tilde X_s)=1\Leftrightarrow \mathbb N\vDash\varphi(\vec n,\vec X).$$

In particular, if (the universal closure of) φ is valid in $\mathbb N$, then $[\![\varphi]\!]_{\vec n,\vec m}$ is a sequence of tautologies. The translation of compound formulas is defined by

$$\begin{aligned}
[\![\varphi\circ\psi]\!]_{\vec n,\vec m}&=[\![\varphi]\!]_{\vec n,\vec m}\circ[\![\psi]\!]_{\vec n,\vec m},\qquad \circ\in\{\wedge,\vee,\neg\},\\
[\![\exists x\le t\,\varphi]\!]_{\vec n,\vec m}&=\bigvee_{k\le b_t(\vec n,\vec m)}([\![x\le t]\!]_{k,\vec n,\vec m}\wedge[\![\varphi]\!]_{k,\vec n,\vec m}),\\
[\![\forall x\le t\,\varphi]\!]_{\vec n,\vec m}&=\bigwedge_{k\le b_t(\vec n,\vec m)}([\![x\le t]\!]_{k,\vec n,\vec m}\to[\![\varphi]\!]_{k,\vec n,\vec m}),
\end{aligned}$$

where b_t is a suitable L_0-term such that $t(\vec n,\vec X)\le b_t(\vec n,\vec m)$ whenever $|X_i|\le m_i$ for each i. The definition of $[\![\varphi]\!]$ for atomic formulas φ is more tedious and involves translation of terms as well as formulas, but it proceeds in a more-or-less expected way, we refer the reader to §V.5 for details.

Theorem 5.3 (Thm. V.5.2) *If φ is an NC^1_*-formula such that $VNC^1_*\vdash\varphi$, then the tautologies $[\![\varphi]\!]_{\vec n,\vec m}$ have Frege proofs constructible in time* $\mathrm{poly}(\vec n,\vec m)$. □

Sorting a 0–1 input amounts to counting the number of ones, hence the AKS network evaluated on a 0–1 input gives monotone circuits for threshold functions of logarithmic depth, which can be unwinded into polynomial-size formulas. (We mention here that there is also an elegant simple construction of monotone polynomial-size formulas for threshold functions by Valiant [175]. Unfortunately, this construction is probabilistic, hence it does not give concrete formulas with any hope of being formalizable by short Frege proofs.) Since fundamental properties of the network are provable in VNC^1_*, we can use Theorem 5.3 to construct polynomial-time Frege proofs of (6)–(8). We proceed with the details.

Let $N(n)$ be the NC^1_*-function computing a log-depth sorting network as in Theorem 4.12. Let $\varphi(n,e,f,h,l)$ be an NC^1_*-formula expressing that there exists a comparator in $N(n)$ whose input edges are e,f, and whose higher and lower output edges are h and l, respectively. Using Lemma 2.3, there is an NC^1_*-formula $\psi(n,e,X)$ expressing that edge e in $N(n)$ evaluates to 1 on a 0–1 input X. Finally, let $\chi(n,i,X)$ denote the NC^1_*-formula $i\in\mathrm{eval}(N(n),\le,X)$. We define a monotone propositional formula $A_{n,e}$ for each edge e of $N(n)$ as follows. If e is the outgoing edge of the ith input node, we put

$$A_{n,e}=p_i.$$

If $\varphi(n,e,f,h,l)$, we define

$$\begin{aligned}
A_{n,h}&=A_{n,e}\vee A_{n,f},\\
A_{n,l}&=A_{n,e}\wedge A_{n,f}.
\end{aligned}$$

Notice that the depth of $A_{n,e}$ is the depth of e in $N(n)$, which is bounded by $O(\log n)$, thus $A_{n,e}$ has polynomial size. If $0 < m \leq n$, and e is the incoming edge of the $(n-m)$th output node of $N(n)$, we put

$$T^n_m = A_{n,e}.$$

We also define

$$\begin{aligned} T^n_{n+1} &= \bot, \\ T^n_0 &= \top. \end{aligned}$$

Lemma 5.4 *There are polynomial-time Frege proofs of the formulas*

$$T^n_m \leftrightarrow [\![m \leq \operatorname{card} X]\!]_{m,n}.$$

Proof: As VNC^1_* proves the formula

$$\begin{aligned} \alpha = \varphi(n,e,f,h,l) \to [&(\psi(n,h,X) \leftrightarrow \psi(n,e,X) \vee \psi(n,f,X)) \\ &\wedge (\psi(n,l,X) \leftrightarrow \psi(n,e,X) \wedge \psi(n,f,X))], \end{aligned}$$

its translation $[\![\alpha]\!]_{n,e,f,h,l,n}$ has polynomial-time Frege proofs. If e, f, h, l are the respective input and output edges of a comparator in $N(n)$, then $[\![\varphi]\!]_{n,e,f,h,l}$ is a true Boolean sentence, hence it has a polynomial-time Frege proof. We obtain proofs of the formulas

$$\begin{aligned} &[\![\psi]\!]_{n,h,n} \leftrightarrow [\![\psi]\!]_{n,e,n} \vee [\![\psi]\!]_{n,f,n}, \\ &[\![\psi]\!]_{n,l,n} \leftrightarrow [\![\psi]\!]_{n,e,n} \wedge [\![\psi]\!]_{n,f,n}. \end{aligned}$$

If e is the outgoing edge of the ith input node in $N(n)$, and f is the incoming edge of the ith output node, we can similarly construct proofs of

$$\begin{aligned} &[\![\psi]\!]_{n,e,n} \leftrightarrow p_i, \\ &[\![\psi]\!]_{n,f,n} \leftrightarrow [\![\chi(n,i,X)]\!]_{n,i,n}. \end{aligned}$$

Then we can construct proofs of

$$A_{n,e} \leftrightarrow [\![\psi]\!]_{n,e,n}$$

by induction on the depth of e, and we derive

$$T^n_m \leftrightarrow [\![\chi]\!]_{n,n-m,n}$$

for $0 < m \leq n$. By Theorem 4.12 and the proof of Lemma 2.6, VNC^1_* proves

$$|X| \leq n \to (\chi(n,i,X) \leftrightarrow \operatorname{card} X \geq n-i).$$

As there are short proofs of $[\![|X| \leq n]\!]_{n,n}$, we obtain short proofs of

$$[\![\chi]\!]_{n,n-m,n} \leftrightarrow [\![\operatorname{card} X \geq m]\!]_{m,n},$$

and we conclude

$$T^n_m \leftrightarrow [\![\mathrm{card}\, X \geq m]\!]_{m,n}$$

for $0 < m \leq n$. The cases $m = 0$ and $m = n + 1$ follow from translation of the formulas

$$\mathrm{card}\, X \geq 0,$$
$$|X| \leq n \to \neg(\mathrm{card}\, X \geq n + 1),$$

provable in VNC^1_*. □

Theorem 5.5 *Under Assumption 3.2, the monotone sequent calculus MLK p-simulates LK-proofs of monotone sequents.*

Proof: In view of Theorem 5.2, it suffices to construct polynomial-time Frege proofs of (6)–(8) for the formulas T^n_m defined above. (6) and (7) are trivial. VNC^1_* proves

$$\forall u < n\, (u \neq k \wedge X(u) \to Y(u)) \wedge \neg X(k) \wedge Y(k) \wedge |Y| \leq n \\ \to (m \leq \mathrm{card}\, X \to m + 1 \leq \mathrm{card}\, Y),$$

thus its (slightly simplified) propositional translation

$$\bigwedge_{\substack{u<n\\ u\neq k}} (p_u \to q_u) \wedge \neg p_k \wedge q_k \to ([\![u \leq \mathrm{card}\, X]\!]_{m,n}(\vec{p}) \to [\![u \leq \mathrm{card}\, X]\!]_{m+1,n}(\vec{q}))$$

for $n \in \omega$, $m \leq n$, and $k < n$ has poly-time constructible Frege proofs. We substitute $\bot$ for p_k, $\top$ for q_k, and p_u for q_u, $u \neq k$, in the proof. We obtain

$$[\![u \leq \mathrm{card}\, X]\!]_{m,n}(p_0, \dots, p_{k-1}, \bot, p_{k+1}, \dots, p_{n-1}) \\ \to [\![u \leq \mathrm{card}\, X]\!]_{m+1,n}(p_0, \dots, p_{k-1}, \top, p_{k+1}, \dots, p_{n-1}),$$

from which we derive

$$T^n_m(p_0, \dots, p_{k-1}, \bot, p_{k+1}, \dots, p_{n-1}) \to T^n_{m+1}(p_0, \dots, p_{k-1}, \top, p_{k+1}, \dots, p_{n-1})$$

using Lemma 5.4. □

6 Open problems

The main problems we left open were already mentioned:

Problem 6.1 *Is Assumption 3.2 valid?*

Problem 6.2 *Does MLK p-simulate LK on monotone sequents?*

We also touched a problem of a more computational nature: as mentioned in the Introduction, the reason for using VNC^1_* instead of VNC^1 is that we do not know whether the AKS network is sufficiently uniform. In the most important 0–1 case, we can formulate it as the following problem in circuit complexity.

Definition 6.3 A language $L \subseteq \{0,1\}^*$ belongs to *uniform monotone* NC^1 (mNC^1 for short) if it satisfies any of the following conditions, which can be shown equivalent by a straightforward adaptation of the arguments by Ruzzo [162]:

(i) L is computable by a log-time alternating Turing machine whose input queries are restricted so that they force the machine to halt with the same result as the queried input bit.

(ii) L is computable by a U_E-uniform sequence of log-depth bounded fan-in monotone circuits.

(iii) L is computable by a U_{E^*}-uniform sequence of log-depth bounded fan-in monotone circuits.

(iv) L is computable by a sequence of log-depth monotone formulas, ALOGTIME-uniform in the usual infix notation.

Problem 6.4 *Is* MAJORITY *in* mNC^1*?*

As is, the AKS network only seems to provide U_{D^*}-uniform circuits for MAJORITY (where U_{D^*} is to U_D as U_{E^*} is to U_E).

Acknowledgement

The author was supported by grant IAA1019401 of GA AV ČR, grant 1M0545 of MŠMT ČR, and a grant from the John Templeton Foundation.

Chapter VII

Root finding with threshold circuits

Abstract

We show that for any constant d, complex roots of degree d univariate rational (or Gaussian rational) polynomials—given by a list of coefficients in binary—can be computed to a given accuracy by a uniform TC^0 algorithm (a uniform family of constant-depth polynomial-size threshold circuits). The basic idea is to compute the inverse function of the polynomial by a power series. We also discuss an application to the theory VTC^0 of bounded arithmetic.

1 Introduction

The complexity class TC^0 was originally defined by Hajnal, Maass, Pudlák, Szegedy, and Turán [87] in the nonuniform setting, as the class of problems recognizable by a family of polynomial-size constant-depth circuits with majority gates. It was implicitly studied before by Parberry and Schnitger [141], who consider various models of computation using thresholds (threshold circuits, Boltzmann machines, threshold RAM, threshold Turing machines). The importance of the class follows already from the work of Chandra, Stockmayer, and Vishkin [50], who show (in today's terminology) the TC^0-completeness of several basic problems (integer multiplication, iterated addition, sorting) under AC^0 reductions. Barrington, Immerman, and Straubing [19] establish that there is a robust notion of fully uniform TC^0. (We will use TC^0 to denote this uniform TC^0, unless stated otherwise.)

We can regard TC^0 as the natural complexity class of elementary arithmetical operations: integer multiplication is TC^0-complete, whereas addition, subtraction, and ordering are in $\mathrm{AC}^0 \subseteq \mathrm{TC}^0$. The exact complexity of division took some time to settle. Wallace [177] constructed division circuits of depth $O((\log n)^2)$ and bounded fan-in (i.e., NC^2). Reif [158] improved this bound to $O(\log n \log\log n)$. Beame, Cook, and Hoover [25] proved that division, iterated multiplication, and exponentiation (with exponent given in unary) are TC^0-reducible to each other, and constructed P-uniform TC^0 circuits for these problems. Chiu, Davida, and Litow [52] exhibited logspace-uniform TC^0 circuits for division, showing in particular that division is computable in L. Finally, Hesse, Allender, and Barrington [90] proved that division (and iterated multiplication) is in uniform TC^0.

Using these results, other related problems can be shown to be computable in TC^0, for example polynomial division, iterated multiplication, and interpolation. In particular, using iterated addition and multiplication of rationals, it is possible to approximate in TC^0 functions presented by sufficiently nice power series, such as log, exp, $x^{1/k}$, and trigonometric functions, see e.g. Reif [158], Reif and Tate [159], Maciel and Thérien [126], and Hesse, Allender, and Barrington [90].

Numerical computation of roots of polynomials is one of the oldest problems in mathematics, and countless algorithms have been devised to solve it, both sequential and parallel. The most popular methods are based on iterative techniques that successively derive closer and closer approximations to a root (or, sometimes, to all the roots simultaneously) starting from a suitable initial approximation. Apart from the prototypical Newton–Raphson iteration, there are for instance Laguerre's method [151, §9.5], Brent's method [151, §10.3], the Durand–Kerner method [75, 112], the Jenkins–Traub algorithm [92], and many others. One can also reduce root finding to matrix eigenvalue computation, for which there are iterative methods such as the QR algorithm [84]. Another class of root-finding algorithms are divide-and-conquer approaches: the basic idea is to recursively factorize the polynomial by identifying a suitable contour (typically, a circle) splitting the set of roots roughly in half, and recovering coefficients of the factor whose roots fall inside the contour from the residue theorem by numerical integration. Algorithms of this kind include Pan [137], Ben-Or, Feig, Kozen, and Tiwari [26], Neff [130], Neff and Reif [131], and Pan [138], see Pan [139] for an overview. These algorithms place root finding in NC: for example, the algorithm of [138] can find n-bit approximations to all roots of a polynomial of degree $d \le n$ in time $O((\log n)^2(\log d)^3)$ using $O(nd^2(\log\log n)/(\log d)^2)$ processors on an EREW PRAM. (More specifically, Allender [7] mentions that root finding is known to be in the #L hierarchy, but not known to be in GapL.)

The purpose of this paper is to demonstrate that in the case of constant-degree polynomials, we can push the complexity of root finding down to uniform TC^0 (i.e., constant time on polynomially many processors on a TRAM, in terms of parallel complexity), as in the case of elementary arithmetical operations. (This is clearly optimal: already locating the unique root of a linear polynomial amounts to division, which is TC^0-hard.) As a corollary, the binary expansion of any algebraic constant can be computed in uniform TC^0 when given the bit position in unary. Our primary interest is theoretical, we seek to investigate the power of the complexity class TC^0; we do not expect our algorithm to be competitive with established methods in practice, and we did not make any effort to optimize parameters of the algorithm.

The basic idea of the algorithm is to express the inverse function of the polynomial by a power series, whose partial sums can be computed in TC^0 using the results of Hesse, Allender, and Barrington [90]. We need to ensure that coefficients of the series are TC^0-computable, we need bounds on the radius of convergence and convergence rate of the series, and we need to find a point in whose image to put the centre of the series so that the disk of convergence includes the origin. Doing the latter directly is in fact not much easier than approximating the root in the first place, so we instead construct a suitable polynomial-size set of sample points, and we invert the polynomial at each one of them in parallel.

We formulated our main result in terms of computational complexity, but our original motivation comes from logic (proof complexity). The bounded arithmetical theory VTC^0 (see Cook and Nguyen [68]), whose provably total computable functions are the TC^0 functions, can define addition, multiplication, and ordering on binary integers, and it proves that these operations obey the basic identities making it a discretely ordered ring. The question is which other properties of the basic arithmetical operations are provable in the theory, and in particular, whether it can prove induction (on binary integers) for some class of formulas. Now, it follows easily from known algebraic characterizations of induction for open formulas in the language of ordered rings (*IOpen*, see Shepherdson [164]) and from the witnessing theorem for VTC^0 that VTC^0 proves *IOpen* if and only if for each d there is a TC^0 root-finding algorithm for degree d polynomials whose soundness is provable in VTC^0. Our result thus establishes the computational prerequisites for proving open induction in VTC^0, leaving aside the problem of formalizing the algorithm in the theory. Since the soundness of the algorithm can be expressed as a universal sentence, we can also reformulate this result as follows: the theory $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$ proves *IOpen*.

The paper is organized as follows. In Section 2 we provide some background in the relevant parts of complexity theory and complex analysis. Section 3 contains material on inverting polynomials with power series. Section 4 presents our main result, a TC^0 root-finding algorithm. Finally, in Section 5 we discuss the connection to bounded arithmetic.

2 Preliminaries

A language L is in *nonuniform* TC^0 if there is a sequence of circuits $C_n\colon \{0,1\}^n \to \{0,1\}$ consisting of unbounded fan-in majority and negation gates such that C_n computes the characteristic function of L on strings of length n, and C_n has size at most n^c and depth c for some constant c.

L is in (*uniform*) TC^0, if the sequence $\{C_n : n \in \omega\}$ is additionally DLOGTIME-uniform (U_D-uniform in the terminology of Ruzzo [162]): i.e., we can enumerate the gates in the circuit by numbers $i < n^{O(1)}$ in such a way that one can check the type of gate i and whether gate i is an input of gate j by a deterministic Turing machine in time $O(\log n)$, given n, i, j in binary. There are other equivalent characterizations of TC^0. For one, it coincides with languages recognizable by a threshold Turing machine [141] in time $O(\log n)$ with $O(1)$ thresholds [6]. Another important characterization is in terms of descriptive complexity. We can represent a string $x \in \{0,1\}^n$ by the first-order structure $\langle\{0,\dots,n-1\},<,\mathrm{bit},X\rangle$, where X is a unary predicate encoding the bits of x. Then a language is in TC^0 iff its corresponding class of structures is definable by a sentence of FOM (first-order logic with majority quantifiers). We refer the reader to [19] for more background on uniformity of TC^0.

In some cases it may be more convenient to consider languages in a non-binary alphabet Σ. The definition of TC^0 can be adapted by adjusting the input alphabet of a threshold Turing machine, or by considering more predicates in the descriptive complexity setting. In the original definition using threshold circuits, the same can be accomplished by encoding each symbol of Σ with a binary substring of fixed length. We can also define TC^0 predicates with more than one input in the obvious way.

A function $f\colon \{0,1\}^* \to \{0,1\}^*$ is computable in TC^0 if the length of its output is polynomially bounded in the length of its input, and its bitgraph is a TC^0 predicate. (The bitgraph of f is a binary predicate $b(x,i)$ which holds iff the ith bit of $f(x)$ is 1.) In terms of the original definition, this amounts to allowing circuits $C_n\colon \{0,1\}^n \to \{0,1\}^{m(n)}$, where $m(n) = n^{O(1)}$. TC^0 functions are closed under composition, and under "parallel execution": if f is a TC^0 function, its aggregate function $g(\langle x_0,\dots,x_{m-1}\rangle) = \langle f(x_0),\dots,f(x_{m-1})\rangle$ is also TC^0. We note in this regard that TC^0 functions can do basic processing of lists

$$x_0, x_1, \dots, x_{m-1}$$

where "," is a separator character. Using the fact that TC^0 can count commas (and other symbols), we can for instance extract the ith element from the list, convert the list to and from a representation where each element is padded to some fixed length with blanks, or sort the list according to a given TC^0 comparison predicate.

We will refrain from presenting TC^0 functions in one of the formalisms suggested by the definitions above: we will give informal algorithms, generally consisting of a constant number of simple steps or TC^0 building blocks, sometimes forking into polynomially many parallel threads. The reader should have no difficulty convincing herself that our algorithms are indeed in TC^0.

We will work with numbers of various kinds. Integers will be represented in binary as usual, unless stated otherwise. As we already mentioned in the introduction, elementary arithmetical operations on integers are TC^0 functions: this includes addition, subtraction, ordering, multiplication, division with remainder, exponentiation (with unary exponents), iterated addition, iterated multiplication, and square root approximation. Here, iterated addition is the function $\langle x_0,\dots,x_{m-1}\rangle \mapsto \sum_{i<m} x_i$, and similarly for multiplication. Notice that using iterated multiplication, we can also compute factorials and binomial or multinomial coefficients of unary arguments. Base conversion is also in TC^0.

Rational numbers will be represented as pairs of integers, indicating fractions. We cannot assume fractions to be reduced, since integer gcd is not known to be TC^0-computable. Using integer division, we can convert a fraction to its binary expansion with a given accuracy (the opposite conversion is trivial). Rational arithmetic is reducible to integer arithmetic in the obvious way, hence rational addition, subtraction, ordering, multiplication, division, exponentiation (with unary integer exponents), iterated addition, iterated multiplication, and square root approximation are in TC^0.

In lieu of complex numbers, we will compute with Gaussian rationals (elements of the field $\mathbb{Q}(i)$), represented as pairs of rationals $a+ib$. By reduction to rational arithmetic, we can see that addition, subtraction, complex conjugation, norm square, norm approximation, multiplication, division, and iterated addition of Gaussian rationals are in TC^0. Using the binomial theorem, exponentiation with unary integer exponents is also in TC^0. (In fact, iterated multiplication of Gaussian rationals is in TC^0 using conversion to polar coordinates, but we will not need this.)

We will need some tools from complex analysis. We refer the reader to Ahlfors [2] or Conway [59] for background, however, we review here some basic facts to fix the notation. A function $f\colon U \to \mathbb{C}$, where $U \subseteq \mathbb{C}$ is open, is *holomorphic* (or *analytic*) in U if $f'(a) = \lim_{z\to a}(f(z)-f(a))/(z-a)$ exists for every $a \in U$. The set of all functions holomorphic in U is denoted $H(U)$. Let $B(a,r) := \{z : |z-a| < r\}$ and $\overline{B}(a,r) := \{z : |z-a| \le r\}$. If f is holomorphic in the open disk $B(a,R)$, it can be expressed by a *power series*

$$f(z)=\sum_{n=0}^{\infty}c_n(z-a)^n$$

on $B(a,R)$. More generally, if f is holomorphic in the annulus $A=B(a,R)\smallsetminus\overline{B}(a,r)$, $0\le r<R\le\infty$, it can be written in A as a *Laurent series*

$$f(z)=\sum_{n=-\infty}^{+\infty}c_n(z-a)^n.$$

We denote the coefficients of the series by $[(z-a)^n]f:=c_n$. (Other variables may be used instead of z when convenient.) The *residue* of f at a is $\operatorname{Res}(f,a):=[(z-a)^{-1}]f$. When $a=0$, we write just $[z^n]f$ and $\operatorname{Res}(f)$, respectively. The coefficients of a Laurent series are given by *Cauchy's integral formula*:

$$[(z-a)^n]f=\frac{1}{2\pi i}\int_\gamma\frac{f(z)}{(z-a)^{n+1}}\,dz,$$

where γ is any closed curve in A whose index with respect to a is 1 (such as the circle $\gamma(t)=a+\varrho e^{2\pi it}$, $r<\varrho<R$). The *identity theorem* states that if f,g are holomorphic in a region (i.e., connected open set) U and coincide on a set $X\subseteq U$ which has a limit point in U, then $f=g$. The *open mapping theorem* states that a nonconstant function f holomorphic in a region is an open mapping (i.e., maps open sets to open sets).

If $X\subseteq\mathbb{C}$ and $a\in\mathbb{C}$, we put $\operatorname{dist}(a,X)=\inf\{|z-a|:z\in X\}$.

We will also need some easy facts on zeros of polynomials. Let $f\in\mathbb{C}[x]$ be a degree d polynomial, and write $f(x)=\sum_{j=0}^d a_jx^j$. *Cauchy's bound* [180, L. 6.2.7] states that every zero α of f satisfies

$$\frac{|a_0|}{|a_0|+\max\limits_{0<j\le d}|a_j|}\le|\alpha|\le 1+\max_{j<d}\frac{|a_j|}{|a_d|}.$$

Let $f,g\in(\mathbb{Q}(i))[x]$ be two polynomials of degrees d,e (resp.), and assume $f(\alpha)=g(\beta)=0$, $\alpha\ne\beta$. If $f,g\in(\mathbb{Z}[i])[x]$, we have

$$|\alpha-\beta|\ge\frac{1}{(2^{d+1}\|f\|_\infty)^e\|g\|_2^d},\tag{1}$$

where $\|f\|_p$ denotes the L_p-norm of the vector of coefficients of f [180, §6.8]. In general, we can apply (1) to the polynomials rf and sg, where r is the product of all denominators appearing among the coefficients of f, and similarly for s. If we represent f and g by the lists of their coefficients, which are in turn represented by quadruples of binary integers as detailed above, we obtain easily the following root separation bound:

Lemma 2.1 *For each $j=0,1$, let $f_j\in(\mathbb{Q}(i))[x]$ have degree d_j and total bit size n_j, and assume $f_j(\alpha_j)=0$. If $\alpha_0\ne\alpha_1$, then*

$$|\alpha_0-\alpha_1|\ge 2^{-(d_1n_0+d_0n_1)}\ge 2^{-n_0n_1}.$$

□

3 Inverting polynomials

As already mentioned in the introduction, the main strategy of our algorithm will be to approximate a power series computing the inverse function of the given polynomial f. In this section, we establish the properties of such series needed to make the algorithm work.

The basic fact we rely on is that holomorphic functions with nonvanishing derivative are locally invertible: i.e., if $f \in H(U)$ and $a \in U$ is such that $f'(a) \neq 0$, there exist open neighbourhoods $a \in U_0 \subseteq U$ and $f(a) \in V_0$ such that f is a homeomorphism of U_0 onto V_0, and the inverse function $g = (f \restriction U_0)^{-1}$ is holomorphic in V_0. In particular, g is computable by a power series in a neighbourhood of $f(a)$.

Notice that local inverses of holomorphic functions are automatically two-sided: if $f \in H(U)$, $g \in H(V)$, $a \in U$, $b \in V$, $g(b) = a$, and $f(g(z)) = z$ in a neighbourhood of b, then $g(f(z)) = z$ in a neighbourhood of a.

The coefficients of the power series of an inverse of a holomorphic function are given by the *Lagrange inversion formula* [58, §3.8, Thm. A]:

Fact 3.1 *Let $f \in H(U)$, $g \in H(V)$, $f \circ g = \mathrm{id}_V$, $a = g(b) \in U$, $b = f(a) \in V$, $n > 0$. Then*

$$[(w-b)^n]g(w) = \frac{1}{n}\operatorname{Res}\left(\frac{1}{(f(z)-b)^n}, a\right). \qquad \square$$

We can make the formula even more explicit as follows. First, the composition of two power series is given by *Faà di Bruno's formula* [58, §3.4, Thm. A], which we formulate only for $a = b = 0$ for simplicity:

Fact 3.2 *Let $f \in H(U)$, $g \in H(V)$, $g(0) = 0 \in U$, $f(0) = 0 \in V$, $n \geq 0$. Then*

$$[z^n](g \circ f) = \sum_{\sum_{j=1}^\infty j m_j = n} \binom{\sum_j m_j}{m_1, m_2, \dots} [w^{\sum_j m_j}]g \prod_{j=1}^\infty \left([z^j]f\right)^{m_j}. \qquad \square$$

Note that here and below, the outer sum is finite, and the product has only finitely many terms different from 1, hence the right-hand side is well-defined without extra assumptions on convergence. We can now expand the residue in Fact 3.1 to obtain the following version of Lagrange inversion formula, which only refers to the coefficients of f [58, §3.8, Thm. E]:

Proposition 3.3 *Let $f \in H(U)$, $g \in H(V)$, $f \circ g = \mathrm{id}_V$, $a = g(b) \in U$, $b = f(a) \in V$. Then $[(w-b)^0]g = a$, and for $n > 0$,*

$$[(w-b)^n]g = \frac{1}{n!\,[z-a]f} \sum_{\sum_{j=2}^\infty (j-1)m_j = n-1} \left(\textstyle\sum_j j m_j\right)! \prod_{j=2}^\infty \frac{1}{m_j!}\left(-\frac{[(z-a)^j]f}{([z-a]f)^j}\right)^{m_j}.$$

Proof: Note that $f'(a) \neq 0$. Put $f_1(z) = f(a + z/f'(a)) - b$ and $g_1(w) = f'(a)(g(b+w) - a)$, so that $f_1(0) = 0 = g_1(0)$, $f_1 \circ g_1 = \mathrm{id}$ on a neighbourhood of 0, and $f_1'(0) = 1$. Write

$f_1(z) = z(1-h(z))$, where h is holomorphic in a neighbourhood of 0, and $h(0) = 0$. Then

$$\begin{aligned}[w^n]g_1 = \frac{1}{n}[z^{-1}]\frac{1}{f_1^n} &= \frac{1}{n}[z^{n-1}]\frac{1}{(1-h)^n}\\ &= \frac{1}{n}\sum_{\sum_{j=1}^\infty jm_j=n-1}\frac{(\sum_j m_j)!}{m_1!\,m_2!\cdots}\frac{(\sum_j m_j+n-1)!}{(\sum_j m_j)!\,(n-1)!}\prod_{j=1}^\infty \big([z^j]h\big)^{m_j}\\ &= \frac{1}{n!}\sum_{\sum_{j=1}^\infty jm_j=n-1}\frac{(\sum_j (j+1)m_j)!}{m_1!\,m_2!\cdots}\prod_{j=1}^\infty\big(-[z^{j+1}]f_1\big)^{m_j}\\ &= \frac{1}{n!}\sum_{\sum_{k=2}^\infty (k-1)m_k=n-1}(\textstyle\sum_k km_k)!\displaystyle\prod_{k=2}^\infty\frac{\big(-[z^k]f_1\big)^{m_k}}{m_k!}\end{aligned}$$

using Facts 3.1 and 3.2, and the expansion $[w^r](1-w)^{-n} = \binom{r+n-1}{n-1}$. The result follows by noting that for any $k, n > 0$, $[z^k]f_1 = ([(z-a)^k]f)/(f'(a))^k$, $[w^n]g_1 = f'(a)[(w-b)^n]g$, and $f'(a) = [z-a]f$. □

Let d be a constant. If f in Proposition 3.3 is a polynomial of degree d, then the product is nonzero only when $m_j = 0$ for every $j > d$, hence it suffices to enumerate $m_2, \dots, m_d$. It follows easily that the outer sum has polynomially many (namely, $O(n^d)$) terms, and we can compute $[(w-b)^n]g$ in uniform TC^0 given a, b, and the coefficients of f in binary, and n in unary.

Apart from a description of the coefficients, we also need bounds on the radius of convergence of the inverse series, and on its rate of convergence (i.e., on the norm of its coefficients). Generally speaking, the radius of convergence of a power series is the distance to the nearest singularity. Since a polynomial f is an entire proper map, its inverse cannot escape to infinity or hit a point where f is undefined, thus the only singularities that can happen are branch points. These occur at zeros of f'. This suggests that the main parameter governing the radius of convergence and other properties of the inverse should be the distance of a to the set $C_f = \{z \in \mathbb{C} : f'(z) = 0\}$ of *critical points* of f.

Lemma 3.4 *Let $f \in \mathbb{C}[x]$ be a degree d polynomial with no roots in $B(a,R)$, $R > 0$, and let $\mu > 0$. Then $|f(z) - f(a)| < \big((1+\mu)^d - 1\big)|f(a)|$ for all $z \in B(a, \mu R)$.*

Proof: Write $f(z) = c\prod_{j=1}^d (z-\alpha_j)$. We have

$$\frac{f(z)}{f(a)} = \prod_{j=1}^d \frac{(z-a)+(a-\alpha_j)}{a-\alpha_j} = \sum_{I\subseteq\{1,\dots,d\}}\prod_{j\in I}\frac{z-a}{a-\alpha_j},$$

hence

$$\left|\frac{f(z)}{f(a)} - 1\right| = \Big|\sum_{I\neq\varnothing}\prod_{j\in I}\frac{z-a}{a-\alpha_j}\Big| \le \sum_{I\neq\varnothing}\prod_{j\in I}\frac{|z-a|}{R} < (1+\mu)^d - 1.$$ □

Proposition 3.5 *Let $f \in \mathbb{C}[x]$ have degree $d > 1$, $f(a) = b$, and $0 < R \le \operatorname{dist}(a, C_f)$. Let*

$$g(w) = a + \sum_{n=1}^\infty c_n (w-b)^n$$

satisfy $f \circ g = \mathrm{id}_{B(b,\varrho)}$, *where* $\varrho > 0$ *is the radius of convergence of* g. *Put*

$$\mu = \sqrt[d-1]{2} - 1 \geq \frac{\ln 2}{d-1}, \qquad \nu = \frac{2(d-1)\mu - 1}{d} \geq \frac{\ln 4 - 1}{d},$$
$$\lambda_\delta = \sqrt[d]{1 + \delta d\nu} - 1 \geq \frac{\delta \ln\ln 4}{d}, \qquad \varrho_0 = \nu R\,|f'(a)|$$

for $0 < \delta \leq 1$ (*the inequalities are established below*). *Then:*

(i) f *is injective on* $\overline{B}(a, \mu R)$.

(ii) $\varrho \geq \varrho_0$.

(iii) $g[B(b,\varrho)] \supseteq B(a, \lambda_1 R)$, *and more generally,* $g[B(b, \delta\varrho_0)] \supseteq B(a, \lambda_\delta R)$ *for each* $\delta \in (0,1]$.

(iv) $|c_n| \leq \mu R / n\varrho_0^n$.

Proof: Notice that $e^x - 1 \geq x$ for every $x \in \mathbb{R}$, hence $\sqrt[d-1]{2} - 1 = \exp((\ln 2)/(d-1)) - 1 \geq (\ln 2)/(d-1)$; $\nu \geq (\ln 4 - 1)/d$ immediately follows. Similarly, $\lambda_\delta \geq \ln(1 + \delta(\ln 4 - 1))/d$. We have $\ln(1 + \delta(\ln 4 - 1)) \geq \delta \ln\ln 4$ for $\delta \in [0,1]$ as ln is concave.

(i): Let $u, v \in \overline{B}(a, \mu R)$, $u \neq v$. We have

$$f(v) - f(u) = \int_u^v f'(z)\,dz = (v-u)\left(f'(a) + \int_0^1 f'\big((1-t)u + tv\big) - f'(a)\,dt\right).$$

Since $|f'((1-t)u + tv) - f'(a)| < |f'(a)|$ for all $t \in (0,1)$ by Lemma 3.4, we obtain

$$\left|\int_0^1 f'\big((1-t)u + tv\big) - f'(a)\,dt\right| \leq \int_0^1 \left|f'\big((1-t)u + tv\big) - f'(a)\right|\,dt < |f'(a)|,$$

thus $f(u) \neq f(v)$.

(ii): Let $U = B(a, \mu R)$. Since f is a biholomorphism of U and $f[U]$, $\varrho \geq \mathrm{dist}(b, \mathbb{C} \smallsetminus f[U])$. Since $f[U]$ is open, there exists $w \notin f[U]$ such that $|w - b| = \mathrm{dist}(b, \mathbb{C} \smallsetminus f[U])$. Let $z_n \in U$ be such that $\lim_n f(z_n) = w$. By compactness, $\{z_n\}$ has a convergent subsequence; without loss of generality, there exists $z = \lim_n z_n$. Then $f(z) = w$ by continuity, hence $z \notin U$. However, $z \in \overline{U}$, hence z is in the topological boundary $\partial U = \overline{U} \smallsetminus \mathrm{int}\,U = \overline{U} \smallsetminus U$. We have thus verified that $\varrho \geq \mathrm{dist}(b, f[\partial U])$.

Let $u = a + \mu R e^{i\theta} \in \partial U$. We have

$$f(u) = b + \int_a^u f'(z)\,dz = b + Re^{i\theta}\left(\mu f'(a) + \int_0^\mu f'(a + te^{i\theta}R) - f'(a)\,dt\right).$$

By Lemma 3.4, $|f'(a + te^{i\theta}R) - f'(a)| \leq \big((1+t)^{d-1} - 1\big)|f'(a)|$, hence

$$\left|\int_0^\mu f'(a + te^{i\theta}R) - f'(a)\,dt\right| \leq |f'(a)| \int_0^\mu (1+t)^{d-1} - 1\,dt = |f'(a)|\left(\frac{(1+\mu)^d - 1}{d} - \mu\right)$$
$$= |f'(a)|\,\frac{2(1+\mu) - 1 - d\mu}{d} = |f'(a)|\,\frac{1 - (d-2)\mu}{d}.$$

Thus,

$$|f(u)-b| \ge R\,|f'(a)|\left(\mu - \frac{1-(d-2)\mu}{d}\right) = \nu R\,|f'(a)|.$$

(iii): The proof above shows that $g[B(b,\varrho_0)] \subseteq U$. As f is injective on $U \supseteq B(a,\lambda_\delta R)$, it suffices to show that $f[B(a,\lambda_\delta R)] \subseteq B(b,\delta\varrho_0)$. Let thus $u = a + \lambda R e^{i\theta}$, $\lambda < \lambda_\delta$. As above,

$$f(u) = b + Re^{i\theta}\left(\lambda f'(a) + \int_0^\lambda f'(a+te^{i\theta}R) - f'(a)\,dt\right)$$

and

$$\left|\int_0^\lambda f'(a+te^{i\theta}R) - f'(a)\,dt\right| \le |f'(a)|\left(\frac{(1+\lambda)^d-1}{d} - \lambda\right),$$

hence

$$|f(u)-b| \le R\,|f'(a)|\frac{(1+\lambda)^d-1}{d} < R\,|f'(a)|\frac{(1+\lambda_\delta)^d-1}{d} = \delta\varrho_0.$$

(iv): Let $\gamma(t) = a + \mu R e^{2\pi i t}$. By Fact 3.1 and Cauchy's integral formula,

$$c_n = \frac{1}{2\pi i n}\int_\gamma \frac{dz}{(f(z)-b)^n} = \frac{\mu R}{n}\int_0^1 \frac{e^{2\pi i t}\,dt}{(f(\gamma(t))-b)^n}.$$

The proof of (ii) shows $|f(\gamma(t))-b| \ge \varrho_0$, hence

$$|c_n| \le \frac{\mu R}{n}\int_0^1 \frac{dt}{|f(\gamma(t))-b|^n} \le \frac{\mu R}{n\varrho_0^n}. \qquad \square$$

Example 3.6 Let $f(z) = z^d$, $a = b = 1$. Then $f' = dz^{d-1}$, $C_f = \{0\}$, $R = 1$, $f'(a) = d$. It is not hard to see that f is injective on $B(1,r)$ iff no two points of $B(1,r)$ have arguments differing by $2\pi/d$ iff $r \le \sin(\pi/d) = \pi/d + O(d^{-3})$. Since g must hit a root of f' at the circle of convergence, we must have $\varrho = 1 = (1/d)Rf'(a)$. Finally, $|(1+z)^d - 1|$ is maximized on $\{z : |z| = r\}$ for z positive real, thus $B(1,\lambda R) \subseteq g[B(1,\delta\varrho)]$ iff $(1+\lambda)^d - 1 \le \delta$ iff $\lambda \le (1+\delta)^{1/d} - 1 = \ln(1+\delta)/d + O(d^{-2})$. Thus, in Proposition 3.5, μ, ν, and λ_δ are optimal up to a linear factor.

Remark 3.7 We prefer to give a simple direct proof of Proposition 3.5 for the benefit of the reader. Nevertheless, we could have assembled the bounds (with somewhat different constants) from several more sophisticated results in the literature. The Grace–Heawood theorem (or rather its corollary, originally due to Alexander, Kakeya, and Szegő; see [128, Thm. 23,2]) states that (i) holds with $\mu = \sin(\pi/d)$ (which is tight in view of the z^d example). Then the Koebe 1/4-theorem [60, Thm. 14.7.8] implies (ii) with $\nu = \mu/4$, and one more application of the theorem yields (iii) with $\lambda_\delta = \nu\delta/4$.

4 Root finding in TC^0

We start with the core part of our root-finding algorithm. While it is conceptually simple, its output is rather crude, so we will have to combine it with some pre- and postprocessing to obtain the desired result (Theorem 4.5).

Theorem 4.1 *Let d be a constant. There exists a uniform TC^0 function which, given the coefficients of a degree d polynomial $f \in (\mathbb{Q}(i))[x]$ in binary and t in unary, computes a list $\{z_j : j < s\} \subseteq \mathbb{Q}(i)$ such that every complex root of f is within distance 2^{-t} of some z_j.*

Proof: If $d = 1$, it suffices to divide the coefficients of f. Assume $d \geq 2$. Let $\mu, \nu, \lambda = \lambda_{1/2}$ be as in Proposition 3.5 (more precisely, we should use their fixed rational approximations; we will ignore this for simplicity). Let $A = 1 + \lambda/5$, $p = \lceil 5\pi/\lambda \rceil$, and $\xi = e^{2\pi i/p}$ (approximately, again). Consider the TC^0 algorithm given by the following description:

(i) Input: $f = \sum_{j \leq d} f_j z^j$ with $f_j \in \mathbb{Q}(i)$, $f_d \neq 0$, and $t > 0$ in unary.

(ii) Put $\varepsilon = 2^{-t}$. Compute recursively a list $C = \{\alpha_j : j < s\}$ including $\varepsilon/4$-approximations of all roots of f'.

(iii) Output (in parallel) each α_j.

(iv) Put $c = 2 + \max_{j<d} |f_j/f_d|$ and $k_{\max} = \lceil \log(2c\varepsilon^{-1})/\log A \rceil$.

(v) For every $j < s$, $k < k_{\max}$, and $q < p$, do the following in parallel.

(vi) Let $a = \alpha_j + \varepsilon A^k \xi^q$, $b = f(a)$, $R = \frac{1}{2}|a - \alpha_j|$, $N = \lceil \log_2(\mu R \varepsilon^{-1}) \rceil$.

(vii) For each $h \leq d$, let $\tilde{f}_h = \sum_{u=h}^{d} \binom{u}{h} f_u a^{u-h}$.

(viii) Compute and output

$$z_{j,k,q} = a + \sum_{\substack{m_2,\dots,m_d \\ \sum_h (h-1)m_h < N}} \frac{(2m_2 + \cdots + dm_d)!\,(-\tilde{f}_2)^{m_2} \cdots (-\tilde{f}_d)^{m_d} (-b)^{1+m_2+\cdots+(d-1)m_d}}{m_2! \cdots m_d!\,(1 + m_2 + \cdots + (d-1)m_d)!\,\tilde{f}_1^{1+2m_2+\cdots+dm_d}}.$$

Let $f(\alpha) = 0$, we have to show that one of the numbers output by the algorithm is ε-close to α. If $|\alpha - \alpha_j| < \varepsilon$ for some j, we are done by step (iii). We can thus assume $\mathrm{dist}(\alpha, C) \geq \varepsilon$, which implies $\mathrm{dist}(\alpha, C_f) \geq 3\varepsilon/4$. Assume that α_j is an $\varepsilon/4$-approximation of the root $\tilde{\alpha}_j$ of f' nearest to α. Since all roots of f or f' have modulus bounded by $c - 1$ by Cauchy's bound, we have $\varepsilon \leq |\alpha - \alpha_j| < 2c$, thus there exists $k < k_{\max}$ such that $\varepsilon A^k \leq |\alpha - \alpha_j| < \varepsilon A^{k+1}$. Let $q < p$ be such that the argument of $\alpha - \alpha_j$ differs from $2\pi q/p$ by at most π/p, and consider steps (v)–(viii) for this particular choice of j, k, q (cf. Fig. 4.1). We have

$$|\alpha - a| \leq \left(\frac{\pi}{p} + 1 - \frac{1}{A}\right) |\alpha - \alpha_j| \leq \frac{2\lambda}{5} |\alpha - \alpha_j| < \frac{1}{5} |\alpha - \alpha_j|.$$

Notice that

$$\mathrm{dist}(\alpha, C_f) = |\alpha - \tilde{\alpha}_j| \geq |\alpha - \alpha_j| - \frac{\varepsilon}{4}$$

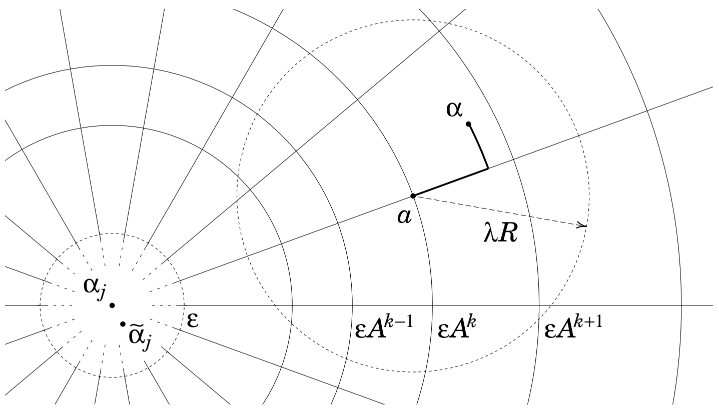

Figure 4.1: The spiderweb.

by the choice of $\tilde{\alpha}_j$ and α_j, hence

$$\operatorname{dist}(a, C_f) \geq \operatorname{dist}(\alpha, C_f) - |a - \alpha| \geq |\alpha - \alpha_j| - \frac{\varepsilon}{4} - \frac{1}{5}\,|\alpha - \alpha_j| \geq \frac{1}{2}\,|\alpha - \alpha_j| \geq R.$$

Since

$$|a - \alpha_j| \geq |\alpha - \alpha_j| - |a - \alpha| > \frac{4}{5}\,|\alpha - \alpha_j|,$$

we also have

$$|a - \alpha| < \frac{5}{4}\,\frac{2\lambda}{5}\,|a - \alpha_j| = \lambda R.$$

Let

$$g(w) = a + \sum_{n=1}^{\infty} c_n (w - b)^n$$

be an inverse of f in a neighbourhood of b, and let ϱ be its radius of convergence. By Proposition 3.5, $|-b| = |f(\alpha) - b| < \varrho_0/2$, where $\varrho_0 = \nu R\,|f'(a)| \leq \varrho$. Thus, $g(f(\alpha)) = g(0) = \alpha$. Since $\sum_h \tilde{f}_h z^h = f(z + a)$ by the binomial formula, $\tilde{f}_h = [(z - a)^h]f$. Then it follows from Proposition 3.3 that

$$z_{j,k,q} = a + \sum_{n=1}^{N} c_n (-b)^n.$$

Since

$$|c_n(-b)^n| \leq \frac{\mu R}{n \varrho_0^n} |b|^n < \frac{\mu R}{2^n}$$

by Proposition 3.5, we have

$$|\alpha - z_{j,k,q}| = \left| \sum_{n=N+1}^{\infty} c_n (-b)^n \right| < \frac{\mu R}{2^N} \leq \varepsilon. \qquad \square$$

Most of the algorithm described in Theorem 4.1 is independent of the assumption of d being constant (or it can be worked around). There are two principal exceptions. First, the recursion in step (ii) amounts to d sequential invocations of the algorithm. Second, while N is still linear in the size of the input, the main sum in step (viii) has roughly N^d terms. Thus, approximation of roots of arbitrary univariate polynomials can be done by (uniform) threshold circuits of depth $O(d)$ and size $n^{O(d)}$, where n is the total length of the input. (The known NC algorithms for root finding can do much better for large d.)

The algorithm from Theorem 4.1 does the hard work in locating the roots of f, but it suffers from several drawbacks:

- Its output includes a lot of bogus results that are not actually close to any root of f.
- There may be many elements on the list close to the same root, and we do not get any information on the multiplicity of the roots.
- The roots have no "identity": if we run the algorithm for two different ts, we do not know which approximate roots on the output lists correspond to each other.
- It may be desirable to output the binary expansions of the roots rather than just approximations.

We are going to polish the output of the algorithm to fix these problems. Let us first formulate precisely the goal.

Definition 4.2 The *t-digit binary expansion* of $a \in \mathbb{C}$ is the pair $\langle \lfloor \mathrm{Re}(a2^t) \rfloor, \lfloor \mathrm{Im}(a2^t) \rfloor \rangle$, where both integers are written in binary. A *root-finding algorithm* for a set of polynomials $P \subseteq (\mathbb{Q}(i))[x]$ is an algorithm with the following properties:

(i) The input consists of a polynomial $f \in P$ given by a list of its coefficients in binary, and a positive integer t in unary.

(ii) The output is a list of pairs $\{\langle z_j(f,t), e_j(f,t)\rangle : j < s(f,t)\}$.

(iii) For every $f \in P$, there exists a factorization

$$f(z) = c \prod_{j<s} (z - a_j)^{e_j},$$

where $c \in \mathbb{Q}(i)$, $a_j \in \mathbb{C}$, $a_j \neq a_k$ for $j \neq k$, and $e_j > 0$, such that for every t: $s(f,t) = s$, $e_j(f,t) = e_j$, and $z_j(f,t)$ is the t-digit binary expansion of a_j.

We note that the choice of base 2 in the output is arbitrary, the algorithm can output expansions in any other base if needed.

Lemma 4.3 *Let d be a constant. Given a degree d polynomial $f \in (\mathbb{Q}(i))[x]$, we can compute in uniform* TC^0 *a list of pairwise coprime square-free nonconstant polynomials f_j, $c \in \mathbb{Q}(i)$, and integers $e_j > 0$ such that $f = c \prod_{j<k} f_j^{e_j}$, where $k, e_j \leq d$.*

Proof: Since d is constant, division of degree d polynomials takes $O(1)$ arithmetical operations, hence it can be implemented in uniform TC^0. The same holds for gcd, using the Euclidean algorithm. We compute a list $L = \langle f_j : j < k\rangle$, $k \le d$, of nonconstant polynomials such that $f = \prod_j f_j$ as follows:

(i) Start with $L = \langle f\rangle$. Repeat the following steps until none of them is applicable.

(ii) If f_j is not square-free, replace it with $\gcd(f_j, f_j')$ and $f_j/\gcd(f_j, f_j')$.

(iii) If $f_h \mid f_j$, $f_j \nmid f_h$ for some h, j, replace f_j in L with $f_h, f_j/f_h$.

(iv) If $g := \gcd(f_h, f_j) \ne 1$ for some h, j such that $f_h \nmid f_j$, $f_j \nmid f_h$, replace f_h, f_j in L with $g, g, f_h/g, f_j/g$.

The algorithm terminates after at most d steps, hence it is in TC^0. Clearly, it computes a list of square-free polynomials such that for every h, j, f_h is coprime to f_j or f_h is a scalar multiple of f_j. It remains to collect scalar multiples of the same polynomial together. □

Lemma 4.4 *Let d be a constant. Given a degree d square-free polynomial $f \in (\mathbb{Q}(i))[x]$ and t in unary, we can compute in uniform* TC^0 *a list $\{z_j : j < s\}$ such that every root of f is within distance 2^{-t} of some z_j, and every z_j is within distance 2^{-t} of some root.*

Proof: We use the notation from the proof of Theorem 4.1. We modify the algorithm from that proof as follows:

- We compute an $\varepsilon_0 > 0$ such that the distance of any root of f to any root of f' is at least ε_0 using Lemma 2.1. In step (ii), we put $\varepsilon = \min\{2^{-t}, \varepsilon_0/3\}$.
- We skip step (iii).
- In step (vi), we check that $|b| < \frac{1}{2}\nu|f'(a)|R$ and $|a - \alpha_{j'}| \ge R + \varepsilon/4$ for every $j' < s$. If either condition is violated, we output a symbol "$*$" instead of a number, and skip the remaining two steps.

The result is a list of numbers and $*$'s; it is easy to construct the sublist consisting of only numbers by a TC^0 function.

Let $z_{j,k,q}$ be one of the numbers output by the algorithm. In step (vi) we ensured $\operatorname{dist}(a, C) \ge R + \varepsilon/4$, hence $\operatorname{dist}(a, C_f) \ge R$. Moreover, $|0 - b| < \varrho_0/2$, hence 0 is within the radius of convergence of g, and $\alpha = g(0)$ is a root of f whose distance from $z_{j,k,q}$ is

$$|\alpha - z_{j,k,q}| = \left|\sum_{n=N+1}^{\infty} c_n(-b)^n\right| < \frac{\mu R}{2^N} \le \varepsilon.$$

On the other hand, let α be a root of f. Since $\operatorname{dist}(\alpha, C_f) \ge \varepsilon_0$, we have $\operatorname{dist}(\alpha, C) \ge \varepsilon$, hence we can choose j, k, q such that $|\alpha - z_{j,k,q}| < \varepsilon$ as in the proof of Theorem 4.1. We have to show that the extra conditions in step (vi) are satisfied. $|b| < \frac{1}{2}\nu R|f'(a)|$ was verified in the proof of Theorem 4.1. Moreover,

$$|a - \alpha_{j'}| \ge |\alpha - \tilde\alpha_{j'}| - |a - \alpha| - \frac{\varepsilon}{4} \ge |a - \tilde\alpha_j| - |a - \alpha| - \frac{\varepsilon}{4} \ge \frac{4}{5}|\alpha - \alpha_j| - \frac{\varepsilon}{2} \ge R + \frac{\varepsilon}{4}$$

as $|\alpha - \alpha_j| \ge \varepsilon_0 - \varepsilon/4 > \frac{5}{2}\varepsilon$. □

We can now finish the proof of the main result of this paper:

Theorem 4.5 *For every constant d, there exists a uniform* TC^0 *root-finding algorithm for degree d polynomials in the sense of Definition 4.2.*

Proof: We employ the notation of Definition 4.2. By Lemma 4.3, we can assume f to be square-free (in which case we will have $e_j(f,t) = 1$ for all j, so we only need to compute the roots). Consider the following TC^0 algorithm:

(i) Using Lemma 2.1, compute an $\eta > 0$ such that all roots of f are at distance at least η from each other.

(ii) Using Lemma 4.4, compute a list $\{r'_j : j < u\}$ such that every root of f is within distance $\eta/5$ of some r'_j, and vice versa.

(iii) Note that if r'_h and r'_j correspond to the same root, then $|r'_h - r'_j| < \frac{2}{5}\eta$, otherwise $|r'_h - r'_j| > \frac{3}{5}\eta$. Use this criterion to omit duplicate roots from the list, creating a list $\{r_j : j < d\}$ which contains $\eta/5$-approximations of all roots of f, each of them exactly once.

(iv) If $\varepsilon := 2^{-t} \geq \eta/5$, output $z_j := r_j$ and halt. Otherwise use Lemma 4.4 to construct a list $\{z'_h : h < s\}$ consisting of ε-approximations of roots of f.

(v) For each $j < d$, output $z_j := z'_{h(j)}$, where $h(j)$ is the smallest $h < s$ such that $|z'_h - r_j| < \eta/2$.

Notice that the computation of r_j is independent of t. Let a_j be the unique root of f such that $|a_j - r_j| < \eta/5$. Given t and i, let j' be such that $|z'_h - a_{j'}| < \varepsilon$. Then $|z'_h - r_j| < \varepsilon + \eta/5 \leq \frac{2}{5}\eta$ if $j = j'$, otherwise $|z'_h - r_j| > \frac{4}{5}\eta - \varepsilon \geq \frac{3}{5}\eta$. Thus, the definition of $h(j)$ in the last step is sound, and guarantees $|z_j - a_j| < \varepsilon$.

It follows that this TC^0 function has all the required properties, except that it computes approximations instead of binary expansions. We can fix this as follows. Using the algorithm we have just described, we can compute integers u, v such that $|u + iv - 2^t a_j| < 1$. Then $\lfloor \mathrm{Re}(2^t a_j) \rfloor$ is either u or $u - 1$, hence it remains to find the sign of $\mathrm{Re}(2^t a_j) - u$ (the case of Im is similar).

Let $g(z) = f(2^{-t}(2z + u))$, $h(z) = \overline{g}(-z)$, and $\alpha = \frac{1}{2}(2^t a_j - u)$. Then $g(\alpha) = 0 = h(-\overline{\alpha})$ and $\alpha - (-\overline{\alpha}) = \mathrm{Re}(2^t a_j) - u$. Using Lemma 2.1, we can compute $\xi > 0$ such that $|\alpha - (-\overline{\alpha})| \geq \xi$ whenever it is nonzero. Using the algorithm above, we can compute rational u', v' such that $|u' + iv' - 2^t a_j| < \xi/4$. If $|u - u'| < \xi/2$, then $\mathrm{Re}(2^t a_j) = u$. Otherwise, $|\mathrm{Re}(2^t a_j) - u| \geq \xi$, hence the sign of $u' - u$ agrees with the sign of $\mathrm{Re}(2^t a_j) - u$. □

Corollary 4.6 *If α is a fixed real algebraic number, then the kth bit of α can be computed in uniform* TC^0, *given k in unary.* □

(Note that this corollary is only interesting in the uniform setting, since the language is unary.)

5 Open induction in VTC^0

As we already mentioned in the introduction, our primary motivation for studying root finding for constant-degree polynomials comes from bounded arithmetic. We will now describe the connection in more detail. A reader not interested in bounded arithmetic may safely stop reading here.

The basic objects of study in bounded arithmetic are weak first-order theories based on integer arithmetic. There is a loose correspondence of arithmetical theories to complexity classes: in particular, if a theory T corresponds to a class C, then the provable total computable functions of T are functions from C (or more precisely, FC). The following is one of the natural problems to study in this context: assume we have a concept (say, a language or a function) from the computational class C. Which properties of this concept are provable in the theory T? (This asks for a form of feasible reasoning: what can we show about the concept when we are restricted to tools not exceeding its complexity?)

Here we are concerned with the theory VTC^0, corresponding to TC^0. We refer the reader to Cook and Nguyen [68] for a comprehensive treatment of VTC^0. Let us briefly recall that VTC^0 is a two-sorted theory, with one sort intended for natural numbers (which we think of as given in unary), and one sort for finite sets of these unary numbers (which we also regard as finite binary strings, or as numbers written in binary). We are primarily interested in the binary number sort, we consider the unary sort to be auxiliary. We use capital letters $X, Y, \dots$ for variables of the binary (set) sort, and lowercase letters $x, y, \dots$ for the unary sort. The language of the theory consists of basic arithmetical operations on the unary sort, the elementhood (or bit) predicate $x \in X$, and a function $|X|$ which extracts an upper bound on elements of a set X. The axioms of VTC^0 include comprehension for Σ_0^B formulas (formulas with number quantifiers bounded by a term and no set quantifiers)—which also implies induction on unary numbers for Σ_0^B formulas—and an axiom ensuring the existence of counting functions for any set. The provably total computable (i.e., Σ_1^1-definable: Σ_1^1 formulas consist of a block of existential set quantifiers in front of a Σ_0^B formula) functions of VTC^0 are the TC^0 functions.

In VTC^0, we can define the basic arithmetical operations $+, \cdot, \leq$ on binary integers. Our main question is, what properties of these operations are provable in VTC^0. (We can make this more precise as follows: which theories in the usual single-sorted language of arithmetic $L_{PA} = \langle 0, 1, +, \cdot, \leq \rangle$ are interpreted in VTC^0 by the corresponding operations on the binary sort?) It is not hard to show that VTC^0 proves binary integers to form a discretely ordered ring (DOR). What we would especially like to know is whether VTC^0 can prove the induction schema on the binary sort

$$\varphi(0) \wedge \forall X\, (\varphi(X) \to \varphi(X+1)) \to \forall X\, \varphi(X)$$

for some nontrivial class of formulas φ. In particular, we want to know whether VTC^0 includes the theory $IOpen$ (axiomatized by induction for open formulas of L_{PA} over DOR) introduced by Shepherdson [164] and widely studied in the literature.

Now, assume for a moment that $VTC^0 \vdash IOpen$. Then for each constant d, VTC^0 proves

$$X < Y \wedge F(X) \leq 0 < F(Y) \to \exists Z\, (X \leq Z < Y \wedge F(Z) \leq 0 < F(Z+1))$$

where $F(X) = \sum_{j \le d} U_j X^j$ is a degree d integer polynomial whose coefficients are parameters of the formula. This is (equivalent to) a Σ^1_1 formula, hence the existential quantifier is, provably in VTC^0, witnessed by a TC^0 function $G(U_0, \dots, U_d, X, Y)$. Since any rational polynomial is a scalar multiple of an integer polynomial, and we can pass from a polynomial $F(X)$ to $2^{td}F(2^{-t}X)$ to reduce the error from 1 to 2^{-t}, we see that there is a TC^0 algorithm solving the following root-finding problem: given a degree d rational polynomial and two rational bounds where it assumes opposite signs, approximate a real root of the polynomial between the two bounds up to a given accuracy. Using a slightly more complicated argument, one can also obtain a root-finding algorithm in the set-up we considered earlier : i.e., we approximate all complex roots of the polynomial, and the input of the algorithm is only the polynomial and the desired error of approximation. Thus, a TC^0 root-finding algorithm is a necessary prerequisite for showing *IOpen* in VTC^0.

We can in a sense reverse the argument above to obtain a proof of open induction from a root-finding algorithm, but there is an important caveat. The way we used the witnessing theorem for VTC^0, we lost the information that the soundness of the algorithm is provable in VTC^0. Indeed, if we are only concerned with the computational complexity of witnessing functions, then witnessing of Σ^1_1 formulas is unaffected by addition of true universal (i.e., $\forall\Sigma^B_0$) axioms to the theory. In other words, the same argument shows the existence of a root-finding algorithm from the weaker assumption $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N}) \vdash$ *IOpen*, where $\mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$ denotes the set of all $\forall\Sigma^B_0$ sentences true in the standard model of arithmetic. Now, this formulation of the argument can be reversed:

Theorem 5.1 *The theory $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$ proves IOpen for the binary number sort.*

Proof: Let M be a model of $VTC^0 + \mathrm{Th}_{\forall\Sigma^B_0}(\mathbb{N})$, and D be the discretely ordered ring of the binary integers of M. For any constant d, we can use Theorem 4.1 to construct a TC^0 function which, given the coefficients of an integer polynomial of degree d, computes a list of integers $a_0 < a_1 < \cdots < a_k$, $k \le d$, such that the sign of the polynomial is constant on each of the *integer* intervals (a_j, a_{j+1}), $(-\infty, a_0)$, $(a_k, +\infty)$. This property of the function is expressible by a $\forall\Sigma^B_0$ sentence (when the coefficients of the polynomial and the a_j are taken from the binary sort), hence it holds in D that such elements $a_0, \dots, a_k$ exist for every polynomial over D.

Any atomic formula $\varphi(x)$ of L_{PA} with parameters from D is equivalent in *DOR* to the formula $f(x) \le 0$ for some $f \in D[x]$, hence $\varphi(D) := \{x : D \models \varphi(x)\}$ is a finite union of intervals. Sets of this kind form a Boolean algebra, hence $\varphi(D)$ is a finite union of intervals for every open formula φ. This implies induction for φ: if $D \models \varphi(0) \wedge \neg\varphi(u)$ for some $u > 0$, the interval I of $\varphi(D)$ containing 0 cannot be infinite from above, hence its larger end-point $v \in D$ satisfies $D \models \varphi(v) \wedge \neg\varphi(v+1)$. $\square$

Problem 5.2 *Does VTC^0 prove IOpen?*

In light of the discussion above, Problem 5.2 is essentially equivalent to the following: are there TC^0 root-finding algorithms for constant-degree polynomials *whose correctness is provable in VTC^0*? We remark that the complex-analytic tools we used in the proof of Theorem 4.1 are not available in VTC^0.

We note that already proving the totality of integer division in VTC^0 (i.e., formalization of a TC^0 integer division algorithm in VTC^0) is a nontrivial open[1] problem, thus Problem 5.2 may turn out to be too ambitious a goal. The following is a still interesting version of the question, which may be easier to settle:

Problem 5.3 *Does $VTC^0 + IMUL$ prove IOpen, where IMUL is a natural axiom postulating the totality of iterated integer multiplication?*

We also mention that it is not hard to prove in VTC^0 that binary integers form a $\mathbb{Z}$-ring, which implies all universal consequences of *IOpen* in the language of ordered rings. The problem is thus only with statements with a genuinely existential import (note that *IOpen* is a $\forall\exists$ theory).

Acknowledgements

I am grateful to Paul Beame and Yuval Filmus for useful discussions, and to anonymous referees for helpful suggestions.

The research was supported by grant IAA100190902 of GA AV ČR, project 1M0545 of MŠMT ČR, and RVO: 67985840.

[1]Hesse, Allender, and Barrington [90, Cor. 6.6] claim that the totality of integer division is provable in VTC^0 (or rather, in the theory C^0_2 of Johannsen and Pollett [109], RSUV-isomorphic to $VTC^0 + \Sigma^B_0\text{-}AC$, which is $\forall\Sigma^1_1$-conservative over VTC^0). However, the way it is stated there with no proof as an "immediate" corollary strongly suggests that the claim is due to a misunderstanding. See also [68, §IX.7.3].

Chapter VIII

Open induction in a bounded arithmetic for TC^0

Abstract

The elementary arithmetic operations $+, \cdot, \leq$ on integers are well-known to be computable in the weak complexity class TC^0, and it is a basic question what properties of these operations can be proved using only TC^0-computable objects, i.e., in a theory of bounded arithmetic corresponding to TC^0. We will show that the theory VTC^0 extended with an axiom postulating the totality of iterated multiplication (which is computable in TC^0) proves induction for quantifier-free formulas in the language $\langle +, \cdot, \leq \rangle$ (*IOpen*), and more generally, minimization for Σ^b_0 formulas in the language of Buss's S_2.

1 Introduction

Proof complexity is sometimes presented as the investigation of a three-way correspondence between propositional proof systems, theories of bounded arithmetic, and computational complexity classes. In particular, we can associate to a complexity class C satisfying suitable regularity conditions a theory T such that on the one hand, the provably total computable functions of T of certain logical form define exactly the C-functions in the standard model of arithmetic, and on the other hand, T proves fundamental deductive principles such as induction and comprehension for formulas that correspond to C-predicates. In this sense T provides a formalization of C-feasible reasoning: we can interpret provability in T as capturing the idea of what can be demonstrated when our reasoning capabilities are restricted to manipulation of objects and concepts of complexity C. The complexity class corresponding to a "minimal" theory that proves a given logical or combinatorial statement can be seen as a gauge of its proof complexity. Then a particularly natural question is, given a function or predicate X, which properties of X can be proved by reasoning whose complexity does not exceed that of X, that is, in a theory corresponding to the complexity class for which X is complete.

The main theme of this paper is what we can feasibly prove about the basic integer arithmetic operations $+, \cdot, \leq$. The matching complexity class is TC^0: $+$ and $\leq$ are computable in $\mathrm{AC}^0 \subseteq \mathrm{TC}^0$, while $\cdot$ is in TC^0, and it is in fact TC^0-complete under AC^0 (Turing) reductions. (In this paper, all circuit classes like TC^0 are assumed DLOGTIME-uniform unless stated otherwise.) TC^0 also includes many other functions related to arithmetic. First, $+$ and $\cdot$ are also TC^0-computable on rationals or Gaussian rationals. An important result of Hesse, Allender, and Barrington [90] based on earlier work by Beame, Cook, and Hoover [25] and Chiu, Davida, and Litow [52] states that integer division and iterated multiplication are TC^0-computable. As a consequence, one can compute in TC^0 approximations of functions presented by sufficiently nice power series, such as log, sin, or $x^{1/k}$, see e.g. Reif [158], Reif and Tate [159], Maciel and Thérien [126], and Hesse, Allender, and Barrington [90].

The more-or-less canonical arithmetical theory corresponding to TC^0 is VTC^0 (see Cook and Nguyen [68]). This is a two-sorted theory in the setup of Zambella [182], extending the base AC^0-theory V^0 by an axiom stating the existence of suitable counting functions, which gives it the power of TC^0. VTC^0 is equivalent ($RSUV$-isomorphic) to the one-sorted theory Δ^b_1-CR by Johannsen and Pollett [110], which is in turn $\forall\exists\Sigma^b_1$-conservative under the theory C^0_2 [109].

VTC^0 can define addition and multiplication on binary integers, and it proves basic identities governing these operations, specifically the axioms of discretely ordered rings (DOR). We are interested in what other properties of integers expressible in the language $L_{OR} = \langle 0, 1, +, -, \cdot, \leq\rangle$ of ordered rings are provable in VTC^0, and in particular, whether the theory can prove induction for a nontrivial class of formulas. Note that we should not expect the theory to prove induction for bounded existential formulas, or even its weak algebraic consequences such as the Bézout property: this would imply that integer gcd is computable in TC^0, while it is not even known to be in NC. However, this leaves the possibility that VTC^0 could prove induction for *open* (quantifier-free) formulas of L_{OR}, i.e., that it includes the theory *IOpen* introduced by Shepherdson [164].

Using an algebraic characterization of open induction and a witnessing theorem for VTC^0, the provability of *IOpen* in this theory is equivalent to the existence of TC^0 algorithms for approximation of real or complex roots of constant-degree univariate polynomials whose soundness can be proved in VTC^0. The existence of such algorithms in the "real world" is established in Chapter VII, but the argument extensively relies on tools from complex analysis (Cauchy integral formula, ...) that are not available in bounded arithmetic, hence it is unsuitable for formalization in VTC^0 or a similar theory.

The purpose of this paper is to demonstrate that *IOpen* is in fact provable in a mild extension of VTC^0. The argument naturally splits into two parts. We first formalize by a direct inductive proof a suitable version of the Lagrange inversion formula (LIF), which was also the core ingredient in the algorithm in Chapter VII. This allows us to compute approximations of a root of a polynomial f by means of partial sums of a power series expressing the inverse function of f, but only for polynomials obeying certain restrictions on coefficients. The second part of the argument is model-theoretic, using basic results from the theory of valued fields. The question whether a given DOR is a model of *IOpen* can be reduced to the question whether the completion of its fraction field under a valuation induced by its ordering is real-closed, and there is a simple criterion for recognizing real-closed valued fields. In our situation, LIF ensures the relevant field is henselian, which implies that the criterion is satisfied.

We do not work with VTC^0 itself, but with its extension $VTC^0 + IMUL$ including an axiom ensuring the totality of iterated multiplication. This theory corresponds to TC^0 just like VTC^0 does, as iterated multiplication is TC^0-computable. We need the extra axiom because it is not known whether VTC^0 can formalize the TC^0 algorithms for division and iterated multiplication of Hesse, Allender, and Barrington [90], and this subtle problem is rather tangential to the question of open induction and root approximation. As explained in more detail in Section 3, the *IMUL* axiom is closely related to the integer division axiom *DIV* which is implied by *IOpen*, hence its use is unavoidable in one way or another. In terms of the original theory VTC^0, our results show that $VTC^0 \vdash \mathit{IOpen}$ if and only if $VTC^0 \vdash \mathit{DIV}$.

We can strengthen the main result if we switch from L_{OR} to the language of Buss's one-sorted theories of bounded arithmetic. By formalizing the description of bounded Σ^b_0-definable sets due to Mantzivis [127], $VTC^0 + \mathit{IMUL}$ can prove the RSUV-translation of Buss's theory T^0_2, and in fact, of the Σ^b_0-minimization schema. In other words, T^0_2 and Σ^b_0-MIN are included in the theory Δ^b_1-$\mathit{CR} + \mathit{IMUL}$.

2 Preliminaries

A structure $\langle D, 0, 1, +, -, \cdot, \leq\rangle$ is an *ordered ring* if $\langle D, 0, 1, +, -, \cdot\rangle$ is a commutative (associative unital) ring, $\leq$ is a linear order on D, and $x \leq y$ implies $x + z \leq y + z$ and $xz \leq yz$ for all $x, y, z \in D$ such that $z \geq 0$. If D is an ordered ring, D^+ denotes $\{a \in D : a > 0\}$. A *discretely ordered ring* (*DOR*) is an ordered ring D such that 1 is the least element of D^+. Every DOR is an integral domain. An *ordered field* is an ordered ring which is a field. A *real-closed field* (*RCF*) is an ordered field R satisfying any of the following equivalent conditions:

- Every $a \in R^+$ has a square root in R, and every $f \in R[x]$ of odd degree has a root in R.
- R has no proper algebraic ordered field extension.
- The field $R(\sqrt{-1})$ is algebraically closed.
- R is elementarily equivalent to $\mathbb{R}$.

(In a RCF, $\leq$ is definable in terms of the ring structure, thus we can also call a field $\langle R, +, \cdot\rangle$ real-closed if it is the reduct of a RCF.) The *real closure* of an ordered field F is a RCF $\tilde{F}^{\mathrm{real}} \supseteq F$ which is an algebraic extension of F. Every ordered field has a unique real closure up to a unique F-isomorphism.

The theory *IOpen* consists of the axioms of ordered rings and the induction schema

$$\varphi(0) \land \forall x\, (\varphi(x) \to \varphi(x + 1)) \to \forall x \geq 0\, \varphi(x)$$

for open formulas φ (possibly with parameters). An *integer part* of an ordered field F is a discretely ordered subring $D \subseteq F$ such that every element of F is within distance 1 from an element of D. The following well-known characterization is due to Shepherdson [164].

Theorem 2.1 *Models of IOpen are exactly the integer parts of real-closed fields.* □

The criterion is often stated with the real closure of the fraction field of the model instead of a general real-closed field, but these two formulations are clearly equivalent, as an integer part D of a field R is also an integer part of any subfield $D \subseteq R' \subseteq R$.

In particular, models of *IOpen* are integer parts of their fraction fields. This amounts to provability of the division axiom

$$(DIV) \qquad \forall x > 0 \, \forall y \, \exists q, r \, (y = qx + r \wedge 0 \leq r < x)$$

in *IOpen*. (The uniqueness of q and r holds in any DOR.)

We define AC^0 as the class of languages recognizable by a DLOGTIME-uniform family of polynomial-size constant-depth circuits using $\neg$ and unbounded fan-in $\wedge$ and $\vee$ gates, or equivalently, languages computable by an $O(\log n)$-time alternating Turing machine with $O(1)$ alternations, or by a constant-time CRAM with polynomially many processors [91]. If we represent an n-bit binary string w by the finite structure $\langle \{0, \dots, n-1\}, <, +, \cdot, P_w \rangle$, where $P_w(i)$ iff the ith bit of w is 1, then AC^0 coincides with FO (languages definable by first-order sentences). A language B is AC^0*-reducible* to a language A if B is computable by a DLOGTIME-uniform family of polynomial-size constant-depth circuits using unbounded fan-in $\wedge$, $\vee$, $\neg$, and A-gates. The class of languages AC^0-reducible to A is its AC^0*-closure*.

TC^0, originally introduced as a nonuniform class by Hajnal, Maass, Pudlák, Szegedy, and Turán [87], is defined for our purposes as the AC^0-closure of MAJORITY. (Several problems TC^0-complete under AC^0 reductions are noted in Chandra, Stockmeyer, and Vishkin [50]; any of these could be used in place of MAJORITY.) Equivalently, TC^0 coincides with languages computable by $O(\log n)$-time threshold Turing machines with $O(1)$ thresholds, or by constant-time TRAM with polynomially many processors [141]. In terms of descriptive complexity, a language is in TC^0 iff the corresponding class of finite structures is definable in FOM, i.e., first-order logic with majority quantifiers [19].

In connection with bounded arithmetic, it is convenient to consider not just the complexity of languages, but of predicates $P(x_1, \dots, x_n, X_1, \dots, X_m)$ with several inputs, where X_i are binary strings as usual, and x_i are natural numbers written in unary. It is straightforward to generalize AC^0, TC^0, and similar classes to this context, see [68, §IV.3] for details. Likewise, we can consider computability of functions: if C is a complexity class, a unary number function $f(\vec{x}, \vec{X})$ is in FC if it is bounded by a polynomial in $\vec{x}$ and the lengths of $\vec{X}$, and its graph $f(\vec{x}, \vec{X}) = y$ is in C; a string function $F(\vec{x}, \vec{X})$ is in FC if the length of the output is polynomially bounded as above, and the bitgraph $G_F(\vec{x}, \vec{X}, y) \Leftrightarrow (F(\vec{x}, \vec{X}))_y = 1$ is in C. For simplicity, functions from FC will also be called just C-functions.

We will work with two-sorted (second-order) theories of bounded arithmetic in the form introduced by Zambella [182] as a simplification of Buss [37]. We refer the reader to Cook and Nguyen [68] for a general background on these theories as well as a detailed treatment of VTC^0, however, we include the main definitions here in order to fix our notation.

The language $L_2 = \langle 0, S, +, \cdot, \leq, \in, \|\cdot\| \rangle$ of second-order bounded arithmetic is a first-order language with equality with two sorts of variables, one for unary natural numbers, and one for finite sets thereof, which can also be interpreted as binary strings, or binary integers. The standard convention is that variables of the first sort are written with lowercase letters $x, y, z, \dots$, and variables of the second sort with uppercase letters $X, Y, Z, \dots$. While we adhere to this convention in the introductory material on the theories and their basic properties, we will not follow it in the less formal main part of the paper (we will mostly work with binary integers or rationals, and it looks awkward to write them all in uppercase). The symbols $0, S, +, \cdot, \leq$ of L_2 denote the usual arithmetic operations and relation on the unary sort; $x \in X$ is the elementhood predicate, and the intended meaning of the $\|X\|$ function is the least unary number strictly greater than all elements of X. This function is usually denoted as $|X|$, however (apart from the section on Buss's theories) we reserve the latter symbol for the absolute value on binary integers and rationals, which we will use more often. We write $x < y$ as an abbreviation for $x \leq y \wedge x \neq y$.

Bounded quantifiers are introduced by

$$\exists x \leq t\, \varphi \Leftrightarrow \exists x\, (x \leq t \wedge \varphi),$$
$$\exists X \leq t\, \varphi \Leftrightarrow \exists X\, (\|X\| \leq t \wedge \varphi),$$

where t is a term of unary sort not containing x or X (resp.). Universal bounded quantifiers, as well as variants of bounded quantifiers with strict inequalities, are defined in a similar way. A formula is Σ^B_0 if it contains no second-order quantifiers, and all its first-order quantifiers are bounded. The Σ^B_0-definable predicates in the standard model of arithmetic are exactly the AC^0 predicates. A formula is Σ^B_i if it consists of i alternating (possibly empty) blocks of bounded quantifiers, the first of which is existential, followed by a Σ^B_0 formula. We define Π^B_i formulas dually. Similarly, a formula is Σ^1_i (Π^1_i) if it consists of i alternating blocks of (possibly unbounded) quantifiers, the first of which is existential (universal, resp.), followed by a Σ^B_0 formula[1].

The theory V^0 in L_2 can be axiomatized by the basic axioms

$$\begin{array}{ll}
x + 0 = x & x + Sy = S(x + y) \\
x \cdot 0 = 0 & x \cdot Sy = x \cdot y + x \\
Sy \leq x \to y < x & \|X\| \neq 0 \to \exists x\, (x \in X \wedge \|X\| = Sx) \\
x \in X \to x < \|X\| & \forall x\, (x \in X \leftrightarrow x \in Y) \to X = Y
\end{array}$$

and the comprehension schema

$$(\varphi\text{-}\mathit{COMP}) \qquad \exists X \leq x\, \forall u < x\, (u \in X \leftrightarrow \varphi(u))$$

for Σ^B_0 formulas φ, possibly with parameters not shown (but with no occurrence of X). We denote the set X whose existence is postulated by φ-*COMP* as $\{u < x : \varphi(u)\}$. Using *COMP*, V^0 proves the induction and minimization schemata

$$(\varphi\text{-}\mathit{IND}) \qquad \varphi(0) \wedge \forall x\, \big(\varphi(x) \to \varphi(x+1)\big) \to \forall x\, \varphi(x),$$
$$(\varphi\text{-}\mathit{MIN}) \qquad \varphi(x) \to \exists y\, \big(\varphi(y) \wedge \forall z < y\, \neg\varphi(z)\big)$$

[1]Notice that bounded second-order quantifiers still count towards i, so these formula classes do not correspond in the one-sorted setting to the usual arithmetical hierarchy Σ^0_i, but to its restricted version where the formula after the main quantifier prefix is sharply bounded. We follow [68] in this usage; they only appear to define Σ^1_1, but we find it convenient to extend this notation to higher levels as well.

for Σ_0^B formulas φ. In particular, V^0 includes $I\Delta_0$ on the unary number sort.

Let $\langle x,y\rangle$ be a V^0-definable pairing function on unary numbers, e.g., $\langle x,y\rangle = (x+y)(x+y+1)/2+y$. We define $X^{[u]} = \{x : \langle u,x\rangle \in X\}$; this provides an encoding of sequences of sets by sets. We can encode sequences of unary numbers by putting $X^{(u)} = \|X^{[u]}\|$ (this is easily seen to be a Σ_0^B-definable function). For convenience, we also extend the pairing function to (standard-length) k-tuples by $\langle x_1,\dots,x_{k+1}\rangle = \langle\langle x_1,\dots,x_k\rangle, x_{k+1}\rangle$, and we write $X^{[u_1,\dots,u_k]} = X^{[\langle u_1,\dots,u_k\rangle]}$, $X^{(u_1,\dots,u_k)} = X^{(\langle u_1,\dots,u_k\rangle)}$.

VTC^0 is the extension of V^0 by the axiom

$$\forall n, X\, \exists Y \bigl(Y^{(0)} = 0 \wedge \forall i<n \bigl((i\notin X \to Y^{(i+1)} = Y^{(i)}) \wedge (i\in X \to Y^{(i+1)} = Y^{(i)}+1)\bigr)\bigr),$$

whose meaning is that for every set X there is a sequence Y supplying the counting function $Y^{(i)} = \mathrm{card}(X\cap\{0,\dots,i-1\})$.

Let Γ be a class of formulas, and T an extension of V^0. A string function $F(\vec{x},\vec{X})$ is a *provably total* Γ*-definable function* of T if its graph is definable in $\mathbb{N}$ by a formula $\varphi(\vec{x},\vec{X},Y)\in\Gamma$ such that $T\vdash \forall\vec{x},\vec{X}\,\exists!Y\,\varphi(\vec{x},\vec{X},Y)$; similarly for number functions. If $\Gamma = \Sigma_1^1$, such functions are also called *provably total recursive functions* of T. Note that one function may have many different definitions that are not T-provably equivalent; some of them may be provably total, while other are not.

The provably total recursive functions of V^0 and VTC^0 are FAC^0 and FTC^0, respectively. Moreover, we can use these functions freely in the sense that if we expand the languages of the theories with the corresponding function symbols, the resulting conservative extensions of V^0 and VTC^0 (respectively) prove the comprehension and induction schemata for Σ_0^B formulas of the expanded language; we will see more details in the next section.

Being AC^0, the ordering on binary integers is definable by a Σ_0^B formula, and addition is provably total in V^0. Likewise, multiplication and iterated addition are provably total Σ_1^B-definable functions of VTC^0. In fact, as shown in [68], the natural Σ_0^B definitions of $X<Y$ and $X+Y$ provably satisfy basic properties like commutativity and associativity in V^0, and similarly, there are natural definitions of $X\cdot Y$ and $\sum_{i<n} X^{[i]}$ provably total in VTC^0 such that VTC^0 proves their basic properties, including the inductive clauses

$$\sum_{i<0} X^{[i]} = 0,$$
$$\sum_{i<n+1} X^{[i]} = \sum_{i<n} X^{[i]} + X^{[n]}.$$

While Cook and Nguyen [68] normally use second-sort objects to denote nonnegative integers, it will be more convenient for us to make them represent all integers, which is easily accomplished by using one bit for sign. The definitions of $<$, $+$, $\cdot$, and $\sum_{i<n} X^{[i]}$ can be adapted in a straightforward way to this setting so that VTC^0 still proves their relevant properties, that is, the axioms of discretely ordered rings.

3 Iterated multiplication and division

As we already mentioned, it is not known whether VTC^0 can formalize the TC^0 algorithms of Hesse, Allender, and Barrington [90] for integer division and iterated multiplication. In particular, it is not known whether VTC^0 proves the sentence *DIV* (formulated for binary integers), which is a consequence of *IOpen*. This problem is rather tangential to the formalization of root finding, whence we bypass it by strengthening our theory appropriately.

It might seem natural just to work in the theory $VTC^0 + DIV$, however we will instead consider an axiom stating the totality of iterated multiplication in the following form:

$$(IMUL) \qquad \forall X, n\,\exists Y\,\forall i \le j < n\,\bigl(Y^{[i,i]} = 1 \land Y^{[i,j+1]} = Y^{[i,j]} \cdot X^{[j]}\bigr).$$

(The meaning is that for any sequence X of n binary integers, there is a triangular matrix Y with entries $Y^{[i,j]} = \prod_{k=i}^{j-1} X^{[k]}$.) One reason is simply that we need to use iterated multiplication at various places in the argument (in particular, to compute partial sums of power series), and we do not know whether $VTC^0 + DIV \vdash IMUL$. The more subtle reason is that we need the theory to be well-behaved in a certain technical sense that we will describe in more detail below, and it turns out that $VTC^0 + IMUL$ is the smallest well-behaved extension of $VTC^0 + DIV$.

Consider an extension $T \supseteq V^0$ proving that a particular polynomially bounded recursive (i.e., Σ^1_1-definable) function F is total, e.g. *DIV* or *IMUL*. While the most simplistic arguments employing F can get away with the mere fact that the value computed by F exists for a particular input, usually we need more than that. For example, we may want to use induction on a formula $\varphi(x)$ which involves F applied to an argument depending on x; since induction is obtained over V^0 by considering the least element of the set $\{x < a : \neg\varphi(x)\}$, we effectively need comprehension for (simple enough) formulas containing F, say, $\Sigma^B_0(F)$-*COMP*.

From a computational viewpoint, it is desirable that we can combine provably total recursive functions in various ways. For example, one of the basic TC^0 functions is iterated addition, and a natural way how we would like to apply it is to compute $\sum_{x<a} F(x)$ for a given provably total function F. More generally, we want the class of provably total recursive functions to be closed under AC^0 (or even TC^0 in our case) reductions, and as a simple special case, under *parallel repetition*: if we can compute a function $F(X)$, we want to be able to compute its *aggregate function* $F^*\colon \langle X_0, \dots, X_{n-1}\rangle \mapsto \langle F(X_0), \dots, F(X_{n-1})\rangle$ (where n is a part of the input). In more logical terms, it is desirable that T is closed under the *choice rule* Σ^B_0-AC^R: if $T \vdash \forall X\,\exists Y\,\varphi(X, Y)$, where $\varphi \in \Sigma^B_0$, then also $T \vdash \forall n\,\forall W\,\exists Z\,\forall i < n\,\varphi(W^{[i]}, Z^{[i]})$. This is a derived rule corresponding to the axiom of choice, also called replacement or bounded collection:

$$(\Sigma^B_0\text{-}AC) \qquad \forall i < n\,\exists Y \le m\,\varphi(i, Y, P) \to \exists Z\,\forall i < n\,\varphi(i, Z^{[i]}, P).$$

Unfortunately, none of the desiderata mentioned in the last two paragraphs hold automatically, even for theories of the simple form $V^0 + \forall X\,\exists! Y\,F(X) = Y$ (note that $VTC^0 + DIV$ is of such form): this axiom implies the totality of functions making a constant number of calls to F, but we cannot a priori construct functions involving an unbounded number of applications of F, such as the aggregate function F^*. However, Cook and Nguyen [68] show that the simple expedient of using F^* in the axiomatization instead of F leads to theories satisfying all the properties above.

Definition 3.1 Let $\delta(X,Y)$ be a Σ^B_0-formula such that V^0 proves

$$\delta(X,Y) \to \|Y\| \le t(X),$$
$$\delta(X,Y) \wedge \delta(X,Y') \to Y = Y'$$

for some term $t(X)$. The *Cook–Nguyen* (*CN*) *theory*[2] associated with δ is

$$V(\delta) = V^0 + \forall W, n\,\exists Z\,\forall i < n\,\delta(W^{[i]}, Z^{[i]}).$$

(That is, if F is a polynomially bounded function with an AC^0 graph defined by δ, which V^0 proves to be a partial function, then $V(\delta)$ is axiomatized by the statement that the aggregate function F^* is total.)

For example, VTC^0 can be formulated as a CN theory, as shown in [68, §IX.3].

Theorem 3.2 *Let $V(\delta)$ be a CN theory, and F the function whose graph is defined by δ.*

(i) *The provably total Σ^1_1-definable (or Σ^B_1-definable) functions of $V(\delta)$ are exactly the functions in the AC^0-closure of F.*

(ii) *$V(\delta)$ has a universal definitional (and therefore conservative) extension $\overline{V(\delta)}$ in a language $L_{\overline{V(\delta)}}$ consisting of Σ^B_1-definable functions of $V(\delta)$. The theory $\overline{V(\delta)}$ has quantifier elimination for $\Sigma^B_0(L_{\overline{V(\delta)}})$-formulas, and it proves $\Sigma^B_0(L_{\overline{V(\delta)}})$-COMP, $\Sigma^B_0(L_{\overline{V(\delta)}})$-IND, and $\Sigma^B_0(L_{\overline{V(\delta)}})$-MIN.*

(iii) *$V(\delta)$ is closed under Σ^B_0-AC^R, and $V(\delta) + \Sigma^B_0$-AC is Π^1_2-conservative over $V(\delta)$.*

Proof:

(i) and (ii) are Theorems IX.2.3, IX.2.14, and IX.2.16 in Cook and Nguyen [68].

(iii): If $V(\delta) \vdash \forall X\,\exists Y\,\varphi(X,Y)$ with $\varphi \in \Sigma^B_0$, there is an $L_{\overline{V(\delta)}}$-term $G(X)$ such that $\overline{V(\delta)} \vdash \varphi(X, G(X))$ by Herbrand's theorem, as $\overline{V(\delta)}$ is a universal theory, and φ is equivalent to an open formula. Then $\overline{V(\delta)}$, hence $V(\delta)$, proves

$$\forall W, n\,\exists Z\,Z = \{\langle i, y\rangle : i < n, y \in G(W^{[i]})\}$$

using $\Sigma^B_0(L_{\overline{V(\delta)}})$-$COMP$.

The Π^1_2-conservativity of Σ^B_0-AC over $V(\delta)$ follows from the closure under Σ^B_0-AC^R by cut elimination. Alternatively, see Theorem V.4.19 for a model-theoretic proof generalizing the result of Zambella [182] for V^0. □

[2]In [68], $V(\delta)$ is denoted VC, where the complexity class C is the AC^0-closure of F, and it is called the minimal theory associated with C. We refrain from this terminology as the theory is not uniquely determined by the complexity class: it depends on the choice of the C-complete function F, and of a particular Σ^B_0-formula defining the graph of F in $\mathbb{N}$. In particular, both VTC^0 and $VTC^0 + IMUL$ are "minimal" theories for the same class (TC^0), and it would be rather confusing to call them as such.

Lemma 3.3

(i) *$VTC^0 + IMUL$ is a CN theory.*

(ii) *$VTC^0 + IMUL \vdash DIV$.*

Proof:

(i): The main observation is that $VTC^0 + IMUL$ proves the totality of the aggregate function of iterated multiplication, that is,

$$(IMUL^*) \qquad \forall W, m, n\, \exists Z\, \forall k < m\, \forall i \le j < n\, \bigl(Z^{[k,i,i]} = 1 \wedge Z^{[k,i,j+1]} = Z^{[k,i,j]} \cdot W^{[k,j]}\bigr).$$

Given W, m, n, put $X = \{\langle nk + j, x\rangle : k < m, j < n, x \in W^{[k,j]}\}$ so that $X^{[nk+j]} = W^{[k,j]}$ for all $k < m$ and $j < n$, and let Y be as in ($IMUL$) for X, mn. Define

$$Z = \bigl\{\langle k, i, j, y\rangle : k < m, i \le j \le n, y \in Y^{[nk+i,nk+j]}\bigr\},$$

so that $Z^{[k,i,j]} = Y^{[nk+i,nk+j]}$ for $k < m$ and $i \le j \le n$. Then Z satisfies ($IMUL^*$).

Thus, $VTC^0 + IMUL = VTC^0 + IMUL^*$. The latter looks almost like a CN theory, except that the graph of the function specified in the axiom is not Σ^B_0, as it involves multiplication. (The official definition also does not allow an extra unary input, but this is benign as we could easily code X, n into a single set.) There are several ways how to get around this problem. For one, the whole machinery from [68, §IX.2] works fine if we take VTC^0 instead of V^0 as a base theory, and allow the use of $\Sigma^B_0(L_{\overline{VTC^0}})$ formulas. Alternatively, we can rewrite $IMUL$ to incorporate the definition of multiplication, say

$$\begin{aligned}
&\forall X, n\, \exists Y, Z\, \forall i \le j < n\, \forall x < \|X\|\, \bigl(Y^{[i,i]} = 1 \wedge Z^{[i,j,0]} = 0 \wedge Z^{[i,j,\|X\|]} = Y^{[i,j+1]}\\
(IMUL') \qquad &\qquad\qquad\qquad \wedge \bigl(x \notin X^{[j]} \to Z^{[i,j,x+1]} = Z^{[i,j,x]}\bigr)\\
&\qquad\qquad\qquad \wedge \bigl(x \in X^{[j]} \to Z^{[i,j,x+1]} = Z^{[i,j,x]} + 2^x Y^{[i,j]}\bigr)\bigr),
\end{aligned}$$

where $+$ and multiplication by 2^x can be given easy Σ^B_0 definitions. Since the entries of Z can be expressed as products of suitable Σ^B_0-definable sequences of integers, one can show in the same way as above that $IMUL'$, as well as the axiom $IMUL'^*$ stating the totality of the corresponding aggregate function, is provable in $VTC^0 + IMUL$. Conversely, the CN theory $V^0 + IMUL'^*$ proves VTC^0 (as it implies the totality of usual multiplication), hence it is equivalent to $VTC^0 + IMUL$.

(ii) can be shown by formalizing the reduction from [25]. Assume that we want to find $\lfloor Y/X \rfloor$, where $X \ge 1$. Choose $n, m > 0$ such that $2^{n-1} \le X \le 2^n$ and $Y \le 2^m$, and put

$$Z = \sum_{i<m} (2^n - X)^i 2^{n(m-1-i)}.$$

An easy manipulation of the sum shows that $XZ = 2^{nm} - (2^n - X)^m$, hence

$$2^{nm} - 2^{(n-1)m} \le XZ \le 2^{nm}.$$

Put $Q = \lfloor YZ/2^{nm} \rfloor$. Then

$$2^{nm} Y \ge XYZ \ge 2^{nm} QX > XYZ - 2^{nm} X \ge 2^{nm}(Y - X - 1),$$

hence $QX \le Y \le (Q + 1)X$. □

The more complicated converse reduction of iterated multiplication to division was formalized in bounded arithmetic by Johannsen [108] (building on Johannsen and Pollett [109]), but in a different setting, so let us see what his result gives us here. Johannsen works with a one-sorted theory $C^0_2[div]$, whose language consists of the usual Buss's language for S_2 expanded with $\dot{-}$, MSP, and most importantly $\lfloor x/y \rfloor$. It is axiomatized by a suitable version of $BASIC$, the defining axiom for division, the quantifier-free $LIND$ schema, and the axiom of choice $BB\Sigma^b_0$ for Σ^b_0 formulas in the expanded language.

We claim that $C^0_2[div]$ is $RSUV$-isomorphic to the theory $VTC^0 + DIV + \Sigma^B_0\text{-}AC$. We leave the interpretation of the latter theory in $C^0_2[div]$ to the reader as we will not need it, and focus on the other direction. It is straightforward to translate the symbols of the language save division to the corresponding operations on binary integers, and prove the translation of $BASIC$ in VTC^0. Of course, DIV allows us to translate the division function and prove its defining axiom, hence the only remaining problem is with the $LIND$ and BB schemata. Here we have to be a bit careful, as Σ^b_0 (or even quantifier-free) formulas in the language of $C^0_2[div]$ do not translate to Σ^B_0 formulas in the language of V^0.

Let DIV^* denote the axiom stating the totality of the aggregate function of division, or rather, of its expanded version with witnesses for multiplication as in the proof of Lemma 3.3, so that $T = VTC^0 + DIV^*$ is a CN theory. By an application of choice, $VTC^0 + DIV + \Sigma^B_0\text{-}AC$ proves DIV^*. Let $\overline{T}$ be the universal conservative extension of T from Theorem 3.2, which includes function symbols for division and for TC^0 functions like multiplication. Since Σ^b_0 formulas in the language of $C^0_2[div]$ translate to $\Sigma^B_0(L_{\overline{T}})$ formulas, Theorem 3.2 implies that $\overline{T}$, and therefore $T \subseteq VTC^0 + DIV + \Sigma^B_0\text{-}AC$, proves the translation of open (or even Σ^b_0) $LIND$. As for the axiom of choice, every $\Sigma^B_0(L_{\overline{T}})$ formula is equivalent to a Σ^B_1 formula in the language of V^0, and $\Sigma^B_0\text{-}AC$ implies $\Sigma^B_1\text{-}AC$, hence the translation of $BB\Sigma^b_0$ is provable in $\overline{T} + \Sigma^B_0\text{-}AC$, and thus in $VTC^0 + DIV + \Sigma^B_0\text{-}AC$ by the conservativity of $\overline{T}$ over T.

This, together with provability of iterated multiplication in $C^0_2[div]$, implies the following:

Theorem 3.4 (Johannsen [108]) *$VTC^0 + DIV + \Sigma^B_0\text{-}AC$ proves $IMUL$.* □

Corollary 3.5 *$VTC^0 + IMUL = VTC^0 + DIV^*$ is the smallest CN theory including $VTC^0 + DIV$.*

Proof: Since $VTC^0 + DIV^*$ is a CN theory, Theorem 3.2 implies that $VTC^0 + DIV + \Sigma^B_0\text{-}AC$ is Π^1_2-conservative over $VTC^0 + DIV^*$, hence $VTC^0 + DIV^* \vdash IMUL$ by Theorem 3.4. Conversely, every CN theory (such as $VTC^0 + IMUL$, by Lemma 3.3) that proves DIV also proves DIV^*, using its closure under $\Sigma^B_0\text{-}AC^R$. □

Corollary 3.6 *$VTC^0 \vdash DIV$ if and only if $VTC^0 \vdash IMUL$.*

Proof: VTC^0 is a CN theory. □

The alert reader may have noticed that the reason why $IMUL$ yields a CN theory while this is unclear for DIV is not due to any deep property of iterated multiplication that would make it inherently better-behaved than division, but because we made it so by formulating the axiom in the slightly redundant form using a triangular matrix of partial products. There does not seem to be any particular reason we should expect to get a CN theory if we formulate the axiom more economically, using only a one-dimensional array consisting of the products $\prod_{j<i} X^{[j]}$. In view of this, the decision to axiomatize the theory using *IMUL* rather than DIV^* is mostly a matter of esthetic preference and convenience. Even in its triangular form, the *IMUL* axiom is a fairly natural rendering of the idea of computing iterated products, whereas the usage of an aggregate function in DIV^* is overtly a technical crutch. Moreover, we will be using iterated products more often than division, and while *DIV* has a straightforward proof in $VTC^0 + IMUL$ as indicated above, we would have to rely on the complicated argument from [108] to derive *IMUL* if we based the theory on DIV^*, making the main result of the paper less self-contained.

We mention another possibility for axiomatization of our theory, using the powering axiom

$$(POW) \qquad \forall X, n\, \exists Y\, \forall i < n\, \bigl(Y^{[0]} = 1 \wedge Y^{[i+1]} = Y^{[i]} \cdot X\bigr)$$

(here it makes no difference whether we use a linear or triangular array of witnesses) and its aggregate function version POW^*. Over VTC^0, we clearly have $IMUL \vdash POW^* \vdash POW$. The argument in Lemma 3.3 (ii) only needed the sequence of powers $(2^n - X)^i$, $i \le m$ apart from VTC^0, hence it actually shows $POW \vdash DIV$. Since $VTC^0 + POW^*$ is a CN theory, this implies $VTC^0 + POW^* = VTC^0 + IMUL$. In fact, one can also show that $VTC^0 + POW = VTC^0 + DIV$ by formalizing the reduction of powering to division from [25]. The key point is that the result of a single division is enough to reconstruct the whole sequence of powers $X^0, \dots, X^n$, hence we do not need any aggregate functions. If $X < 2^k$ and $m = k(n+1)+1$, let $2^{nm} = (2^m - X)Q + R$ with $R < 2^m - X$ using *DIV*, write $Q = \sum_{i<n} Y^{[i]} 2^{(n-1-i)m}$ with $Y^{[i]} < 2^m$, and put $Y^{[n]} = R$. Then one can show $Y^{[0]} = 1$ and

$$Y^{[j]} \le 2^{kj} \wedge \forall i < j\, Y^{[i+1]} = XY^{[i]}$$

by induction on $j \le n$. We leave the details to the interested reader.

Let us also mention that while it is unclear whether the soundness of the Hesse–Allender–Barrington algorithms for division and iterated multiplication is provable in VTC^0, it seems very likely that it is provable in $VTC^0 + IMUL$. If true, this would imply that $VTC^0 + IMUL$ is Π^1_1-axiomatizable over VTC^0 by the sentence asserting the soundness of the algorithm, and it can be formulated as a purely universal theory in the language of $\overline{VTC^0}$. A priori, the *IMUL* axiom is only $\forall\Sigma^B_1$.

Even though we do not know whether *IMUL* is provable in VTC^0 itself, we can place it reasonably low in the usual hierarchy of theories for small complexity classes: it is straightforward to show that $VTC^0 + IMUL$ is included in the theory VNC^2 (and even VTC^1, if anyone bothered to define such a theory) by formalizing the computation of iterated products by a balanced tree of binary products.

As stated in the Introduction, the provability of *IOpen* in VTC^0 or $VTC^0 + IMUL$ can be phrased in terms of TC^0 root-finding algorithms. There are several ways of expressing this connection precisely; one version reads as follows.

Proposition 3.7 *$VTC^0 + IMUL$ proves IOpen if and only if for every constant $d > 0$ there exist $L_{\overline{VTC^0+IMUL}}$-terms $R_-(A_0, \dots, A_d, X, Y, E)$ and $R_+(A_0, \dots, A_d, X, Y, E)$ such that the theory* *proves*

$$(1)\quad X < Y \wedge F(X) < 0 < F(Y) \wedge E > 0 \wedge Z_\pm = R_\pm(A_0, \dots, A_d, X, Y, E)$$
$$\to X < Z_- < Z_+ < Y \wedge Z_+ - Z_- < E \wedge F(Z_-) < 0 < F(Z_+),$$

where all second-sort variables are interpreted as binary rational numbers (fractions), and $F(X)$ denotes $A_dX^d + A_{d-1}X^{d-1} + \cdots + A_0$.

Proof:

Left-to-right: the statement that for every $A_0, \dots, A_d, X, Y, E$ there exist Z_-, Z_+ satisfying (1) is provable in *IOpen* (in the real closure of the model, there is a root of F between X and Y where F changes sign, and this root can be arbitrarily closely approximated from either side in the fraction field of the model using Theorem 2.1). By assumption, the same statement is also provable in $\overline{VTC^0 + IMUL}$. Since the latter is a universal theory whose terms are closed under definitions by cases, Herbrand's theorem implies that there are terms R_-, R_+ witnessing Z_-, Z_+.

Right-to-left: Let D be a DOR induced by a model of $VTC^0 + IMUL$, K its fraction field, and F a polynomial with coefficients in D. Since F can change sign only $\deg(F)$ times, a repeated use of (1) gives us elements $Z_0 < Z_1 < \cdots < Z_k$ of K such that F has (in K) a constant sign on each interval $(-\infty, Z_0)$, (Z_k, ∞), and (Z_i, Z_{i+1}), except when $Z_{i+1} - Z_i < 1$. We have $D \vDash DIV$, hence we can approximate each Z_i in D within distance 1; it follows that in D, F is positive on a finite union of (possibly degenerate) intervals. Every L_{OR} open formula φ is equivalent to a Boolean combination of formulas of the form $F(X) > 0$, hence $\{X \in D : X \geq 0 \wedge \neg\varphi(X)\}$ is also a finite union of intervals, and as such it has a least element if nonempty. Thus, D satisfies induction for φ. $\square$

Note that $L_{\overline{VTC^0+IMUL}}$-terms denote TC^0 algorithms (employing iterated multiplication), hence the gist of the conclusion of Proposition 3.7 is that $VTC^0 + IMUL$ proves the soundness of a TC^0 degree-d polynomial root-approximation algorithm for each d. The details can be varied; for example, we could drop X and Y, and make the algorithm output approximations to all real roots of the polynomial, or even complex roots. However, such modifications make it more difficult to state what exactly the "soundness" of the algorithm means.

4 Working in $VTC^0 + IMUL$

As we already warned the reader, the objects we work with most often in this paper are binary numbers (integer or rational), and we will employ common mathematical notation rather than the formal conventions used in [68]: in particular, we will typically denote numbers by lowercase letters (conversely, we will occasionally denote unary numbers by capital letters), and we will write x_i for the ith member of a sequence x (which may be a constant-length tuple, a variable-length finite sequence encoded by a set as in Section 2, or an infinite sequence given by a TC^0 function with unary input i). We do not distinguish binary and unary numbers in notation; we will either explicitly mention which numbers are unary, or it will be assumed from the context:

unary natural numbers appear as indices and lengths of sequences, as powering exponents, and as bound variables in iterated sums $\sum_{i=0}^n x_i$ and products $\prod_{i=0}^n x_i$.

By Theorem 3.2, we can use $L_{\overline{VTC^0+IMUL}}$-function symbols (i.e., TC^0 algorithms) freely in the arguments. In particular, we can use basic arithmetic operations on integers, including iterated sums and products. Iterated sums satisfy the recursive identities

$$\sum_{i<0} x_0 = 0,$$
$$\sum_{i<n+1} x_i = \sum_{i<n} x_i + x_n,$$

and other basic properties can be easily proved by induction, for example

$$\begin{aligned}\sum_{i<n}(x_i+y_i) &= \sum_{i<n} x_i + \sum_{i<n} y_i,\\ \sum_{i<n} yx_i &= y\sum_{i<n} x_i,\\ \sum_{i<n+m} x_i &= \sum_{i<n} x_i + \sum_{i<m} x_{n+i}.\end{aligned} \tag{2}$$

In particular, $VTC^0 + IMUL$ proves that if π is a permutation of $\{0,\dots,n-1\}$, then

$$\sum_{i<n} x_i = \sum_{i<n} x_{\pi(i)}. \tag{3}$$

(In order to see this, show $\sum_{i<m} x_i = \sum_{i<n} x_{\pi(i)}[\pi(i) < m]$ by induction on $m \le n$ using (2), where $[\cdots]$ denotes the Iverson bracket.) This allows us to make sense of more general sums $\sum_{i\in I} x_i$ where the indices run over a TC^0-definable collection of objects (e.g., tuples of unary numbers) that can be enumerated by a subset of some $\{0,\dots,n-1\}$; the identity (3) shows that the value of such a sum is independent of the enumeration. For example, we can write

$$f(n) = \sum_{i+j=n} x_{i,j},$$

meaning a sum over all pairs of numbers $\langle i,j\rangle$ such that $i+j=n$. We can also prove the double counting identity

$$\sum_{i<n}\sum_{j<m} x_{i,j} = \sum_{\substack{i<n\\ j<m}} x_{i,j} = \sum_{j<m}\sum_{i<n} x_{i,j} \tag{4}$$

by first showing $\sum_{i<n}\sum_{j<m} x_{i,j} = \sum_{k<nm} x_{\lfloor k/m\rfloor,k \bmod m}$ by induction on n using (2), and then (3) implies that other enumerations of the same set of pairs give the same result. Likewise, we can show

$$\Bigl(\sum_{i<n} x_i\Bigr)\Bigl(\sum_{i<m} y_i\Bigr) = \sum_{\substack{i<n\\ j<m}} x_i y_j. \tag{5}$$

Iterated products can be treated the same way as sums, mutatis mutandis.

Rational numbers can be represented in $VTC^0 + IMUL$ as pairs of integers standing for fractions a/b, where $b > 0$. We will not assume fractions to be reduced, as we cannot compute integer gcd. Arithmetic operations can be extended to rational numbers in $VTC^0 + IMUL$ in the obvious way, for example

$$\sum_{i<n} \frac{a_i}{b_i} := \frac{\sum_{i<n} a_i \prod_{j \neq i} b_j}{\prod_{i<n} b_i}.$$

$VTC^0 + IMUL$ knows the rationals form an ordered field, being the fraction field of a DOR. The properties of iterated sums and products we established above for integers also hold for rationals.

Using iterated products, we can define factorials and binomial coefficients

$$n! = \prod_{i=1}^{n} i, \qquad \binom{n}{m} = \frac{n!}{m!(n-m)!}$$

for unary natural numbers $n \geq m$. A priori, $n!$ is a binary integer, and $\binom{n}{m}$ a binary rational; however, the definition easily implies the identities

$$\binom{n}{0} = \binom{n}{n} = 0, \qquad \binom{n+1}{m+1} = \binom{n}{m} + \binom{n}{m+1},$$

from which one can show by induction on n that $\binom{n}{m}$ is an integer for all $m \leq n$. We can also prove by induction on n the binomial formula

$$(x+y)^n = \sum_{i \leq n} \binom{n}{i} x^i y^{n-i}$$

for rational x, y. More generally, we can define the multinomial coefficients

$$\binom{n}{n_1, \dots, n_d} = \frac{n!}{n_1! \cdots n_d!} = \binom{n}{n_1}\binom{n-n_1}{n_2} \cdots \binom{n - n_1 - \cdots - n_{d-1}}{n_d}$$

for a standard constant d and unary $n = n_1 + \cdots + n_d$, and we can prove the multinomial formula

$$(6) \qquad (x_1 + \cdots + x_d)^n = \sum_{n_1 + \cdots + n_d = n} \binom{n}{n_1, \dots, n_d} x_1^{n_1} \cdots x_d^{n_d}$$

by metainduction on d.

5 Lagrange inversion formula

The Lagrange inversion formula (LIF) is an expression for the coefficients of the (compositional) inverse $g = f^{-1}$ of a power series f. In this section, we will formalize in $VTC^0 + IMUL$ variants of LIF for the special case where f is a constant-degree polynomial; we first show that g inverts f as a formal power series, and then with the help of a suitable bound on the coefficients of g, we show that the series $g(w)$ is convergent for small enough w; this means that under some restrictions, partial sums of $g(-a_0)$ approximate a root of the polynomial $f(x)+a_0$.

LIF, specifically the equivalent identity (9), has a simple combinatorial interpretation in terms of trees which allows for a straightforward bijective proof. However, this proof relies on exact counting of exponentially many objects, and as such it cannot be formalized in $VTC^0 + IMUL$. In contrast, the inductive proof we give below proceeds by low-level manipulations of sums and products; while it lacks conceptual clarity, it is elementary enough to go through in our weak theory.

We introduce some notation for convenience. Let us fix a standard constant $d \geq 1$. We are going to work extensively with sequences $m = \langle m_2, \dots, m_d \rangle$ of length $d-1$ of unary nonnegative integers. We will use subscripts $i = 2, \dots, d$ to extract elements of the sequence as indicated, and we will employ superscripts (and primes) to label various sequences used at the same time; these do not denote exponentiation. If m^1 and m^2 are two such sequences, we define $m^1 + m^2$ and $m^1 - m^2$ coordinatewise (i.e., $(m^1+m^2)_i = m^1_i + m^2_i$), we write $m^1 \leq m^2$ if $m^1_i \leq m^2_i$ for all $i = 2, \dots, d$, and $m^1 \lneq m^2$ if $m^1 \leq m^2$ and $m^1 \neq m^2$. We define the generalized Catalan numbers

$$C_m = \frac{\left(\sum_{i=2}^d i m_i\right)!}{\left(\sum_{i=2}^d (i-1)m_i + 1\right)! \prod_{i=2}^d m_i!}.$$

Theorem 5.1 *$VTC^0 + IMUL$ proves the following for every constant $d \geq 1$: let*

$$f(x) = x + \sum_{k=2}^d a_k x^k$$

be a rational polynomial, and let

$$g(w) = \sum_{n=1}^\infty b_n w^n$$

be the formal power series (*with unary indices*) *defined by*

$$b_n = \sum_{\sum_i (i-1)m_i = n-1} C_m \prod_{i=2}^d (-a_i)^{m_i}. \tag{7}$$

Then $f(g(w)) = w$ as formal power series.

Remark 5.2 The sum in (7) runs over sequences $m = \langle m_2, \dots, m_d \rangle$ satisfying the constraint $\sum_{i=2}^d (i-1)m_i = n-1$; since this implies $m_2, \dots, m_d < n$, there are at most n^{d-1} such sequences, hence the sum makes sense in $VTC^0 + IMUL$.

The power series identity $f(g(w)) = w$ in the conclusion of the theorem amounts to $b_1 = 1$, and the recurrence

$$b_n = \sum_{k=2}^d (-a_k) \sum_{n_1 + \dots + n_k = n} b_{n_1} \cdots b_{n_k} \qquad (n > 1). \tag{8}$$

Rather than developing a general theory of formal power series in $VTC^0 + IMUL$, we take this as a *definition* of $f(g(w)) = w$.

Proof: After plugging in the definition of b_n, both sides of (8) can be written as polynomials in $-a_2, \dots, -a_d$ with rational (actually, integer) coefficients by several applications of (5). Moreover, b_{n_j} contains only monomials $\prod_i (-a_i)^{m_i^j}$ with $\sum_i (i-1)m_i^j = n_j - 1$. Thus, the right-hand side contains monomials $\prod_i (-a_i)^{m_i}$ with $m_i = \sum_j m_i^j + \delta_i^k$, where δ_i^k is Kronecker's delta. We have $\sum_i (i-1)m_i = \sum_{i,j}(i-1)m_i^j + k - 1 = \sum_j (n_j - 1) + k - 1 = n - 1$, which is the same constraint as on the left-hand side. In order to prove (8), it thus suffices to show that the coefficients of the monomials $\prod_i (-a_i)^{m_i}$ satisfying $\sum_i (i-1)m_i = n-1$ are the same on both sides of (8). This is easily seen to be equivalent to the following identity for every sequence m:

$$C_m = \sum_{k=2}^{d} \sum_{m^1 + \dots + m^k = m - \delta^k} C_{m^1} \cdots C_{m^k} \qquad (m \neq \vec{0}). \tag{9}$$

(Here, we treat Kronecker's delta as the sequence $\delta^k = \langle \delta_2^k, \dots, \delta_d^k \rangle$.) We will prove (9) by induction on $\sum_i m_i$, simultaneously with the identities

$$\sum_{m'+m''=m} \bigl(\textstyle\sum_i (i-1)m_i' + 1\bigr) C_{m'} C_{m''} = \bigl(\textstyle\sum_i i m_i + 1\bigr) C_m, \tag{10}$$

$$\sum_{m^1+\dots+m^k=m} C_{m^1} \cdots C_{m^k} = \frac{\bigl(\sum_i i m_i + k - 1\bigr)!\, k}{\bigl(\sum_i (i-1)m_i + k\bigr)! \prod_i m_i!} \qquad (k = 1, \dots, d). \tag{11}$$

The reader may find it helpful to consider the following combinatorial explanation of the identities, even though it cannot be expressed in $VTC^0 + IMUL$. First, C_m counts the number of ordered rooted trees with $m_2, \dots, m_d$ nodes of out-degree $2, \dots, d$, respectively, and the appropriate number (i.e., $\sum_i (i-1)m_i + 1$) of leaves. Indeed, such a tree can be uniquely described by the sequence of out-degrees of its nodes in preorder. One checks easily that every string with $m_2, \dots, m_d$ occurrences of $2, \dots, d$, resp., and $\sum_i (i-1)m_i + 1$ occurrences of 0, has a unique cyclic shift that is a valid representation of a tree, so there are

$$\frac{1}{\sum_i i m_i + 1} \binom{\sum_i i m_i + 1}{\sum_i (i-1)m_i + 1, m_2, \dots, m_d} = C_m$$

such trees. The left-hand side of (11) thus counts k-tuples of trees with a prescribed total number of nodes of out-degree $2, \dots, d$; a similar argument as above shows their number equals the right-hand side (every string with the appropriate number of symbols of each kind has exactly k cyclic shifts that are concatenations of representations of k trees). The main identity (9) expresses that a tree with more than one node can be uniquely decomposed as a root of out-degree $k = 2, \dots, d$ followed by a k-tuple of trees. Finally, (10) expresses that a pair of trees t', t'' together with a distinguished leaf x of t' uniquely represent a tree t with a distinguished node x, namely the tree obtained by identifying the root of t'' with x.

Let us proceed with the formal proof by induction. Assume that (9), (10), and (11) hold for all m' such that $m' \lneq m$, we will prove them for m.

(9): If $m \neq \vec{0}$, we have

$$\sum_{k=2}^{d} \sum_{m^1+\dots+m^k=m-\delta^k} C_{m^1}\cdots C_{m^k} = \sum_{\substack{k=2\\ m_k>0}}^{d} \frac{\bigl(\sum_i im_i - 1\bigr)!\,k}{\bigl(\sum_i (i-1)m_i+1\bigr)!\prod_{i\neq k} m_i!\,(m_k-1)!}$$
$$= \frac{\bigl(\sum_i im_i - 1\bigr)!}{\bigl(\sum_i (i-1)m_i+1\bigr)!\prod_i m_i!}\sum_{\substack{k=2\\ m_k>0}}^{d} km_k$$
$$= \frac{\bigl(\sum_i im_i\bigr)!}{\bigl(\sum_i (i-1)m_i+1\bigr)!\prod_i m_i!} = C_m,$$

using (11) for $m-\delta^k \lneq m$.

(10): If $m=\vec{0}$, the statement holds. Otherwise, we have

$$\bigl(\textstyle\sum_i im_i+1\bigr)C_m$$
$$= C_m + \bigl(\textstyle\sum_i im_i\bigr)\displaystyle\sum_{k=2}^{d}\sum_{m^1+\dots+m^k=m-\delta^k} C_{m^1}\cdots C_{m^k}$$
$$= C_m + \sum_{k=2}^{d}\sum_{m^1+\dots+m^k=m-\delta^k}\sum_{j=1}^{k}\bigl(\textstyle\sum_i im_i^j+1\bigr)C_{m^1}\cdots C_{m^k}$$
$$= C_m + \sum_{k=2}^{d} k \sum_{m^1+\dots+m^k=m-\delta^k}\bigl(\textstyle\sum_i im_i^k+1\bigr)C_{m^1}\cdots C_{m^k} \tag{12}$$
$$= C_m + \sum_{k=2}^{d} k \sum_{m^1+\dots+m^k+m''=m-\delta^k}\bigl(\textstyle\sum_i (i-1)m_i^k+1\bigr)C_{m^1}\cdots C_{m^k}C_{m''}$$
$$= C_m + \sum_{\substack{m'+m''=m\\ m'\neq\vec{0}}} C_{m''}\sum_{k=2}^{d} k \sum_{m^1+\dots+m^k=m'-\delta^k}\bigl(\textstyle\sum_i (i-1)m_i^k+1\bigr)C_{m^1}\cdots C_{m^k}$$
$$= C_m + \sum_{\substack{m'+m''=m\\ m'\neq\vec{0}}} C_{m''}\sum_{k=2}^{d}\sum_{m^1+\dots+m^k=m'-\delta^k}\sum_{j=1}^{k}\bigl(\textstyle\sum_i (i-1)m_i^j+1\bigr)C_{m^1}\cdots C_{m^k} \tag{13}$$
$$= C_m + \sum_{\substack{m'+m''=m\\ m'\neq\vec{0}}} C_{m''}\sum_{k=2}^{d}\sum_{m^1+\dots+m^k=m'-\delta^k}\bigl(\textstyle\sum_i (i-1)m_i'+1\bigr)C_{m^1}\cdots C_{m^k}$$
$$= C_m + \sum_{\substack{m'+m''=m\\ m'\neq\vec{0}}}\bigl(\textstyle\sum_i (i-1)m_i'+1\bigr)C_{m''}C_{m'}$$
$$= \sum_{m'+m''=m}\bigl(\textstyle\sum_i (i-1)m_i'+1\bigr)C_{m'}C_{m''},$$

using (9) for m and $m' \le m$, and (10) for $m^k \lneq m$. We derive line (12) by observing that the

k sums

$$\sum_{m^1+\dots+m^k=m-\delta^k} \bigl(\textstyle\sum_i im_i^j+1\bigr)C_{m^1}\cdots C_{m^k} \qquad (j=1,\dots,k)$$

have the same value due to symmetry (i.e., by an application of (3)). Line (13) is similar.

(11): By metainduction on $k=1,\dots,d$. The case $k=1$ is the definition of C_m. Assuming the statement holds for k, we prove it for $k+1$ from the identity

$$\begin{aligned}
k\bigl(\textstyle\sum_i(i-1)m_i+k+1\bigr) &\sum_{m^1+\dots+m^{k+1}=m} C_{m^1}\cdots C_{m^{k+1}}\\
&= k\sum_{m^1+\dots+m^{k+1}=m}\sum_{j=1}^{k+1}\bigl(\textstyle\sum_i(i-1)m_i^j+1\bigr)C_{m^1}\cdots C_{m^{k+1}}\\
&= k(k+1)\sum_{m^1+\dots+m^{k+1}=m}\bigl(\textstyle\sum_i(i-1)m_i^{k+1}+1\bigr)C_{m^1}\cdots C_{m^{k+1}}\\
&= k(k+1)\sum_{m^1+\dots+m^k=m}C_{m^1}\cdots C_{m^{k-1}}\sum_{m'+m''=m^k}\bigl(\textstyle\sum_i(i-1)m_i'+1\bigr)C_{m'}C_{m''}\\
&= k(k+1)\sum_{m^1+\dots+m^k=m}\bigl(\textstyle\sum_i im_i^k+1\bigr)C_{m^1}\cdots C_{m^k}\\
&= (k+1)\sum_{m^1+\dots+m^k=m}\sum_{j=1}^{k}\bigl(\textstyle\sum_i im_i^j+1\bigr)C_{m^1}\cdots C_{m^k}\\
&= (k+1)\bigl(\textstyle\sum_i im_i+k\bigr)\sum_{m^1+\dots+m^k=m}C_{m^1}\cdots C_{m^k}
\end{aligned}$$

using (10) for $m^k\le m$. □

Lemma 5.3 *VTC^0+IMUL proves: let f,g be as in Theorem 5.1, and $a=\max\bigl\{1,\sum_i|a_i|\bigr\}$. Then $|b_n|\le(4a)^{n-1}$ for every n.*

Proof: We can estimate

$$\begin{aligned}
|b_n| &\le a^{n-1}\sum_{\sum_i(i-1)m_i=n-1}C_m\prod_{i=2}^d\bigl(a^{1-i}|a_i|\bigr)^{m_i}\\
&= \frac{a^{n-1}}{n}\sum_{\sum_i(i-1)m_i=n-1}\binom{n-1+\sum_i m_i}{n-1,m_2,\dots,m_d}\prod_{i=2}^d\bigl(a^{1-i}|a_i|\bigr)^{m_i}\\
&\le \frac{a^{n-1}}{n}\sum_{t=n-1}^{2(n-1)}\sum_{s+\sum_i m_i=t}\binom{t}{s,m_2,\dots,m_d}\prod_{i=2}^d\bigl(a^{-1}|a_i|\bigr)^{m_i}\\
&= \frac{a^{n-1}}{n}\sum_{t=n-1}^{2(n-1)}\Bigl(1+a^{-1}\sum_{i=2}^d|a_i|\Bigr)^t\\
&\le a^{n-1}\Bigl(1+a^{-1}\sum_{i=2}^d|a_i|\Bigr)^{2(n-1)}\le a^{n-1}2^{2(n-1)}
\end{aligned}$$

using the multinomial formula (6). □

Example 5.4 The bound in Lemma 5.3 is reasonably tight even in the "real world". Let $a > 0$ be a real number, and put $f(x) = x - ax^2$. Then g is its inverse function $g(w) = (1 - \sqrt{1 - 4aw})/2a$, whose radius of convergence is the modulus of the nearest singularity, namely $1/4a$. Thus, for every $\varepsilon > 0$, $|b_n| \geq (4a - \varepsilon)^n$ for infinitely many n. In fact, the Stirling approximation for Catalan numbers gives $b_n = \Theta\big((4a)^n n^{-3/2}\big)$.

Theorem 5.5 *$VTC^0 + IMUL$ proves the following for every constant $d \geq 1$. Let $h(x) = \sum_{i=0}^{d} a_i x^i$ be a rational polynomial with linear coefficient $a_1 = 1$. Put $f = h - a_0$, let g and b_n be as in Theorem 5.1, $a = \max\{1, \sum_{i=2}^{d}|a_i|\}$, $\alpha = 4a|a_0|$, and let*

$$x_N = \sum_{n=1}^{N} b_n(-a_0)^n$$

denote the Nth partial sum of $g(-a_0)$ for every unary natural number N. If

$$|a_0| < \frac{1}{4a},$$

then

$$|x_N| \leq \frac{|a_0|}{1-\alpha}, \tag{14}$$

$$|x_N - x_M| \leq \frac{|a_0|\alpha^{N-1}}{1-\alpha}, \tag{15}$$

$$|h(x_N)| \leq N^d |a_0| \alpha^N \tag{16}$$

for every unary $M \geq N \geq 1$.

Proof: Lemma 5.3 gives

$$|x_N| \leq \sum_{n=1}^{N} |a_0|^n (4a)^{n-1} = |a_0| \sum_{n=0}^{N-1} \alpha^n \leq \frac{|a_0|}{1-\alpha}.$$

The proof of (15) is similar. As for (16), we have

$$\begin{aligned} h(x_N) &= a_0 + \sum_{k=1}^{d} a_k \sum_{n_1,\dots,n_k=1}^{N} b_{n_1} \cdots b_{n_k} (-a_0)^{n_1+\dots+n_k} \\ &= \sum_{k=1}^{d} a_k \sum_{\substack{n_1,\dots,n_k=1 \\ n_1+\dots+n_k>N}}^{N} b_{n_1} \cdots b_{n_k} (-a_0)^{n_1+\dots+n_k}, \end{aligned} \tag{17}$$

as

$$\sum_{k=1}^{d} a_k \sum_{n_1+\dots+n_k=n} b_{n_1} \cdots b_{n_k} = \delta_n^1$$

for all $n \leq N$ by Theorem 5.1. Note that the inner sum in (17) is empty for $k = 1$, thus

$$\begin{aligned} |h(x_N)| &\leq \sum_{k=2}^{d} |a_k| \sum_{\substack{n_1,\dots,n_k=1 \\ n_1+\dots+n_k>N}}^{N} (4a)^{-k} \big(4a|a_0|\big)^{n_1+\dots+n_k} \\ &\leq \sum_{k=2}^{d} |a_k| \left(\frac{N}{4a}\right)^{k} \alpha^{N+1} \\ &\leq a \max\left\{\frac{N^2}{(4a)^2}, \frac{N^d}{(4a)^d}\right\} \alpha^{N+1} \\ &\leq \max\left\{\frac{N^2}{4}, \frac{N^d}{4^{d-1}}\right\} |a_0|\alpha^N \leq N^d |a_0| \alpha^N, \end{aligned}$$

using Lemma 5.3 and $a \geq 1$. □

Intuitively, the conclusion of Theorem 5.5 says that x_N is a Cauchy sequence with an explicit modulus of convergence whose limit is a root of h of bounded modulus.

6 Valued fields

Theorem 5.5 shows that $VTC^0 + IMUL$ can compute roots of polynomials of a special form, however it would still be rather difficult to extend it to a full-blown root-finding algorithm. We will instead give a model-theoretic argument using well-known properties of valued fields to bridge the gap between Theorem 5.5 and approximation of roots of general polynomials.

In order to prove $VTC^0 + IMUL \vdash IOpen$, it suffices to show that every model of $VTC^0 + IMUL$ is a model of *IOpen*. First, since $VTC^0 + IMUL \vdash DIV$, we can reformulate Theorem 2.1 in terms of fields.

Lemma 6.1 *Let D be a DOR, and F its fraction field. The following are equivalent.*

(i) *$D \vDash IOpen$.*

(ii) *$D \vDash DIV$, and F is a dense subfield of a RCF R.* □

The condition that F is dense in R means that elements of R can be well approximated in F, i.e., R cannot be too large, while the condition that R is real-closed (or at least contains the real closure $\tilde{F}^{\mathrm{real}}$) means that R cannot be too small, so these two conditions work against each other. One canonical choice of R is the smallest RCF extending F, i.e., $\tilde{F}^{\mathrm{real}}$. We obtain that a DOR $D \vDash DIV$ is a model of *IOpen* iff F is dense in $\tilde{F}^{\mathrm{real}}$. However, it will be useful for us to consider another choice: it turns out that there exists the largest ordered field extension $\hat{F} \supseteq F$ in which F is dense, and a DOR $D \vDash DIV$ is a model of *IOpen* iff $\hat{F}$ is real-closed.

The existence of $\hat{F}$ was shown by Scott [163]. One way to prove it is by generalization of the construction of $\mathbb{R}$ using Dedekind cuts. Consider pairs $\langle A, B\rangle$, where $F = A \cup B$, B has no smallest element, and

$$\inf\{b - a : a \in A, b \in B\} = 0.$$

One can show that the collection of all such cuts can be given the structure of an ordered field in a natural way, and it has the property needed of $\hat{F}$. However, we will use a different construction of $\hat{F}$ which may look more complicated on first sight, but has the advantage of allowing us to employ tools from the theory of valuations to explore its properties (such as being real-closed). It can be thought of as generalizing the construction of $\mathbb{R}$ by means of Cauchy sequences.

We refer the reader to [76] for the theory of valued fields, however we will review our notation and some basic facts below to make sure we are on the same page.

A *valuation* on a field K is a surjective mapping $v\colon K \twoheadrightarrow \Gamma \cup \{\infty\}$, where $\langle \Gamma, +, \le \rangle$ is a totally ordered abelian group (called the *value group*), and v satisfies

(i) $v(a) = \infty$ only if $a = 0$,

(ii) $v(ab) = v(a) + v(b)$,

(iii) $v(a + b) = \min\{v(a), v(b)\}$,

where we put $\infty + \gamma = \gamma + \infty = \infty$ and $\gamma \le \infty$ for every $\gamma \in \Gamma$. (Elements with large valuation should be thought of as being small; the order is upside down for historical reasons.) Valuations $v\colon K \to \Gamma \cup \{\infty\}$, $v'\colon K \to \Gamma' \cup \{\infty\}$ are *equivalent* if there is an ordered group isomorphism $f\colon \Gamma \to \Gamma'$ such that $v' = f \circ v$.

The *valuation ring* of v is

$$O = \{a \in K : v(a) \ge 0\},$$

with its unique *maximal ideal* being

$$I = \{a \in K : v(a) > 0\}.$$

The quotient field $k = O/I$ is called the *residue field.* If $a \in O$, we will denote its image under the natural projection $O \to k$ as $\overline{a}$.

More abstractly, a valuation ring for a field K is a subring $O \subseteq K$ such that $a \in O$ or $a^{-1} \in O$ for every $a \in K^\times$. Any such ring corresponds to a valuation: we take $\Gamma = K^\times/O^\times$ ordered by $aO^\times \le bO^\times$ iff $b \in aO$, and define v as the natural projection $v(a) = aO^\times$. A valuation is determined uniquely up to equivalence by its valuation ring; thus, either of the structures $\langle K, v\rangle$ and $\langle K, O\rangle$ can be called a *valued field.* A valued field $\langle K', v'\rangle$ is an *extension* of $\langle K, v\rangle$ if K is a subfield of K', and $v \subseteq v'$. (In terms of valuation rings, the latter means $O = O' \cap K$.) A valuation (or valuation ring or valued field) is *nontrivial* if $\Gamma \ne \{0\}$, or equivalently, if $O \ne K$.

A valuation $v\colon K \to \Gamma \cup \{\infty\}$ induces a *topology* on K with basic open sets

$$B(a, \gamma) = \{b \in K : v(b - a) > \gamma\}, \qquad a \in K, \gamma \in \Gamma.$$

(Note that $B(a, \gamma) = B(a', \gamma)$ for any $a' \in B(a, \gamma)$.) This makes K a topological field, and as with any topological group, it also makes K a uniform space (with a fundamental system of entourages of the form $\{\langle a, b\rangle \in K^2 : v(a - b) > \gamma\}$ for $\gamma \in \Gamma$). Consequently, we have the notions of Cauchy nets, completeness, and completion; for the particular case of valued fields, they can be stated as follows. A *Cauchy sequence* in K is $\{a_\gamma : \gamma \in \Gamma\} \subseteq K$ such that $v(a_\gamma - a_\delta) > \min\{\gamma, \delta\}$ for every $\gamma, \delta \in \Gamma$. (Alternatively, it would be enough if Cauchy sequences were indexed over a cofinal subset of Γ.) Such a sequence converges to $a \in K$ if $v(a - a_\gamma) > \gamma$ for every $\gamma \in \Gamma$. The valued field $\langle K, v\rangle$ is *complete* if every Cauchy sequence in K converges. A *completion* of $\langle K, v\rangle$ is an extension $\langle \hat{K}, \hat{v}\rangle$ of $\langle K, v\rangle$ which is a complete valued field such that K is (topologically) dense in $\hat{K}$. (The last condition implies that $\hat{K}$ is an *immediate* extension of K, i.e., the natural embeddings $\Gamma \subseteq \hat{\Gamma}$ and $k \subseteq \hat{k}$ are isomorphisms.)

Theorem 6.2 ([76, Thm.2.4.3]) *Every valued field $\langle K, v\rangle$ has a completion, which is unique up to a unique valued field isomorphism identical on K.* □

Now we turn to the interaction of valuation and order [76, §2.2.2]. Let $\langle K, O\rangle$ be a valued field. If $\le$ is an order on K (i.e., $\langle K, \le\rangle$ is an ordered field) such that O is *convex* (i.e., $a \le b \le c$ and $a, c \in O$ implies $b \in O$), then an order is induced on the residue field k by $\overline{a} \le \overline{b} \Leftrightarrow a \le b$. Conversely, any order on k is induced from an order $\le$ on K making O convex in this way. If Γ is 2-divisible, such a $\le$ is unique, and can be defined explicitly by

$$a > 0 \iff \exists b \in K\,(ab^2 \in O^\times \wedge \overline{ab^2} > 0).$$

In general, the structure of all such orders $\le$ is described by the Baer–Krull theorem [76, Thm. 2.2.5]. Notice also that every convex subring of an ordered field is a valuation ring.

Lemma 6.3 *If $\langle K, \le\rangle$ is an ordered field, and O a nontrivial convex subring of K, then the valuation topology on K coincides with the interval topology. In particular, a subset $X \subseteq K$ is topologically dense iff it is order-theoretically dense.*

Proof: The convexity of O implies that every $B(a, \gamma)$ is also convex. If $c \in (a, b)$, and $\gamma \ge v(c - a), v(c - b)$, then $c \in B(c, \gamma) \subseteq (a, b)$. On the other hand, if $c \in B(a, \gamma)$, pick $e > 0$ with $v(e) > \gamma$ (which exists as the valuation is nontrivial). Then $c \in (c - e, c + e) \subseteq B(a, \gamma)$. □

For any ordered field $\langle K, \le\rangle$, the set of its bounded elements

$$O = \{a \in K : \exists q \in \mathbb{Q}^+\,(-q \le a \le q)\}$$

is a convex valuation ring for K with the set of infinitesimal elements

$$I = \{a \in K : \forall q \in \mathbb{Q}^+\,(-q \le a \le q)\}$$

being its maximal ideal. The corresponding valuation is the *natural valuation* induced by $\le$. The residue field is an archimedean ordered field, and as such it can be uniquely identified with a subfield $k \subseteq \mathbb{R}$. Here is the promised construction of the largest dense extension of an ordered field.

Lemma 6.4 *Let $\langle K, \le\rangle$ be a nonarchimedean ordered field, v its natural valuation, and $\langle \hat{K}, \hat{v}\rangle$ its completion. There is a unique order on $\hat{K}$ extending $\le$ that makes $\hat{O}$ convex. Its natural valuation is $\hat{v}$, and it satisfies:*

(i) *$\hat{K}$ is an ordered field extension of K such that K is dense in $\hat{K}$.*

(ii) *If K' is any ordered field extension of K in which K is dense, there is a unique ordered field embedding of K' in $\hat{K}$ identical on K.*

Proof: Since $\hat{K}$ is an immediate extension of K, for every $a \in \hat{K}^\times$ there exists an $a_0 \in K^\times$ such that $aa_0^{-1} \in 1 + \hat{I}$, or equivalently, $\hat{v}(a - a_0) > \hat{v}(a) = v(a_0)$. Any order $\hat{\le}$ on $\hat{K}$ extending $\le$ such that $\hat{O}$ is convex (which implies $1 + \hat{I} \subseteq \hat{K}^+$) must satisfy

$$a \mathrel{\hat{>}} 0 \iff a_0 > 0, \tag{18}$$

which specifies it uniquely. On the other hand, we claim that (18) defines an order on $\hat{K}$. First, the definition is independent of the choice of a_0: if $a_1 \in K^\times$ is such that $aa_1^{-1} \in 1 + \hat{I}$, then $a_0a_1^{-1} \in 1+I$ is positive, whence a_0 and a_1 have the same sign. Clearly, exactly one of a and $-a$ is positive for any $a \in \hat{K}^\times$. Let $a, b \in \hat{K}^\times$, $a, b \mathrel{\hat{>}} 0$. Since $(ab)(a_0b_0)^{-1} \in 1 + \hat{I}$, we have $ab \mathrel{\hat{>}} 0$. Also, $v(a_0 + b_0) = \min\{v(a_0), v(b_0)\}$ as they have the same sign, thus

$$\hat{v}\big((a + b) - (a_0 + b_0)\big) \ge \min\{\hat{v}(a - a_0), \hat{v}(b - b_0)\} > \min\{v(a_0), v(b_0)\} = v(a_0 + b_0).$$

This means we can take $a_0 + b_0$ for $(a + b)_0$, showing that $a + b \mathrel{\hat{>}} 0$.

If $a \mathrel{\hat{<}} b \mathrel{\hat{<}} c$, $a, c \in \hat{O}$, we may assume $(c - a)_0 = (c - b)_0 + (b - a)_0$ by the argument above, hence $(c - b)_0 + (b - a)_0 \in O$. Since $(c - b)_0, (b - a)_0 > 0$, this implies $(b - a)_0 \in O$, hence $b - a \in \hat{O}$, and $b \in \hat{O}$. Thus, $\hat{O}$ is convex under $\hat{\le}$.

Since $\langle K, \le\rangle$ is nonarchimedean, the valuations v and $\hat{v}$ are nontrivial. Thus, K is an order-theoretically dense subfield of $\hat{K}$ by Lemma 6.3, which shows (i). Also, in view of the convexity of $\hat{O}$, this implies that O is dense in $\hat{O}$, hence

$$\hat{O} = \{a \in \hat{K} : \exists q \in \mathbb{Q}^+ \, (-q \mathrel{\hat{\le}} a \mathrel{\hat{\le}} q)\},$$

i.e., $\hat{v}$ is the natural valuation of $\langle \hat{K}, \hat{\le}\rangle$.

(ii): Let v' be the natural valuation on K', and $\langle \hat{K}', \hat{v}'\rangle$ its completion. By Lemma 6.3, $\langle K, v\rangle$ is topologically dense in its complete extension $\langle \hat{K}', \hat{v}'\rangle$, hence there is an isomorphism of $\langle \hat{K}', \hat{v}'\rangle$ and $\langle \hat{K}, \hat{v}\rangle$ identical on K by Theorem 6.2. It restricts to an embedding $f\colon \langle K', v'\rangle \to \langle \hat{K}, \hat{v}\rangle$. For any $a \in K'$, we can see from (18) that $f(a) \mathrel{\hat{>}} 0$ implies $a_0 > 0$ for some $a_0 \in K^\times$ such that $aa_0^{-1} \in 1 + I'$, whence $a >' 0$. Thus, f is order-preserving. The uniqueness of f follows from the density of K in $\hat{K}$. □

(If K is archimedean, its natural valuation is trivial, hence the induced topology is discrete, and $\hat{K} = K$. However, the largest ordered field extension of K where K is dense is $\mathbb{R}$.)

We will rely on the following important characterization of real-closed fields in terms of valuations [76, Thm. 4.3.7].

Theorem 6.5 *Let $\langle K, \le\rangle$ be an ordered field, and O a convex valuation ring of K. The following are equivalent.*

(i) *K is real-closed.*

(ii) *Γ is divisible, k is real-closed, and O is henselian.* □

There are many equivalent definitions of henselian valuation rings or valued fields (cf. [76, Thm. 4.1.3]). It will be most convenient for our purposes to adopt the following one: a valuation ring O or a valued field $\langle K, O\rangle$ is *henselian* iff every polynomial $h(x) = \sum_{i=0}^{d} a_i x^i \in O[x]$ such that $a_0 \in I$ and $a_1 = 1$ has a root in I.

The basic intuition behind Theorem 6.5 is that in order to find a root a of a polynomial in K, we use the divisibility of Γ to get a ballpark estimate of a, we refine it to an approximation up to an infinitesimal relative error using the real-closedness of k, and then use the henselian property to compute a. Complications arise from interference with other roots of the polynomial.

It is well known that the completion of a henselian valued field is henselian. In fact, we have the following simple criterion, where we define a valued field $\langle K, O\rangle$ to be *almost henselian* if for every polynomial h as above, and every $\gamma \in \Gamma$, there is $a \in I$ such that $v(h(a)) > \gamma$. (Equivalently, $\langle K, O\rangle$ is almost henselian iff the quotient ring O/P is henselian for every nonzero prime ideal $P \subseteq O$ [176].)

Lemma 6.6 *The completion $\langle \hat{K}, \hat{v}\rangle$ is henselian iff $\langle K, v\rangle$ is almost henselian.*

Proof: First, we observe that if $h = \sum_{i=0}^{d} a_i x^i \in O[x]$ has $a_1 = 1$, then

$$v(h(b) - h(c)) = v(b - c) \tag{19}$$

for any $b, c \in I$. Indeed, if $b \neq c$, we have

$$\frac{h(b) - h(c)}{b - c} = a_1 + \sum_{i=2}^{d} a_i(b^{i-1} + b^{i-2}c + \cdots + c^{i-1}) \in 1 + I \subseteq O^{\times}.$$

Left to right: assume that $h = \sum_{i=0}^{d} a_i x^i \in O[x]$, $a_1 = 1$, $a_0 \in I$, and $\gamma \in \Gamma$. Without loss of generality, $\gamma \geq 0$. Since $\hat{K}$ is henselian, there is $\hat{a} \in \hat{I}$ such that $h(\hat{a}) = 0$. By the density of K in $\hat{K}$, we can find $a \in K$ such that $\hat{v}(a - \hat{a}) > \gamma$. Then $a \in I$, and $v(h(a)) > \gamma$ by (19).

Right to left: let $h = \sum_{i=0}^{d} a_i x^i \in \hat{O}[x]$ with $a_1 = 1$ and $a_0 \in \hat{I}$. For any $\gamma \in \Gamma$, $\gamma \geq 0$, we choose $a_{i,\gamma} \in K$ such that $\hat{v}(a_i - a_{i,\gamma}) > \gamma$, and put $h_\gamma = \sum_i a_{i,\gamma} x^i$. Then $h_\gamma \in O[x]$, $a_{0,\gamma} \in I$, and we could have picked $a_{1,\gamma} = 1$, hence by assumption, there is $b_\gamma \in I$ such that $v(h_\gamma(b_\gamma)) > \gamma$. By the choice of h_γ, this implies $\hat{v}(h(b_\gamma)) > \gamma$. Moreover, $v(b_\gamma - b_\delta) = \hat{v}(h(b_\gamma) - h(b_\delta)) > \min\{\gamma, \delta\}$ by (19), hence $\{b_\gamma : \gamma \geq 0\}$ is a Cauchy sequence. Since $\hat{K}$ is complete, there is $b \in \hat{K}$ such that $\hat{v}(b - b_\gamma) > \gamma$ for every γ. Then $b \in \hat{I}$. Since $\hat{v}(h(b) - h(b_\gamma)) > \gamma$ by (19), we have $\hat{v}(h(b)) > \gamma$ for every $\gamma \in \Gamma$, i.e., $h(b) = 0$. □

Putting all the things together, we obtain the following characterization of open induction. We note that the fact that the completion of a real-closed field is real-closed was shown by Scott [163].

Lemma 6.7 *Let D be a nonstandard DOR such that $D \vDash DIV$, F its fraction field endowed with its natural valuation, and $\hat{F}$ its completion. The following are equivalent.*

(i) *$D \vDash IOpen$.*

(ii) *$\hat{F}$ is real-closed.*

(iii) *F is almost henselian, its value group is divisible, and its residue field is real-closed.*

Proof:

(ii) and (iii) are equivalent by Theorem 6.5 and Lemma 6.6, using the fact that $\hat{F}$ is an immediate extension of F.

(ii) $\to$ (i) follows from Lemma 6.1 as F is dense in $\hat{F}$. Conversely, assume that F is a dense subfield of a RCF R. By Theorem 6.5, R is henselian, its value group is divisible, and its residue field is a RCF. The completion $\hat{R}$ is also henselian by Lemma 6.6, and it has the same Γ and k as R, hence it is a RCF by Theorem 6.5. However, the density of F in $\hat{R}$ implies $\hat{F} \simeq \hat{R}$ by Lemma 6.4, hence $\hat{F}$ is a RCF. □

We remark that we could have used any nontrivial convex subring in place of the natural valuation in Lemma 6.4 (any two such valuations determine the same uniform structure by Lemma 6.3, which means that their completions are the same qua topological fields, and one checks easily that they also carry the same order). Likewise, Lemma 6.7 continues to hold when F is endowed with any nontrivial valuation with a convex valuation ring; this may make a difference for verification of condition (iii). Notice that such valuation rings correspond to proper cuts (in the models-of-arithmetic sense) on D closed under multiplication.

We can now prove the main result of this paper.

Theorem 6.8 *$VTC^0 + IMUL$ proves $IOpen$ on binary integers.*

Proof: Let $M \vDash VTC^0 + IMUL$, and D be its ring of binary integers, we need to show that $D \vDash IOpen$. We may assume without loss of generality that M, and therefore D, is ω-saturated. Since $VTC^0 + IMUL \vdash DIV$, it suffices to check the conditions of Lemma 6.7 (iii).

As we have mentioned above, the residue field k of any ordered field under its natural valuation is a subfield of $\mathbb{R}$. The ω-saturation of D implies that every Dedekind cut on $\mathbb{Q}$ is realized by an element of F, hence in fact $k = \mathbb{R}$, which is a real-closed field.

Every element of the value group Γ is the difference of valuations of two (positive) elements of D. Let thus $a \in D^+$, and $k \in \mathbb{Z}^+$. Put $n = \|a\| - 1$, which is a unary integer of M such that $2^n \le a < 2^{n+1}$. Put $m = \lfloor n/k \rfloor$ and $b = 2^m$. Then $b^k \le a < 2^k b^k$, hence $kv(b) = v(a)$. This shows that Γ is divisible.

Let $\gamma \in \Gamma$, and $h(x) = \sum_{i \le d} a_i x^i \in F[x]$ be such that $v(a_i) \ge 0$, $v(a_0) > 0$, and $a_1 = 1$. Then $a = \max\{1, \sum_{i=2}^{d} |a_i|\}$ is bounded by a standard integer, whereas a_0 is infinitesimal, thus $\alpha = 4a|a_0|$ is also infinitesimal. Let N be a nonstandard unary integer of M such that $v(2^{-N}) > \gamma$, and let x_N be as in Theorem 5.5. Then using a crude estimate,

$$|h(x_N)| \le N^d |a_0| \alpha^N \le 2^N 4^{-N} = 2^{-N},$$

which means that $v(h(x_N)) > \gamma$. Moreover, $|x_N| \le |a_0|/(1-\alpha)$ is infinitesimal. Thus, F is almost henselian. □

As explained in Section 3, Theorem 6.8 implies that for any constant d, $VTC^0 + IMUL$ can formalize a TC^0 algorithm for approximation of roots of degree d rational polynomials. The reader might find it disappointing that we have shown its existence nonconstructively using the abstract nonsense from this section, so let us give at least a rough idea how this algorithm may actually look like; it is somewhat different from the one in Chapter VII.

Clearly, one ingredient is Theorem 5.5, which gives an explicit description of a TC^0 algorithm for approximation of roots of polynomials of a special form (small constant coefficient and large linear coefficient). The remaining part is a reduction of general root approximation to this special case, and this happens essentially in Theorem 6.5. This theorem has a proof with a fairly algorithmic flavour using Newton polygons (cf. [21, §2.6], where a similar argument is given in the special case of real Puiseux series). The Newton polygon of a polynomial $f(x)=\sum_{i=0}^{d}a_ix^i\in K[x]$ is the lower convex hull of the set of points $\{e_i=\langle i,v(a_i)\rangle : i=0,\dots,d\}\subseteq\mathbb{Q}\times\Gamma$.

The basic idea is as follows. Take an edge of the Newton polygon with endpoints e_{i_0},e_{i_1}. The slope of the edge is in Γ due to its divisibility, hence we can replace $f(x)$ by a suitable polynomial of the form $af(bx)$ to ensure $v(a_{i_0})=v(a_{i_1})=0$. Then $f\in O[x]$, its image $\overline{f}\in k[x]$ has degree i_1, and the least exponent of its nonzero coefficient is i_0. If we find a nonzero root $\overline{a}\in k^\times$ of $\overline{f}$ of multiplicity m using the real-closedness of k, the Newton polygon of the shifted polynomial $f(x+a)$ will have an edge whose endpoints satisfy $i_0'<i_1'\le m\le i_1-i_0$, since m is the least exponent with a nonzero coefficient in $\overline{f}(x+\overline{a})$. This is strictly shorter than the original edge unless $\overline{f}$ is a constant multiple of $x^{i_0}(x-\overline{a})^{i_1-i_0}$, which case has to be handled separately. If we set up the argument properly, we can reduce f by such linear substitutions in at most d steps into a polynomial whose Newton polygon has e_0,e_1 for vertices, and then we can apply the henselian property to find its root in K.

One can imagine that a proper TC^0 algorithm working over $\mathbb{Q}$ instead of a nonarchimedean field can be obtained along similar lines by replacing "infinitesimal" with a suitable notion of "small enough" (e.g., employing an approximation of $-\log|a|$ as a measure of magnitude in place of $v(a)$). However, the details are bound to be quite unsightly due to complications arising from the loss of the ultrametric inequality of v.

7 Application to Buss's theories

While VTC^0+IMUL does not stand much chance of proving induction for interesting classes of formulas with quantifiers in the language of ordered rings, we will show in this section that we can do better in the richer language $L_B=\langle 0,1,+,\cdot,\le,\#,|x|,\lfloor x/2\rfloor\rangle$ of Buss's one-sorted theories of bounded arithmetic—VTC^0+IMUL proves the $RSUV$-translation of T^0_2, and even minimization for sharply bounded formulas (Σ^b_0-MIN). The main tool is a description of Σ^b_0-definable sets discovered by Mantzivis [127], whose variants were also given in [33, 113]: in essence, a Σ^b_0-definable subset of $[0,2^n)$ can be written as a union of $n^{O(1)}$ intervals on each residue class modulo 2^c, where c is a standard constant. As we will see, this property can be formalized in VTC^0+IMUL using the provability of $IOpen$ for the base case of polynomial inequalities, and as a consequence, our theory proves minimization and induction for Σ^b_0 formulas. (We stress that as in the case of $IOpen$, these are minimization and induction over binary numbers. Despite the same name, the schemata denoted as IND and MIN in the two-sorted framework only correspond to $LIND$ and minimization over lengths in Buss's language, respectively.) We will present the messier part of the argument as a normal form for Σ^b_0 formulas over a weak base theory, in the hope that this will make the result more reusable.

We will assume the reader is familiar with definitions of Buss's theories (see e.g. [37, 116]), in particular, with *BASIC*. Recall that a formula is sharply bounded if all its quantifiers are of the form $\exists x \le |t|$ or $\forall x \le |t|$. We reserve Σ^b_0 for the class of sharply bounded formulas of L_B, whereas sharply bounded formulas in an extended language $L_B \cup L'$ will be denoted $\Sigma^b_0(L')$. Let $BASIC^+$ denote the extension of Buss's *BASIC* by the axioms

$$x(yz) = (xy)z, \tag{20}$$

$$y \le x \to \exists z\,(y + z = x), \tag{21}$$

$$u \le |x| \to \exists y\,(|y| = u), \tag{22}$$

$$z < x \mathbin{\#} y \to |z| \le |x||y|, \tag{23}$$

$$|x| \le x. \tag{24}$$

(The quantifiers in (21), (22) could be bounded by x, if desired.) On top of *BASIC*, axioms (20) and (21) imply the theory of nonnegative parts of discretely ordered rings, hence we can imagine the universe is extended with negative numbers in the usual fashion. In particular, we can work with integer polynomials. We introduce two extra functions by

$$\begin{aligned} x \mathbin{\dot-} y = z &\iff y + z = x \lor (x < y \land z = 0), \\ 2^{\min\{u,|x|\}} = z &\iff z \mathbin{\#} 1 = 2z \land \big((u \le |x| \land |z| = u + 1) \lor (u > |x| \land |z| = |x| + 1)\big). \end{aligned}$$

$BASIC^+$ proves that $\dot-$ and $2^{\min\{u,|x|\}}$ are well-defined total functions. Notice that $BASIC^+$ is universally axiomatizable in a language with $\dot-$ and $2^{\min\{u,|x|\}}$. We will write 2^u for $2^{\min\{u,|x|\}}$ when a self-evident value of x such that $u \le |x|$ can be inferred from the context (e.g., when u is a sharply bounded quantified variable).

If p is a polynomial with nonnegative integer coefficients, one can construct easily a term t such that $BASIC^+ \vdash p(|x_1|, \dots, |x_k|) \le |t(\vec{x})|$. Conversely, one can check that $BASIC^+$ proves $|xy| \le |x| + |y|$; together with other axioms, this implies that for every term t (even using $\dot-$ and $2^{\min\{u,|x|\}}$) there is a polynomial p such that $BASIC^+ \vdash |t(\vec{x})| \le p(|\vec{x}|)$.

Lemma 7.1 *Let $\varphi(x_1, \dots, x_k)$ be a $\Sigma^b_0(\dot-, 2^{\min\{u,|x|\}})$ formula. Then $BASIC^+$ proves $\varphi(\vec{x})$ equivalent to a formula of the form*

$$\bigvee_{\sigma_1,\dots,\sigma_k < 2^c} \Big(\bigwedge_{i=1}^{k} (x_i \equiv \sigma_i \pmod{2^c}) \\ \land Q_1 u_1 \le p(|\vec{x}|) \cdots Q_l u_l \le p(|\vec{x}|)\, f_{\vec\sigma}(x_1, \dots, x_k, u_1, \dots, u_l, 2^{u_1}, \dots, 2^{u_l}) \ge 0 \Big),$$

where c is a constant, $Q_1, \dots, Q_l \in \{\exists, \forall\}$, p is a nonnegative integer polynomial, $x_i \equiv \sigma_i$ $\pmod{2^c}$ stands for $x_i = \sigma_i + 2^c\lfloor \cdots \lfloor\lfloor x_i/\underbrace{2\rfloor/2\rfloor \cdots /2}_{c}\rfloor$, and $f_{\vec\sigma}$ is an integer polynomial.

Proof: Using the remark before the lemma, we can find a nonnegative integer polynomial p such that $p(|\vec{x}|)$ bounds the values of $|t|$ for every subterm $t(\vec{x}, \vec{u})$ occurring in φ and all possible values of the quantified variables $\vec{u}$. Then we can rewrite φ in the form

$$Q_1 u_1 \leq p(|\vec{x}|) \cdots Q_l u_l \leq p(|\vec{x}|)\, \psi(\vec{x}, \vec{u}),$$

where ψ is open. The next step is elimination of unwanted function symbols. Let $|t|$ be a subterm of ψ, and write $\psi(\vec{x}, \vec{u}) = \psi'(\vec{x}, \vec{u}, |t|)$. Then $\psi(\vec{x}, \vec{u})$ is equivalent to

$$\exists u \leq p(|\vec{x}|)\, (|t| = u \wedge \psi'(\vec{x}, \vec{u}, u)).$$

Using the axioms of $BASIC^+$ and the definition of 2^u, this is equivalent to

$$\exists u \leq p(|\vec{x}|)\, (\lfloor 2^u/2 \rfloor \leq t < 2^u \wedge \psi'(\vec{x}, \vec{u}, u)).$$

Likewise,

$$\psi(\vec{x}, \vec{u}, t \# s) \leftrightarrow \exists u, v, w \leq p(|\vec{x}|)\, (u = |t| \wedge v = |s| \wedge w = uv \wedge \psi(\vec{x}, \vec{u}, 2^w)),$$
$$\psi(\vec{x}, \vec{u}, 2^{\min\{t,|s|\}}) \leftrightarrow \exists u, v \leq p(|\vec{x}|)\, (u = |s| \wedge v = \min\{t, u\} \wedge \psi(\vec{x}, \vec{u}, 2^v)),$$

where we further eliminate $|t|$ and $|s|$ as above, and $\min\{t, u\}$ in an obvious way. Applying successively these reductions, we can eventually write φ as

$$Q_1 u_1 \leq p(|\vec{x}|) \cdots Q_l u_l \leq p(|\vec{x}|)\, \psi(\vec{x}, \vec{u}, 2^{\vec{u}}), \tag{25}$$

where ψ is an open formula in the language $\langle 0, 1, +, \cdot, \dot{-}, \lfloor x/2 \rfloor, \leq \rangle$.

Claim 7.1.1 *Let $t(\vec{x})$ be a $\langle 0, 1, +, \cdot, \dot{-}, \lfloor x/2 \rfloor \rangle$-term such that the nesting depth of $\lfloor x/2 \rfloor$ in t is c, and the number of occurrences of $\dot{-}$ is r. For every $\vec{\sigma} < 2^c$, there are integer polynomials $g_1, \ldots, g_r$ and $\{f_{\vec{\alpha}} : \alpha_1, \ldots, \alpha_r \in \{0, 1\}\}$ such that $BASIC^+$ proves*

$$\bigwedge_{i=1}^{r} (g_i(\vec{x}) \geq 0)^{\alpha_i} \to t(2^c \vec{x} + \vec{\sigma}) = f_{\vec{\alpha}}(\vec{x}), \tag{26}$$

where $\varphi^1 = \varphi$, $\varphi^0 = \neg\varphi$.

Proof: By induction on the complexity of t. For example, assume (26) holds for t, and consider the term $\lfloor t/2 \rfloor$. Let $\vec{\tau} < 2$, and assume that $f_{\vec{\alpha}}(\vec{\tau}) \equiv \rho \pmod 2$, $\rho \in \{0, 1\}$. Notice that all coefficients of $f_{\vec{\alpha}}(2\vec{x} + \vec{\tau}) - \rho$ are even, so $h_{\vec{\alpha}}(\vec{x}) = \frac{1}{2}(f_{\vec{\alpha}}(2\vec{x}) - \rho)$ is again an integer polynomial, and $BASIC^+$ proves

$$\bigwedge_{i=1}^{r} (g_i(2\vec{x} + \vec{\tau}) \geq 0)^{\alpha_i} \to t(2^{c+1}\vec{x} + (2^c \vec{\tau} + \vec{\sigma})) = \left\lfloor \frac{f_{\vec{\alpha}}(2\vec{x})}{2} \right\rfloor = \left\lfloor \frac{2h_{\vec{\alpha}}(\vec{x}) + \rho}{2} \right\rfloor = h_{\vec{\alpha}}(\vec{x}).$$

□ (Claim 7.1.1)

Claim 7.1.2 *Every open formula $\psi(\vec{x})$ in the language $\langle 0,1,+,\cdot,\dot{-},\lfloor x/2\rfloor,\leq\rangle$ is equivalent to a formula of the form*

$$\bigvee_{\vec{\sigma}<2^c}\Big(\bigwedge_i (x_i \equiv \sigma_i \pmod{2^c}) \wedge \psi_{\vec{\sigma}}(\vec{x})\Big)$$

over $BASIC^+$, where each $\psi_{\vec{\sigma}}$ is a Boolean combination of integer polynomial inequalities.

Proof: Using Claim 7.1.1 and $BASIC^+$-provable uniqueness of the representation $x = 2^c y + \sigma$, $\sigma < 2^c$, we obtain an equivalent of ψ in almost the right form except that $\psi_{\vec{\sigma}}$ is a Boolean combination of inequalities of the form

$$f(2^{-c}(\vec{x}-\vec{\sigma})) \geq 0,$$

where f is an integer polynomial. If $d = \deg(f)$, $g(\vec{x}) = 2^{cd} f(2^{-c}(\vec{x}-\vec{\sigma}))$ is an integer polynomial, and the inequality above is equivalent to $g(\vec{x}) \geq 0$. □ (Claim 7.1.2)

Let us apply Claim 7.1.2 to the formula $\psi(\vec{x},\vec{u},2^{\vec{u}})$ in (25). Since $BASIC^+$ knows that $2^0 = 1$ and $2^{u+1} = 2\cdot 2^u$, we can replace $2^{u_i} \equiv \sigma \pmod{2^c}$ with $2^{u_i} = \sigma \vee (u_i \geq c \wedge \sigma = 0)$. Moreover, $u_i \equiv \sigma \pmod{2^c}$ can be written as $\exists v \leq p(|\vec{x}|)\,(u_i = 2^c v + \sigma)$, and $x_i \equiv \sigma \pmod{2^c}$ can be moved outside the quantifier prefix. Thus, $\varphi(\vec{x})$ is equivalent to

$$\bigvee_{\vec{\sigma}<2^c}\Big(\bigwedge_{i=1}^{k}(x_i \equiv \sigma_i \pmod{2^c}) \wedge Q_1 u_1 \leq p(|\vec{x}|)\cdots Q_l u_l \leq p(|\vec{x}|)\,\psi_{\vec{\sigma}}(\vec{x},\vec{u},2^{\vec{u}})\Big),$$

where $\psi_{\vec{\sigma}}$ is a Boolean combination of integer polynomial inequalities. We can reduce $\psi_{\vec{\sigma}}$ to a single inequality using

$$\neg(f \geq 0) \leftrightarrow -f-1 \geq 0,$$
$$f \geq 0 \wedge g \geq 0 \leftrightarrow \forall v \leq p(|\vec{x}|)\,(vf + (1-v)^2 g \geq 0),$$

assuming $p(|\vec{x}|) \geq 1$. □

Lemma 7.2 *$VTC^0 + IMUL$ proves the following for every constant d: if $\{f_u : u < n\}$ is a sequence of integer polynomials of degree d (each given by a $(d+1)$-tuple of binary integer coefficients), and $a > d$ is a binary integer, there exists a double sequence $w = \{w_{u,i} : u < n, i \leq d+1\}$ such that $0 = w_{u,0} < w_{u,1} < \cdots < w_{u,d+1} = a$ and $f_u(x)$ has a constant sign on each interval $[w_{u,i}, w_{u,i+1})$, that is,*

$$(27) \qquad \forall u < n\,\forall x \bigwedge_{i\leq d}\big(w_{u,i} \leq x < w_{u,i+1} \to (f_u(x) \geq 0 \leftrightarrow f_u(w_{u,i}) \geq 0)\big).$$

Proof: Using $IOpen$, $\{x < a : f(x) \geq 0\}$ is a union of at most d intervals for every polynomial f of degree at most d, i.e., $VTC^0 + IMUL$ proves

$$\forall f\,\forall a > d\,\exists 0 = x_0 < \cdots < x_{d+1} = a\,\forall x \bigwedge_{i\leq d}\big(x_i \leq x < x_{i+1} \to (f(x) \geq 0 \leftrightarrow f(x_i) \geq 0)\big).$$

Now we would like to invoke Σ^B_1-AC^R to find a sequence w satisfying (27), but we cannot directly do that as the conclusion is only Π^B_1.

Let $M \vDash VTC^0 + IMUL$, and R be its real closure. Quantifier elimination for RCF furnishes an open formula ϑ in L_{OR} such that $M \vDash \vartheta(x, y, a_0, \dots, a_d)$ iff $f(x) = \sum_{i \le d} a_i x^i$ has no roots in the interval $(x, y]_R$. By replacing f with $2f + 1$ if necessary, we may assume f has no integral roots. Let $\alpha_1 < \cdots < \alpha_c$, $c \le d$, be the list of all roots of f in $(0, a]_R$, and let $x_0, \dots, x_{d+1} \in M$ be the sequence of integers $0, \lceil \alpha_1 \rceil, \dots, \lceil \alpha_c \rceil, a$ (which exist due to *IOpen*) with duplicates removed, and dummy elements added if necessary to make it the proper length. Then f has no roots in the intervals $(x_i, x_{i+1} - 1]_R$. This means we can prove in $VTC^0 + IMUL$ the statement

$$\forall f\, \forall a > d\, \exists 0 = x_0 < x_1 < \cdots < x_{d+1} = a \bigwedge_{i \le d} \vartheta(x_i, x_{i+1} - 1, f),$$

which has the right complexity, hence we can use Σ^B_1-AC^R to derive the existence of a sequence w such that

$$\forall u < n \Big(w_{u,0} = 0 \wedge w_{u,d+1} = a \wedge \bigwedge_{i \le d} \big(w_{u,i} < w_{u,i+1} \wedge \vartheta(w_{u,i}, w_{u,i+1} - 1, f_u) \big) \Big).$$

This implies (27). □

Theorem 7.3 *The RSUV-translation of $BASIC^+ + \Sigma^b_0(\div, 2^{\min\{u,|x|\}})$-MIN, and a fortiori of T^0_2, is provable in $VTC^0 + IMUL$.*

Proof: Work in $VTC^0 + IMUL$. It is straightforward but tedious to verify the $BASIC^+$ axioms. Let $\varphi(x)$ be (the translation of) a $\Sigma^b_0(\div, 2^{\min\{u,|x|\}})$ formula (possibly with other parameters), and a a binary number such that $\varphi(a)$, we have to find the least such number. Since it is enough to do this separately on each residue class modulo 2^c, we can assume using Theorem 7.3 that $\varphi(x)$ is equivalent to

$$Q_1 u_1 \le n \cdots Q_l u_l \le n\, f(x, \vec{u}, 2^{\vec{u}}) \ge 0$$

for $x < a$, where n is a unary number, and f is a polynomial with binary integer coefficients. By Lemma 7.2, there is a sequence w such that

$$w_{\vec{u},i} \le x < w_{\vec{u},i+1} \to (f(x, \vec{u}, 2^{\vec{u}}) \ge 0 \leftrightarrow f(w_{\vec{u},i}, \vec{u}, 2^{\vec{u}}) \ge 0)$$

for all $x < a$, $\vec{u} \le n$, and $i \le d$. As VTC^0 proves that every sequence of integers can be sorted, there is an increasing sequence $\{w'_j : j < m\}$ whose elements include every $w_{\vec{u},i}$. Consequently, the truth value of $\varphi(x)$ is constant on each interval $[w'_j, w'_{j+1})$, and the minimal $x < a$ satisfying $\varphi(x)$, if any, is w'_{j_0}, where

$$j_0 = \min\{j < m : \varphi(w'_j)\}.$$

The latter exists by $\Sigma^B_0(L_{\overline{VTC^0 + IMUL}})$-*COMP*. □

We remark that the proof used nothing particularly special about division by 2, except that *BASIC* conveniently includes the $\lfloor x/2 \rfloor$ function symbol and the relevant axioms. We could allow more general instances of division as long as the values of all denominators encountered when evaluating a Σ^b_0 formula on $[0, a]$ have a common multiple which is a length (unary number); in particular, Theorem 7.3 (along with an appropriate version of Lemma 7.1) holds for Σ^b_0 formulas in a language further expanded by function symbols for $\lfloor x/2^{||y||}\rfloor$ and $\lfloor x/\max\{1,||y||\}\rfloor$.

We formulated Theorem 7.3 for $\mathit{VTC}^0+\mathit{IMUL}$ as we have been working with this two-sorted theory throughout the main part of the paper, however here it is perhaps more natural to state the result directly in terms of one-sorted arithmetic to avoid needless RSUV translation. A theory Δ^b_1-CR corresponding to TC^0 was defined by Johannsen and Pollett [110], and shown RSUV-isomorphic to VTC^0 by Nguyen and Cook [133]. Recall also Johannsen's theory $C^0_2[\mathit{div}]$ from Section 3.

Corollary 7.4 *The theories Δ^b_1-$\mathit{CR}+\mathit{IMUL}$ and $C^0_2[\mathit{div}]$ prove $\Sigma^b_0(\dot-,2^{\min\{u,|x|\}})$-$\mathit{MIN}$ and therefore T^0_2.* □

To put Theorem 7.3 in context, there has been a series of results to the effect that various subsystems of bounded arithmetic axiomatized by sharply bounded schemata are pathologically weak. Takeuti [169] has shown that $S^0_2=\Sigma^b_0$-PIND does not prove the totality of the predecessor function, and Johannsen [107] extended his method to show that S^0_2 in a language including $\dot-$, $\lfloor x/2^y\rfloor$, and bit counting does not prove the totality of division by three (or even of the AC^0 function $\lfloor 2^{|x|}/3\rfloor$). Boughattas and Kołodziejczyk [33] have shown that $T^0_2=\Sigma^b_0$-IND does not prove that nontrivial divisors of powers of two are even, and by Kołodziejczyk [113], it does not even prove $3\nmid 2^{|x|}$. These results also apply to certain mild extensions of T^0_2, nevertheless no unconditional independence result is known for Σ^b_0-MIN, or its subtheory $T^0_2+S^0_2$.

What makes such separations possible is a lack of computational power. It is no coincidence that there are no result of this kind for two-sorted Zambella-style theories, where already the base theory V^0 proves the totality of all AC^0-functions: we can show $V^0(p)\nsubseteq V^0(q)$ for primes $p\neq q$ using the known lower bounds for $\mathrm{AC}^0[p]$, but we have no independence results for stronger theories without complexity assumptions such as $\mathrm{AC}^0[6]\neq\mathrm{PH}$. This is directly related to the expressive power of sharply bounded formulas: while Σ^B_0 formulas can define all AC^0 predicates, the ostensibly quite similar Σ^b_0 formulas (that even involve the TC^0-complete multiplication function) have structural properties that preclude this, as witnessed by Mantzivis's result. Indeed, the pathological behaviour of T^0_2 disappears if we slightly extend its language: as proved in [94], $T^0_2(\lfloor x/2^y\rfloor)=\mathit{PV}_1$, and this can be easily extended to show $\Sigma^b_0(\lfloor x/2^y\rfloor)$-$\mathit{MIN}=T^1_2$.

Theorem 7.3 formally implies only conditional separations: in particular, $\mathit{PV}_1\nsubseteq\Sigma^b_0$-$\mathit{MIN}$ unless $\mathrm{P}=\mathrm{TC}^0$, and Σ^b_0-$\mathit{MIN}\subsetneq T^1_2$ unless $\mathrm{PH}=\mathrm{BH}\subseteq\mathrm{TC}^0/\mathrm{poly}$ and $\mathrm{PLS}=\mathrm{FTC}^0$ (provably in Σ^b_0-MIN). However, heuristically it gives us more. If Σ^b_0-MIN were a "computationally reasonable" theory, we would expect it to coincide with T^1_2 due to its shape, or at the very least to correspond to a class closer to PLS than TC^0. Thus, Theorem 7.3 indicates that it might be a pathologically weak theory in some way, and therefore amenable to unconditional independence results by means of a direct combinatorial construction of models in the spirit of [33, 113].

8 Conclusion

The weakest theory of bounded arithmetic in the setup of [182, 68] that can talk about elementary arithmetic operations on binary integers is VTC^0. We have shown that its strengthening $VTC^0 + IMUL$ proves that these operations are fairly well behaved in that they satisfy open induction. Despite that the theory $VTC^0 + IMUL$ corresponds to the complexity class TC^0 similarly to VTC^0, it is still an interesting problem what properties of integer arithmetic operations are provable in plain VTC^0. In view of Theorem 6.8 and Corollary 3.6, we have:

Corollary 8.1 *VTC^0 proves IOpen if and only if it proves DIV.* □

Question 8.2 *Does VTC^0 prove DIV? In particular, does it prove the soundness of the division algorithm by Hesse, Allender, and Barrington [90]?*

While the analysis of the algorithm in [90] generally relies on quite elementary tools, its formalization in VTC^0 suffers from "chicken-and-egg" problems. For instance, the proof of Lemma 6.1, whose goal is to devise an algorithm for finding small powers in groups, assumes there is a well-behaved powering function, and uses its various properties to establish that its value is correctly computed by the algorithm. This is no good if we need the very algorithm to construct the powering function in the first place. Similarly, integer division is employed throughout Section 4. It is not clear whether one can circumvent these circular dependencies in VTC^0. On the other hand, the requisite operations such as division are available in $VTC^0 + IMUL$, which makes it plausible that $VTC^0 + IMUL$ *can* formalize the arguments.

We remark that it is not difficult to do division by *standard* integers in VTC^0. This means VTC^0 knows that binary integers form a *$\mathbb{Z}$-ring*, and in particular, they satisfy all universal consequences of *IOpen* by a result of Wilkie [178]. (*IOpen* itself is a $\forall\exists$-axiomatized theory, and likewise, DIV is a $\forall\exists$ sentence.)

As explained in Section 7, our main result implies that $VTC^0 + IMUL$ (or better, the corresponding one-sorted theory $\Delta^b_1\text{-}CR + IMUL$) proves minimization for Σ^b_0 formulas in Buss's language, which suggests that the theory axiomatized by $\Sigma^b_0\text{-}MIN$ is rather weak. Consequently, it might be feasible to unconditionally separate this theory from stronger fragments of S_2, nevertheless our argument gives no clue how to do that.

Problem 8.3 *Prove that Σ^b_0-MIN is strictly weaker than T^1_2 without complexity-theoretic assumptions.*

Acknowledgements

I am indebted to Leszek Kołodziejczyk for useful suggestions on an earlier version of this paper.

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 339691. The Institute of Mathematics of the Czech Academy of Sciences is supported by RVO: 67985840.

Bibliography

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, PRIMES *is in* P, Annals of Mathematics 160 (2004), no. 2, pp. 781–793.

[2] Lars V. Ahlfors, *Complex analysis: An introduction to the theory of analytic functions of one complex variable*, McGraw–Hill, New York, 1979.

[3] Miklós Ajtai, *The complexity of the Pigeonhole Principle*, Combinatorica 14 (1994), pp. 417–433.

[4] Miklós Ajtai, János Komlós, and Endre Szemerédi, *Sorting in $c \log n$ parallel steps*, Combinatorica 3 (1983), no. 1, pp. 1–19.

[5] ———, *An $O(n \log n)$ sorting network*, in: Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 1983, pp. 1–9.

[6] Eric Allender, *The permanent requires large uniform threshold circuits*, Chicago Journal of Theoretical Computer Science 1999, article no. 7.

[7] ———, *Arithmetic circuits and counting complexity classes*, in Krajíček [119], pp. 33–72.

[8] Eric Allender and David A. Mix Barrington, *Uniform circuits for division: consequences and problems*, Technical Report TR00-065, Electronic Colloquium on Computational Complexity, 2000.

[9] Noga Alon and Ravi B. Boppana, *The monotone circuit complexity of Boolean functions*, Combinatorica 7 (1987), no. 1, pp. 1–22.

[10] Nesmith C. Ankeny, *The least quadratic non residue*, Annals of Mathematics 55 (1952), no. 1, pp. 65–72.

[11] Toshiyasu Arai, *A bounded arithmetic AID for Frege systems*, Annals of Pure and Applied Logic 103 (2000), pp. 155–199.

[12] Albert Atserias, *The complexity of resource-bounded propositional proofs*, Ph.D. thesis, Universitat Politècnica de Catalunya, Barcelona, 2002.

[13] Albert Atserias, Nicola Galesi, and Ricard Gavaldà, *Monotone proofs of the Pigeon Hole Principle*, Mathematical Logic Quarterly 47 (2001), no. 4, pp. 461–474.

[14] Albert Atserias, Nicola Galesi, and Pavel Pudlák, *Monotone simulations of non-monotone proofs*, Journal of Computer and System Sciences 65 (2002), no. 4, pp. 626–638.

[15] Albert Atserias and Neil Thapen, *The ordering principle in a fragment of approximate counting*, ACM Transactions on Computational Logic 15 (2014), no. 4, article no. 29.

[16] László Babai, *Trading group theory for randomness*, in: Proceedings of the 17th Annual ACM Symposium on Theory of Computing, 1985, pp. 421–429.

[17] László Babai and Shlomo Moran, *Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes*, Journal of Computer and System Sciences 36 (1988), no. 2, pp. 254–276.

[18] Eric Bach, *Explicit bounds for primality testing and related problems*, Mathematics of Computation 55 (1990), no. 191, pp. 355–380.

[19] David A. Mix Barrington, Neil Immerman, and Howard Straubing, *On uniformity within NC^1*, Journal of Computer and System Sciences 41 (1990), no. 3, pp. 274–306.

[20] David A. Mix Barrington, Peter Kadau, Klaus-Jörn Lange, and Pierre McKenzie, *On the complexity of some problems on groups input as multiplication tables*, Journal of Computer and System Sciences 63 (2001), no. 2, pp. 186–200.

[21] Saugata Basu, Richard Pollack, and Marie-Françoise Roy, *Algorithms in real algebraic geometry*, Springer, 2006.

[22] Kenneth E. Batcher, *Sorting networks and their applications*, in: Proceedings of the AFIPS Spring Joint Computer Conference, vol. 32, 1968, pp. 307–314.

[23] Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi, *The relative complexity of* NP *search problems*, Journal of Computer and System Sciences 57 (1998), no. 1, pp. 3–19.

[24] Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, Pavel Pudlák, and Alan Woods, *Exponential lower bounds for the pigeonhole principle*, in: Proceedings of the 24th Annual ACM Symposium on Theory of Computing, 1992, pp. 200–220.

[25] Paul W. Beame, Stephen A. Cook, and H. James Hoover, *Log depth circuits for division and related problems*, SIAM Journal on Computing 15 (1986), no. 4, pp. 994–1003.

[26] Michael Ben-Or, Ephraim Feig, Dexter Kozen, and Prasoon Tiwari, *A fast parallel algorithm for determining all roots of a polynomial with real roots*, SIAM Journal on Computing 17 (1988), no. 6, pp. 1081–1092.

[27] Charles H. Bennett and John Gill, *Relative to a random oracle A,* $\mathbf{P}^A \neq \mathbf{NP}^A \neq \text{co-}\mathbf{NP}^A$ *with probability* 1, SIAM Journal on Computing 10 (1981), no. 1, pp. 96–113.

[28] Alessandro Berarducci and Paola D'Aquino, Δ_0*-complexity of the relation* $y = \prod_{i\leq n} F(i)$, Annals of Pure and Applied Logic 75 (1995), no. 1–2, pp. 49–56.

[29] Alessandro Berarducci and Benedetto Intrigila, *Combinatorial principles in elementary number theory*, Annals of Pure and Applied Logic 55 (1991), no. 1, pp. 35–50.

[30] Olaf Beyersdorff and Sebastian Müller, *A tight Karp–Lipton collapse result in bounded arithmetic*, ACM Transactions on Computational Logic 11 (2010), no. 4, article no. 22.

[31] Errett Bishop and Douglas S. Bridges, *Constructive analysis*, Grundlehren der mathematischen Wissenschaften vol. 279, Springer, 1985.

[32] Andreas Blass, Ioanna M. Dimitriou, and Benedikt Löwe, *Inaccessible cardinals without the axiom of choice*, Fundamenta Mathematicae 194 (2007), pp. 179–189.

[33] Sedki Boughattas and Leszek A. Kołodziejczyk, *The strength of sharply bounded induction requires MSP*, Annals of Pure and Applied Logic 161 (2010), no. 4, pp. 504–510.

[34] Daniel P. Bovet, Pierluigi Crescenzi, and Riccardo Silvestri, *A uniform approach to define complexity classes*, Theoretical Computer Science 104 (1992), no. 2, pp. 263–283.

[35] Joshua Buresh-Oppenheim, *On the* **TFNP** *complexity of factoring*, unpublished note, `http://www.cs.toronto.edu/~bureshop/factor.pdf`, 2006.

[36] ———, private communication.

[37] Samuel R. Buss, *Bounded arithmetic*, Bibliopolis, Naples, 1986, revision of 1985 Princeton University Ph.D. thesis.

[38] ———, *The Boolean formula value problem is in ALOGTIME*, in: Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, pp. 123–131.

[39] ———, *Relating the bounded arithmetic and polynomial time hierarchies*, Annals of Pure and Applied Logic 75 (1995), no. 1–2, pp. 67–77.

[40] ———, *First-order proof theory of arithmetic*, in *Handbook of Proof Theory* [41], pp. 79–147.

[41] Samuel R. Buss (ed.), *Handbook of proof theory*, Studies in Logic and the Foundations of Mathematics vol. 137, Elsevier, Amsterdam, 1998.

[42] ———, *An introduction to proof theory*, in *Handbook of Proof Theory* [41], pp. 1–78.

[43] Samuel R. Buss and Alan S. Johnson, *Propositional proofs and reductions between* NP *search problems*, Annals of Pure and Applied Logic 163 (2012), no. 9, pp. 1163–1182.

[44] Samuel R. Buss, Valentine Kabanets, Antonina Kolokolova, and Michal Koucký, *Expander construction in* VNC^1, Annals of Pure and Applied Logic 171 (2020), no. 7, article no. 102796, 40 pp.

[45] Samuel R. Buss, Leszek A. Kołodziejczyk, and Neil Thapen, *Fragments of approximate counting*, Journal of Symbolic Logic 79 (2014), no. 2, pp. 496–525.

[46] Samuel R. Buss, Leszek A. Kołodziejczyk, and Konrad Zdanowski, *Collapsing modular counting in bounded arithmetic and constant depth propositional proofs*, Transactions of the American Mathematical Society 367 (2015), no. 11, pp. 7517–7563.

[47] Jin-Yi Cai, $\mathrm{S}_2^p \subseteq \mathrm{ZPP}^{\mathrm{NP}}$, Journal of Computer and System Sciences 73 (2007), no. 1, pp. 25–35.

[48] Ran Canetti, *More on BPP and the polynomial-time hierarchy*, Information Processing Letters 57 (1996), no. 5, pp. 237–241.

[49] J. Lawrence Carter and Mark N. Wegman, *Universal classes of hash functions*, Journal of Computer and System Sciences 18 (1979), no. 2, pp. 143–154.

[50] Ashok K. Chandra, Larry Stockmeyer, and Uzi Vishkin, *Constant depth reducibility*, SIAM Journal on Computing 13 (1984), no. 2, pp. 423–439.

[51] Xi Chen and Xiaotie Deng, *Settling the complexity of two-player Nash equilibrium*, in: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, 2006, pp. 261–271.

[52] Andrew Y. Chiu, George I. Davida, and Bruce E. Litow, *Division in logspace-uniform* NC^1, RAIRO – Theoretical Informatics and Applications 35 (2001), no. 3, pp. 259–275.

[53] Peter Clote, *ALOGTIME and a conjecture of S.A. Cook*, Annals of Mathematics and Artificial Intelligence 6 (1992), no. 1–3, pp. 57–106.

[54] Peter Clote and Jan Krajíček (eds.), *Arithmetic, proof theory, and computational complexity*, Oxford Logic Guides vol. 23, Oxford University Press, 1993.

[55] ———, *Open problems*, in *Arithmetic, proof theory, and computational complexity* [54], pp. 1–19.

[56] Peter Clote and Gaisi Takeuti, *Bounded arithmetic for NC, ALogTIME, L and NL*, Annals of Pure and Applied Logic 56 (1992), pp. 73–117.

[57] Alan Cobham, *The intrinsic computational difficulty of functions*, in: Proceedings of the 2nd International Congress of Logic, Methodology and Philosophy of Science (Y. Bar-Hillel, ed.), North-Holland, 1965, pp. 24–30.

[58] Louis Comtet, *Advanced combinatorics: The art of finite and infinite expansions*, D. Reidel Publishing Company, Dordrecht, 1974.

[59] John B. Conway, *Functions of one complex variable*, Springer, New York, 1978.

[60] ———, *Functions of one complex variable II*, Springer, New York, 1995.

[61] Stephen Cook and Tsuyoshi Morioka, *Quantified propositional calculus and a second-order theory for* $\mathbf{NC}^1$, Archive for Mathematical Logic 44 (2005), no. 6, pp. 711–749.

[62] Stephen A. Cook, *The complexity of theorem proving procedures*, in: Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, 1971, pp. 151–158.

[63] __________, *Feasibly constructive proofs and the propositional calculus*, in: Proceedings of the 7th Annual ACM Symposium on Theory of Computing, 1975, pp. 83–97.

[64] __________, *Relating the provable collapse of* $\mathbf{P}$ *to* $\mathbf{NC}^1$ *and the power of logical theories*, in: Proof Complexity and Feasible Arithmetics (P. Beame and S. R. Buss, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science vol. 39, American Mathematical Society, 1998, pp. 73–92.

[65] __________, *Theories for complexity classes and their propositional translations*, in Krajíček [119], pp. 175–227.

[66] Stephen A. Cook and Antonina Kolokolova, *A second-order theory for NL*, in: Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004, pp. 398–407.

[67] Stephen A. Cook and Jan Krajíček, *Consequences of the provability of* $\mathbf{NP} \subseteq \mathbf{P/poly}$, Journal of Symbolic Logic 72 (2007), no. 4, pp. 1353–1371.

[68] Stephen A. Cook and Phuong Nguyen, *Logical foundations of proof complexity*, Perspectives in Logic, Cambridge University Press, New York, 2010.

[69] Stephen A. Cook and Robert A. Reckhow, *The relative efficiency of propositional proof systems*, Journal of Symbolic Logic 44 (1979), no. 1, pp. 36–50.

[70] Charalambos Cornaros, *On Grzegorczyk induction*, Annals of Pure and Applied Logic 74 (1995), no. 1, pp. 1–21.

[71] Charalambos Cornaros and Costas Dimitracopoulos, *The prime number theorem and fragments of PA*, Archive for Mathematical Logic 33 (1994), no. 4, pp. 265–281.

[72] Paola D'Aquino and Angus Macintyre, *Non-standard finite fields over* $I\Delta_0 + \Omega_1$, Israel Journal of Mathematics 117 (2000), pp. 311–333.

[73] __________, *Quadratic forms in models of* $I\Delta_0 + \Omega_1$*. I*, Annals of Pure and Applied Logic 148 (2007), pp. 31–48.

[74] __________, *Quadratic forms in models of* $I\Delta_0 + \Omega_1$*, Part II: Local equivalence*, Annals of Pure and Applied Logic 162 (2011), no. 6, pp. 447–456.

[75] Émile Durand, *Solutions numériques des équations algébriques. Tome I: Équations du type* $F(x) = 0$*: Racines d'un polynôme*, Masson, Paris, 1960 (in French).

[76] Antonio J. Engler and Alexander Prestel, *Valued fields*, Springer Monographs in Mathematics, Springer, 2005.

[77] Paul Erdős, *On a problem in graph theory*, Mathematical Gazette 47 (1963), no. 361, pp. 220–223.

[78] Harvey M. Friedman, *Some systems of second order arithmetic and their use*, in: Proceedings of the International Congress of Mathematicians (Vancouver 1974), vol. 1, Canadian Mathematical Congress, 1975, pp. 235–242.

[79] Gerhard Gentzen, *Die Widerspruchsfreiheit der reinen Zahlentheorie*, Mathematische Annalen 112 (1936), pp. 493–565.

[80] John Gill, *Computational complexity of probabilistic Turing machines*, SIAM Journal on Computing 6 (1977), no. 4, pp. 675–695.

[81] Kurt Gödel, *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I*, Monatshefte für Mathematik und Physik 38 (1931), pp. 173–198.

[82] Oded Goldreich and Avi Wigderson, *Improved derandomization of BPP using a hitting set generator*, in: Proceedings of RANDOM-APPROX '99 (D. S. Hochbaum, K. Jansen, J. D. P. Rolim, and A. Sinclair, eds.), Lecture Notes in Computer Science vol. 1671, Springer, 1999, pp. 131–137.

[83] Shafi Goldwasser and Michael Sipser, *Private coins versus public coins in interactive proof systems*, in: Randomness and Computation (S. Micali, ed.), Advances in Computing Research vol. 5, JAI Press, Greenwich, 1989, pp. 73–90.

[84] Gene H. Golub and Charles F. Van Loan, *Matrix computations*, third ed., Johns Hopkins University Press, Baltimore, 1996.

[85] Ronald L. Graham and Joel H. Spencer, *A constructive solution to a tournament problem*, Canadian Mathematical Bulletin 14 (1971), no. 1, pp. 45–48.

[86] Petr Hájek and Pavel Pudlák, *Metamathematics of first-order arithmetic*, Perspectives in Mathematical Logic, Springer, 1993, second edition 1998.

[87] András Hajnal, Wolfgang Maass, Pavel Pudlák, Márió Szegedy, and György Turán, *Threshold circuits of bounded depth*, Journal of Computer and System Sciences 46 (1993), no. 2, pp. 129–154.

[88] Juris Hartmanis and Lane A. Hemachandra, *Complexity classes without machines: on complete languages for UP*, Theoretical Computer Science 58 (1988), pp. 129–142.

[89] Roger Heath-Brown, *Fermat's two squares theorem*, Invariant 11 (1984), pp. 3–5, Oxford University Invariant Society.

[90] William Hesse, Eric Allender, and David A. Mix Barrington, *Uniform constant-depth threshold circuits for division and iterated multiplication*, Journal of Computer and System Sciences 65 (2002), no. 4, pp. 695–716.

[91] Neil Immerman, *Expressibility and parallel complexity*, SIAM Journal on Computing 18 (1989), no. 3, pp. 625–638.

[92] Michael A. Jenkins and Joseph F. Traub, *A three-stage variable-shift iteration for polynomial zeros and its relation to generalized Rayleigh iteration*, Numerische Mathematik 14 (1970), no. 3, pp. 252–263.

[93] Emil Jeřábek, *Dual weak pigeonhole principle, Boolean complexity, and derandomization*, Annals of Pure and Applied Logic 129 (2004), pp. 1–37.

[94] __________, *The strength of sharply bounded induction*, Mathematical Logic Quarterly 52 (2006), no. 6, pp. 613–624.

[95] __________, *On independence of variants of the weak pigeonhole principle*, Journal of Logic and Computation 17 (2007), no. 3, pp. 587–604.

[96] __________, *Approximate counting in bounded arithmetic*, Journal of Symbolic Logic 72 (2007), no. 3, pp. 959–993.

[97] __________, *Substitution Frege and extended Frege proof systems in non-classical logics*, Annals of Pure and Applied Logic 159 (2009), no. 1–2, pp. 1–48.

[98] __________, *Approximate counting by hashing in bounded arithmetic*, Journal of Symbolic Logic 74 (2009), no. 3, pp. 829–860.

[99] __________, *Abelian groups and quadratic residues in weak arithmetic*, Mathematical Logic Quarterly 56 (2010), no. 3, pp. 262–278.

[100] __________, *On theories of bounded arithmetic for NC^1*, Annals of Pure and Applied Logic 162 (2011), no. 4, pp. 322–340.

[101] __________, *A sorting network in bounded arithmetic*, Annals of Pure and Applied Logic 162 (2011), no. 4, pp. 341–355.

[102] __________, *Proofs with monotone cuts*, Mathematical Logic Quarterly 58 (2012), no. 3, pp. 177–187.

[103] __________, *Root finding with threshold circuits*, Theoretical Computer Science 462 (2012), pp. 59–69.

[104] __________, *Open induction in a bounded arithmetic for* TC^0, Archive for Mathematical Logic 54 (2015), no. 3–4, pp. 359–394.

[105] __________, *Integer factoring and modular square roots*, Journal of Computer and System Sciences 82 (2016), no. 2, pp. 380–394.

[106] __________, *Iterated multiplication in VTC^0*, arXiv:2011.03095 [cs.LO], 2020, `https://arxiv.org/abs/2011.03095`. Accepted to Archive for Mathematical Logic.

[107] Jan Johannsen, *On the weakness of sharply bounded polynomial induction*, in: Computational Logic and Proof Theory, Proceedings of Kurt Gödel Colloquium ’93 (G. Gottlob, A. Leitsch, and D. Mundici, eds.), Lecture Notes in Computer Science vol. 713, Springer, 1993, pp. 223–230.

[108] ―――――, *Weak bounded arithmetic, the Diffie-Hellman problem, and Constable's class K*, in: Proceedings of the 14th Annual IEEE Symposium on Logic in Computer Science, 1999, pp. 268–274.

[109] Jan Johannsen and Chris Pollett, *On proofs about threshold circuits and counting hierarchies (extended abstract)*, in: Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science, 1998, pp. 444–452.

[110] ―――――, *On the Δ^b_1-bit-comprehension rule*, in: Logic Colloquium '98: Proceedings of the 1998 ASL European Summer Meeting held in Prague, Czech Republic (S. R. Buss, P. Hájek, and P. Pudlák, eds.), ASL, 2000, pp. 262–280.

[111] Valentine Kabanets, Charles Rackoff, and Stephen A. Cook, *Efficiently approximable real-valued functions*, Technical Report TR00-034, Electronic Colloquium on Computational Complexity, 2000.

[112] Immo O. Kerner, *Ein Gesamtschrittverfahren zur Berechnung der Nullstellen von Polynomen*, Numerische Mathematik 8 (1966), no. 3, pp. 290–294 (in German).

[113] Leszek A. Kołodziejczyk, *Independence results for variants of sharply bounded induction*, Annals of Pure and Applied Logic 162 (2011), no. 12, pp. 981–990.

[114] Michal Koucký, Valentine Kabanets, and Antonina Kolokolova, *Expanders made easy: The combinatorial analysis of an expander construction*, unpublished manuscript, 2007.

[115] Jan Krajíček, *No counter-example interpretation and interactive computation*, in: Logic From Computer Science, Proceedings of a Workshop held November 13–17, 1989 in Berkeley (Y. N. Moschovakis, ed.), Mathematical Sciences Research Institute Publications vol. 21, Springer, 1992, pp. 287–293.

[116] ―――――, *Bounded arithmetic, propositional logic, and complexity theory*, Encyclopedia of Mathematics and Its Applications vol. 60, Cambridge University Press, 1995.

[117] ―――――, *Uniform families of polynomial equations over a finite field and structures admitting an Euler characteristic of definable sets*, Proceedings of the London Mathematical Society 81 (2000), no. 3, pp. 257–284.

[118] ―――――, *Approximate Euler characteristic, dimension, and weak pigeonhole principles*, Journal of Symbolic Logic 69 (2004), no. 1, pp. 201–214.

[119] Jan Krajíček (ed.), *Complexity of computations and proofs*, Quaderni di Matematica vol. 13, Seconda Universita di Napoli, 2004.

[120] Jan Krajíček and Pavel Pudlák, *Some consequences of cryptographical conjectures for S^1_2 and EF*, Information and Computation 140 (1998), no. 1, pp. 82–94.

[121] Jan Krajíček, Pavel Pudlák, and Gaisi Takeuti, *Bounded arithmetic and the polynomial hierarchy*, Annals of Pure and Applied Logic 52 (1991), no. 1–2, pp. 143–153.

[122] Clemens Lautemann, *BPP and the polynomial hierarchy*, Information Processing Letters 17 (1983), no. 4, pp. 215–217.

[123] Franz Lemmermeyer, *Reciprocity laws: From Euler to Eisenstein*, Springer Monographs in Mathematics, Springer, 2000, see also `https://www.mathi.uni-heidelberg.de/~flemmermeyer/qrg_proofs.html`.

[124] Saunders Mac Lane and Garrett Birkhoff, *Algebra*, third ed., American Mathematical Society, Providence, 1999.

[125] Alexis Maciel, Toniann Pitassi, and Alan R. Woods, *A new proof of the weak pigeonhole principle*, Journal of Computer and System Sciences 64 (2002), no. 4, pp. 843–872.

[126] Alexis Maciel and Denis Thérien, *Efficient threshold circuits for power series*, Information and Computation 152 (1999), no. 1, pp. 62–73.

[127] Spyro-Giorgio Mantzivis, *Circuits in bounded arithmetic part I*, Annals of Mathematics and Artificial Intelligence 6 (1991), no. 1–3, pp. 127–156.

[128] Morris Marden, *The geometry of the zeros of a polynomial in a complex variable*, Mathematical Surveys vol. 3, American Mathematical Society, New York, 1949.

[129] Moritz Müller and Ján Pich, *Feasibly constructive proofs of succinct weak circuit lower bounds*, Annals of Pure and Applied Logic 171 (2020), no. 2, article no. 102735, 45 pp.

[130] C. Andrew Neff, *Specified precision polynomial root isolation is in NC*, Journal of Computer and System Sciences 48 (1994), no. 3, pp. 429–463.

[131] C. Andrew Neff and John H. Reif, *An efficient algorithm for the complex roots problem*, Journal of Complexity 12 (1996), no. 2, pp. 81–115.

[132] Phuong Nguyen, *Bounded reverse mathematics*, Ph.D. thesis, University of Toronto, 2008.

[133] Phuong Nguyen and Stephen A. Cook, *Theories for TC^0 and other small complexity classes*, Logical Methods in Computer Science 2 (2006), no. 1, article no. 3, 39 pp.

[134] Noam Nisan and Amnon Ta-Shma, *Symmetric Logspace is closed under complement*, in: Proceedings of the 27th Annual ACM Symposium on Theory of Computing, 1995, pp. 140–146.

[135] Noam Nisan and Avi Wigderson, *Hardness vs. randomness*, Journal of Computer and System Sciences 49 (1994), no. 2, pp. 149–167.

[136] Kerry E. Ojakian, *Combinatorics in bounded arithmetic*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, 2004.

[137] Victor Y. Pan, *Fast and efficient algorithms for sequential and parallel evaluation of polynomial zeros and of matrix polynomials*, in: Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science, 1985, pp. 522–531.

[138] ———, *Optimal and nearly optimal algorithms for approximating polynomial zeros*, Computers & Mathematics with Applications 31 (1996), no. 12, pp. 97–138.

[139] ———, *Solving a polynomial equation: Some history and recent progress*, SIAM Review 39 (1997), no. 2, pp. 187–220.

[140] Christos H. Papadimitriou, *On the complexity of the parity argument and other inefficient proofs of existence*, Journal of Computer and System Sciences 48 (1994), no. 3, pp. 498–532.

[141] Ian Parberry and Georg Schnitger, *Parallel computation with threshold functions*, Journal of Computer and System Sciences 36 (1988), no. 3, pp. 278–302.

[142] Rohit Parikh, *Existence and feasibility in arithmetic*, Journal of Symbolic Logic 36 (1971), no. 3, pp. 494–508.

[143] Jeff B. Paris and Alex J. Wilkie, Δ_0 *sets and induction*, in: Open days in model theory and set theory. Proceedings of a conference held in September 1981 at Jadwisin, near Warsaw, Poland (W. Guzicki, W. Marek, A. Pelc, and C. Rauszer, eds.), Leeds University Press, 1983, pp. 237–248.

[144] ———, *Counting problems in bounded arithmetic*, in: Methods in Mathematical Logic (C. A. Di Prisco, ed.), Lecture Notes in Mathematics vol. 1130, Springer, 1985, pp. 317–340.

[145] ———, *Counting* Δ_0 *sets*, Fundamenta Mathematicae 127 (1987), no. 1, pp. 67–76.

[146] Jeff B. Paris, Alex J. Wilkie, and Alan R. Woods, *Provability of the pigeonhole principle and the existence of infinitely many primes*, Journal of Symbolic Logic 53 (1988), no. 4, pp. 1235–1244.

[147] Charles Parsons, *On a number theoretic choice schema and its relation to induction*, in: Intuitionism and Proof Theory: Proceedings of the Summer Conference at Buffalo N.Y. 1968 (A. Kino, J. Myhill, and R. E. Vesley, eds.), Studies in Logic and the Foundations of Mathematics vol. 60, North-Holland, 1970, pp. 459–473.

[148] Michael S. Paterson, *Improved sorting networks with* $O(\log N)$ *depth*, Algorithmica 5 (1990), no. 1, pp. 75–92.

[149] Ioseph [Giuseppe] Peano, *Arithmetices principia, nova methodo exposita*, Fratelli Bocca, Torino, 1889 (in Latin).

[150] Ján Pich, *Logical strength of complexity theory and a formalization of the PCP theorem in bounded arithmetic*, Logical Methods in Computer Science 11 (2015), no. 2, article no. 8, 38 pp.

[151] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, *Numerical recipes: The art of scientific computing*, third ed., Cambridge University Press, 2007.

[152] Pavel Pudlák, *Ramsey's theorem in bounded arithmetic*, in: Proceedings of Computer Science Logic '90 (E. Börger, H. K. Büning, M. M. Richter, and W. Schönfeld, eds.), Lecture Notes in Computer Science vol. 533, Springer, 1991, pp. 308–317.

[153] ———, *On the complexity of propositional calculus*, in: Sets and proofs: Invited papers from Logic Colloquium '97 (S. B. Cooper and J. K. Truss, eds.), Cambridge University Press, 1999, pp. 197–218.

[154] Michael O. Rabin, *Digitalized signatures and public-key functions as intractable as factorization*, Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[155] Александр А. Разборов, *Формулы ограниченной глубины в базисе* $\{\&, \oplus\}$ *и некоторые комбинаторные задачи*, in: Сложность вычислений и прикладная математическая логика (С. И. Адян, ed.), Вопросы кибернетики vol. 134, VINITI, Moscow, 1988, pp. 149–166 (in Russian).

[156] Alexander A. Razborov, *Lower bounds on the monotone complexity of some Boolean functions*, Mathematics of the USSR, Doklady 31 (1985), pp. 354–357.

[157] Alexander A. Razborov and Steven Rudich, *Natural proofs*, Journal of Computer and System Sciences 55 (1997), no. 1, pp. 24–35.

[158] John H. Reif, *Logarithmic depth circuits for algebraic functions*, SIAM Journal on Computing 15 (1986), no. 1, pp. 231–242.

[159] John H. Reif and Stephen R. Tate, *On threshold circuits and polynomial computation*, SIAM Journal on Computing 21 (1992), no. 5, pp. 896–908.

[160] Søren M. Riis, *Making infinite structures finite in models of second order bounded arithmetic*, in Clote and Krajíček [54], pp. 289–319.

[161] Alexander Russell and Ravi Sundaram, *Symmetric alternation captures* **BPP**, Computational Complexity 7 (1998), no. 2, pp. 152–162.

[162] Walter L. Ruzzo, *On uniform circuit complexity*, Journal of Computer and System Sciences 22 (1981), no. 3, pp. 365–383.

[163] Dana Scott, *On completing ordered fields*, in: Applications of Model Theory to Algebra, Analysis, and Probability (W. A. J. Luxemburg, ed.), Holt, Rinehart and Winston, New York, 1969, pp. 274–278.

[164] John C. Shepherdson, *A nonstandard model for a free variable fragment of number theory*, Bulletin de l'Académie Polonaise des Sciences, Série des Sciences Mathématiques, Astronomiques et Physiques 12 (1964), no. 2, pp. 79–86.

[165] Stephen G. Simpson, *Subsystems of second order arithmetic*, Perspectives in Mathematical Logic, Springer, Berlin, 1999.

[166] Michael Sipser, *A complexity theoretic approach to randomness*, in: Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 1983, pp. 330–335.

[167] Robert M. Solovay, *Re: AC and strongly inaccessible cardinals*, Foundations of Mathematics mailing list, March 2008, `https://cs.nyu.edu/pipermail/fom/2008-March/012783.html`.

[168] Esther Szekeres and George Szekeres, *On a problem of Schütte and Erdös*, Mathematical Gazette 49 (1965), no. 369, pp. 290–293.

[169] Gaisi Takeuti, *Sharply bounded arithmetic and the function* $a \dot{-} 1$, in: Logic and Computation, Proceedings of a Workshop held at Carnegie Mellon University, June 30–July 2, 1987 (W. Sieg, ed.), Contemporary Mathematics vol. 106, American Mathematical Society, 1990, pp. 281–288.

[170] ――――――, *RSUV isomorphisms*, in Clote and Krajíček [54], pp. 364–386.

[171] Neil Thapen, *A model-theoretic characterization of the weak pigeonhole principle*, Annals of Pure and Applied Logic 118 (2002), no. 1–2, pp. 175–195.

[172] ――――――, *The weak pigeonhole principle in models of bounded arithmetic*, Ph.D. thesis, Oxford University, 2002.

[173] ――――――, *Structures interpretable in models of bounded arithmetic*, Annals of Pure and Applied Logic 136 (2005), no. 3, pp. 247–266.

[174] Seinosuke Toda, *On the computational power of PP and* $\oplus P$, in: Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science, 1989, pp. 514–519.

[175] Leslie G. Valiant, *Short monotone formulae for the majority function*, Journal of Algorithms 5 (1984), no. 3, pp. 363–366.

[176] Peter Vámos, *Decomposition problems for modules over valuation domains*, Journal of the London Mathematical Society s2-41 (1990), no. 1, pp. 10–26.

[177] Christopher S. Wallace, *A suggestion for a fast multiplier*, IEEE Transactions on Electronic Computers 13 (1964), no. 1, pp. 14–17.

[178] Alex J. Wilkie, *Some results and problems on weak systems of arithmetic*, in: Logic Colloquium '77 (A. Macintyre, ed.), North-Holland, 1978, pp. 285–296.

[179] Alex J. Wilkie and Jeff B. Paris, *On the scheme of induction for bounded arithmetic formulas*, Annals of Pure and Applied Logic 35 (1987), pp. 261–302.

[180] Chee Keng Yap, *Fundamental problems in algorithmic algebra*, Oxford University Press, 2000.

[181] Don Zagier, *A one-sentence proof that every prime* $p \equiv 1 \pmod 4$ *is a sum of two squares*, American Mathematical Monthly 97 (1990), no. 2, p. 144.

[182] Domenico Zambella, *Notes on polynomially bounded arithmetic*, Journal of Symbolic Logic 61 (1996), no. 3, pp. 942–966.