

# Combinatorics on Words in Isabelle/HOL

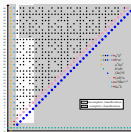
---

Štěpán Holub

October 30, 2024

# FormalCoW Project

- Motivation: classification of binary equality words
  - ŠH, Jiří Sýkora, Jana Hadravová



- GAČR project (2020-2023)
  - with Š. Starosta (ČVUT) and Martin Raška (MFF)

- Gitlab repository
  - ~ 3000 lemmas
  - ~ 50 000 lines of codes



- Archive of formal proofs (AFP)





Home

Topics

Download

Help

Submission

Statistics

About

and [\[Manuch, 01\]](#) for another proof.

in [Computer science/Automata and formal languages](#)

---

## Intersection of two monoids generated by two element codes

Štěpán [Holub](#), Štěpán Starosta

2023

This article provides a formalization of the classification of intersection  $\{x, y\}^* \cap \{u, v\}^*$  of two monoids generated by two element codes. Namely, the intersection has one of the following forms  $\{\beta, \gamma\}^*$  or  $(\beta_0 + \beta(\gamma(1 + \delta + \dots + \delta^t))^* \epsilon)^*$ . Note that it can be infinitely generated. The result is due to [\[Karhumäki, 84\]](#). Our proof uses the terminology of morphisms which allows us to formulate the result in a shorter and more transparent way.

in [Computer science/Automata and formal languages](#)

---

## Combinatorics on Words Basics

Štěpán [Holub](#), Martin Raška, Štěpán Starosta


2021

We formalize basics of Combinatorics on Words. This is an extension of existing theories on lists. We provide additional properties related to prefix, suffix, factor, length and rotation. The topics include prefix and suffix comparability, mismatch, word power, total and reversed morphisms, border, periods, primitivity and roots. We also formalize basic, mostly folklore, results related to word equations.

Search or go to...

FormalCoW /  Combinatorics on Words Formalized

## Project

 Combinatorics on Words Formalized Manage > Plan > Code vMerge requests 0

Repository

Branches

Commits

Tags

Repository graph

Compare revisions

 Build > Help

# Combinatorics on Words Formalized

 Star 2  master v combinatorics-on-words-formalized

History

Find file

Code v







**Update README.md: update references**

Stépán Starosta authored 1 month ago



f6c26e36




Name	Last commit	Last update
 CoW	v1.10.1	1 year ago
 CoW_Equations	v1.9: new session CoW_PCP	1 year ago
 CoW_Graph_Lemma	v1.10.0	1 year ago
 CoW_Infinite	v1.10.1	1 year ago
 CoW_Interpretations	v1.10.1	1 year ago
 CoW_Lyndon	v1.10.0	1 year ago

**Project information**

Formalization of Combinatorics on Words in the generic proof assistant Isabelle/HOL.

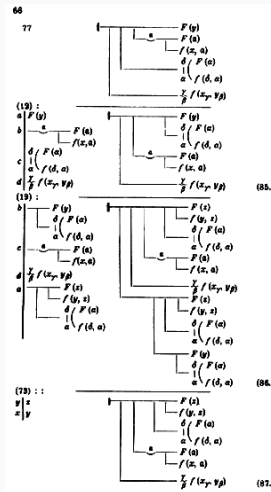
Combinatoric... Isabelle Isabelle/HOL

 56 Commits 4 Branches 4 Tags 3 Releases README BSD 3-Clause "New" or "Revised" License CONTRIBUTING

# Proof assistants

*The dominance of classical first-order logic is challenged every year by something new.*

Larry Paulson, 1989



Mizar



## Freek Wiedijk's 100 theorems

<a href="#">Isabelle</a>	90
<a href="#">HOL Light</a>	87
<a href="#">Coq</a>	79
<a href="#">Lean</a>	76
<a href="#">Metamath</a>	74
<a href="#">Mizar</a>	69
nqthm/ACL2	47
<a href="#">ProofPower</a>	43
PVS	26
<a href="#">Megalodon</a>	12
<a href="#">Naproche</a>	10
NuPRL/MetaPRL	8

- Isar structured language
- sledgehammer
- generic prover  
(meta-logic / object-logic distinction)
- polymorphic simple types
  - 'a list
  - but not  $n$ -tuples



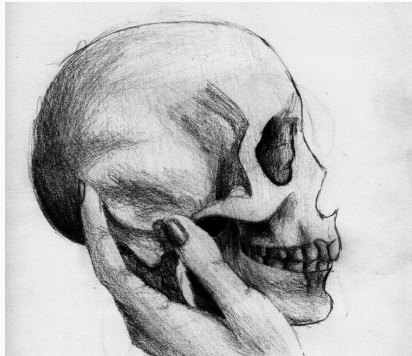
## Freek Wiedijk's 100 theorems

<a href="#">Isabelle</a>	90
<a href="#">HOL_Light</a>	87
<a href="#">Coq</a>	79
<a href="#">Lean</a>	76
<a href="#">Metamath</a>	74
<a href="#">Mizar</a>	69
nqthm/ACL2	47
<a href="#">ProofPower</a>	43
PVS	26
<a href="#">Megalodon</a>	12
<a href="#">Naproche</a>	10
NuPRL/MetaPRL	8

- Isar structured language
- sledgehammer
- generic prover  
(meta-logic / object-logic distinction)
- polymorphic simple types
  - 'a list
  - but not  $n$ -tuples



simple types  
or  
dependent types?





# Grothendieck's schemes : a rivalry case study



Kevin Buzzard's 2020 challenge:

What can Isabelle/HOL actually do before it breaks? Nobody knows. [...] Can your system even do schemes? I don't know. Does anyone know? If it cannot then this would be very valuable to know because it will help mathematician early adopters to make an informed decision about which system to use. [...]

I am not sure that it is even possible to write this code in Isabelle/HOL in such a way that it will run in finite time, [...], and a sheaf is a dependent type, and your clever HOL workarounds will not let you use typeclasses [...]

# Grothendieck's schemes : a rivalry case study



Lawrence Paulson's *Simple Type Theory is not too Simple*, 2021:

Challenge accepted and met [...]

Kevin Buzzard reported on his blog the difficulties he faced during his formalization in Lean:

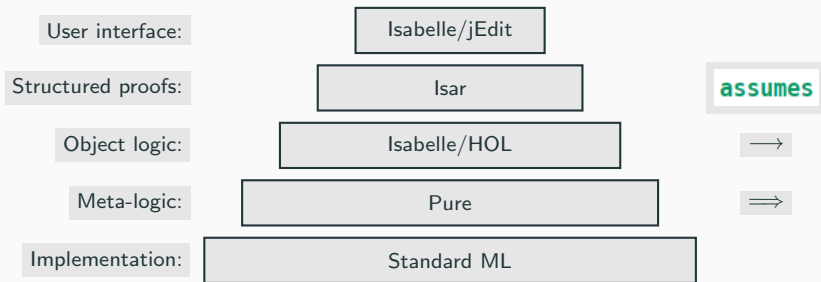
"The project is completely incompatible with modern Lean and mathlib [...]. During our proof we ran into a huge problem [...]"

We did not meet this difficulty in Isabelle [...]

It seems that this difficulty cannot possibly have a direct counterpart in Isabelle, since it arose from the difference between the so-called *equality types*, and the *definitional equality*, a difference which is peculiar to dependent type theories.

## A short history

- A. Church:  $\lambda$ -calculus (untyped, typed, 1930–1940)
- D. Scott: interpretation by continuous function logic (PP $\lambda$ , 1969, 1993)
- D. Milner, L. Paulson: Logic of Computable Functions (LCF, implementation of PP $\lambda$ , 1972–1995)
- D. Milner: Meta Language (ML), programming language for Edinburgh LCF (1973)
- M. Gordon: HOL (based on LCF and higher order logic, originally for hardware verification, 1980s)
- L. Paulson: Isabelle, generic framework for arbitrary object logic



# Binary equality words

A word  $w \in \{a, b\}^*$  for which there are **non-periodic** morphisms  $g$  and  $h$  such that  $g(w) = h(w)$

**Example** (Czeizler, Karhumäki, Laine, ŠH, 2007): *aabbbbaa* is *not* a binary equality word.

That is,

$$xyyyxx = uvvvuu$$

can hold only trivially

