

# Formalising Nominal AC-Unification

Gabriel Ferreira Silva (Universidade de Brasília - UnB)

CICM 2022 Mentor: David Cerna (Czech Academy of Sciences)

Advisor: Mauricio Ayala-Rincón (Universidade de Brasília - UnB)

Co-Advisor: Maribel Fernández (King's College London)

<https://gabriel951.github.io/>

My CICM 2022 mentor is David Cerna. I usually work with Mauricio Ayala-Rincón, Maribel Fernandez and Daniele Nantes.



Figure: David  
Cerna



Figure:  
Mauricio  
Ayala-Rincón



Figure:  
Maribel  
Fernández



Figure:  
Daniele  
Nantes

# Outline

1. Introduction
2. First Order AC-Unification
  - What is Tricky About AC?
  - Example of AC-Unification
3. Adapting to Nominal
4. Conclusion and Future Work
5. More Details About First-Order AC-Unification
6. More Details About Adapting to Nominal AC-Unification

# Systems with Bindings

Systems with bindings appear frequently in mathematics and computer science, but are not captured adequately in first-order syntax.

For instance, the formulas  $\exists x : x \geq 0$  and  $\exists y : y \geq 0$  are not syntactically equal, but should be considered equivalent in a system with binding.

# Nominal

The nominal setting extends first-order syntax, replacing the concept of syntactical equality by  $\alpha$ -equivalence, which let us represent smoothly those systems.

Profiting from the nominal paradigm implies adapting basic notions (substitution, rewriting, equality) to it.

# Nominal Terms

Nominal terms are inductively generated according to the grammar:

$$t, s ::= a \mid \pi \cdot X \mid [a]t \mid f(t_1, \dots, t_n)$$

where  $\pi$  is a permutation that exchanges a finite number of atoms.

# Unification

Unification is about “finding a way” to make two terms equal:

- ▶  $f(a, X)$  and  $f(Y, b)$  can be made equal by “sending”  $X$  to  $b$  and  $Y$  to  $a$ , as they both become  $f(a, b)$ .

Unification has a lot of applications: logic programming, theorem proving, type inference and so on.

# Nominal Unification

Unification with nominal terms is called nominal unification and was solved by Urban, Pitts and Gabbay in [UPG04]. Extensions of nominal unification to equational theories are actively studied today.



# Unification Modulo AC

AC-unification is unification in the presence of associative-commutative function symbols.

For instance, if  $f$  is an AC function symbol, then:

$$f(a, f(b, c)) \approx f(c, f(a, b)).$$

# PhD Goal

PhD goal: propose the first nominal AC-Unification algorithm and formalise its termination, correctness and completeness.

Two steps plan for PhD goal:

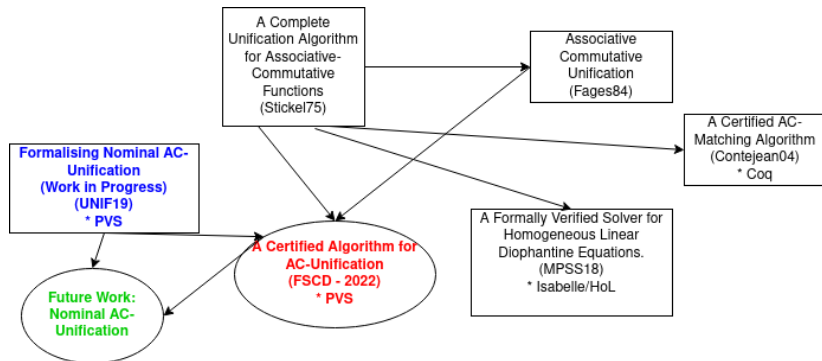
1. Give the first formalisation of a first-order AC-unification algorithm
2. Extend the algorithm to the nominal setting

# Our Work in First Order AC-Unification in a Nutshell

We modified Stickel's seminal AC-unification algorithm to avoid mutual recursion and formalised it in the PVS proof assistant. We proved the adjusted algorithm's termination, soundness and completeness [AFSS22].

The formalisation has 1504 theorems.

# Main Related Work



# What is Tricky About AC? An Example

Let  $f$  be an AC function symbol. The solutions that come to mind when unifying:

$$f(X, Y) \approx^? f(a, Z)$$

are:  $\{X \rightarrow a, Y \rightarrow Z\}$  and  $\{X \rightarrow Z, Y \rightarrow a\}$ .

Are there other solutions?

## What is Tricky About AC? An Example

Yes!

For instance,  $\{X \rightarrow f(a, Z_1), Y \rightarrow Z_2, Z \rightarrow f(Z_1, Z_2)\}$  and  $\{X \rightarrow Z_1, Y \rightarrow f(a, Z_2), Z \rightarrow f(Z_1, Z_2)\}$ .

## Example of AC-Unification

How do we generate a complete set of unifiers for

$$f(a, X) \approx^? f(b, Y)?$$

# Eliminate Common Arguments

Eliminate common arguments in the terms we are trying to unify.

The problem remains:

$$f(a, X) \approx? f(b, Y).$$



# The Connection with Linear Equations

According to the number of times each argument appear in the terms, transform the unification problem into a linear equation on  $\mathbb{N}$ .

After this step, our equation is:

$$X_1 + X_2 = Y_1 + Y_2,$$

where variable  $X_1$  corresponds to argument  $a$ , variable  $X_2$  corresponds to argument  $X$  and so on.

# Basis of Solutions and New Variables

Generate a basis of solutions to the linear equation and associate a new variable with each solution.

Table: Solutions for the Equation  $X_1 + X_2 = Y_1 + Y_2$

$X_1$	$X_2$	$Y_1$	$Y_2$	$X_1 + X_2$	$Y_1 + Y_2$	New Variables
0	1	0	1	1	1	$Z_1$
0	1	1	0	1	1	$Z_2$
1	0	0	1	1	1	$Z_3$
1	0	1	0	1	1	$Z_4$

## Relating Old and New Variables

Observing Table 2, relate the “old” variables and the “new” ones.

After this step, we obtain:

$$X_1 \approx^? Z_3 + Z_4$$

$$X_2 \approx^? Z_1 + Z_2$$

$$Y_1 \approx^? Z_2 + Z_4$$

$$Y_2 \approx^? Z_1 + Z_3$$

# All Possible Cases

Decide whether we will include (set to 1) or not (set to 0) every “new” variable. Observe that every “old” variable must be different than zero.

In our example, we have  $2^4 = 16$  possibilities of including/excluding the variables  $Z_1, \dots, Z_4$ , but after observing that  $X_1, X_2, Y_1, Y_2$  cannot be set to zero, we have 7 branches.

# Branches

The seven branches:

$$\{X_1 \approx^? Z_4, X_2 \approx^? Z_1, Y_1 \approx^? Z_4, Y_2 \approx^? Z_1\}$$

$$\{X_1 \approx^? Z_3, X_2 \approx^? Z_2, Y_1 \approx^? Z_2, Y_2 \approx^? Z_3\}$$

$$\{X_1 \approx^? f(Z_3, Z_4), X_2 \approx^? Z_2, Y_1 \approx^? f(Z_2, Z_4), Y_2 \approx^? Z_3\}$$

$$\{X_1 \approx^? f(Z_3, Z_4), X_2 \approx^? Z_1, Y_1 \approx^? Z_4, Y_2 \approx^? f(Z_1, Z_3)\}$$

$$\{X_1 \approx^? Z_4, X_2 \approx^? f(Z_1, Z_2), Y_1 \approx^? f(Z_2, Z_4), Y_2 \approx^? Z_1\}$$

$$\{X_1 \approx^? Z_3, X_2 \approx^? f(Z_1, Z_2), Y_1 \approx^? Z_2, Y_2 \approx^? f(Z_1, Z_3)\}$$

$$\{X_1 \approx^? f(Z_3, Z_4), X_2 \approx^? f(Z_1, Z_2), Y_1 \approx^? f(Z_2, Z_4), Y_2 \approx^? f(Z_1, Z_3)\}$$

## Dropping Impossible Cases I

Drop the cases where the variables that in fact represent constants or subterms headed by a different AC function symbol are assigned to more than one of the “new” variables.

For instance, the potential new unification problem:

$$\{X_1 \approx^? f(Z_3, Z_4), X_2 \approx^? f(Z_1, Z_2), Y_1 \approx^? f(Z_2, Z_4), Y_2 \approx^? f(Z_1, Z_3)\}$$

should be discarded as the variable  $X_1$ , which represents the constant  $a$ , cannot unify with  $f(Z_3, Z_4)$ .

# Dropping Impossible Cases II

Three branches remain:

$$\{X_1 \approx^? Z_4, X_2 \approx^? Z_1, Y_1 \approx^? Z_4, Y_2 \approx^? Z_1\}$$

$$\{X_1 \approx^? Z_3, X_2 \approx^? Z_2, Y_1 \approx^? Z_2, Y_2 \approx^? Z_3\}$$

$$\{X_1 \approx^? Z_3, X_2 \approx^? f(Z_1, Z_2), Y_1 \approx^? Z_2, Y_2 \approx^? f(Z_1, Z_3)\}$$

## Replacing Variables And Proceeding

Replace variables by the original terms they substituted and proceed with the unification.



# Replacing Variables And Proceeding

The three branches become:

$$\{a \approx^? Z_4, X \approx^? Z_1, b \approx^? Z_4, Y \approx^? Z_1\}$$

$$\{a \approx^? Z_3, X \approx^? Z_2, b \approx^? Z_2, Y \approx^? Z_3\}$$

$$\{a \approx^? Z_3, X \approx^? f(Z_1, Z_2), b \approx^? Z_2, Y \approx^? f(Z_1, Z_3)\}$$

## Solutions For The Example

The solutions will be:

$$\left\{ \begin{array}{l} \sigma_1 = \{Z_3 \rightarrow a, X \rightarrow b, Y \rightarrow a\}, \\ \sigma_2 = \{Z_3 \rightarrow a, X \rightarrow f(b, Z_1), Y \rightarrow f(a, Z_1)\} \end{array} \right\}$$

which, since  $Z_3$  is not part of the original problem, can be simplified to:

$$\left\{ \begin{array}{l} \sigma_1 = \{X \rightarrow b, Y \rightarrow a\}, \\ \sigma_2 = \{X \rightarrow f(b, Z_1), Y \rightarrow f(a, Z_1)\} \end{array} \right\}$$

# Adapting to Nominal

We believe it won't be too hard to adapt the proofs of soundness and completeness to nominal AC-unification.

Termination will be harder. Equations such as

$$f(X, W) \approx^? f(\pi \cdot X, \pi \cdot Y)$$

give us a loop.

# The Loop

After solving the corresponding Diophantine equation, we generate 7 branches. One of them is:

$$\{X \approx? Y_1 + X_1, W \approx? Z_1 + W_1, \pi \cdot X \approx? W_1 + X_1, \pi \cdot Y \approx? Z_1 + Y_1\}$$

and after we instantiate the variables that we can we get:

$$P_1 = \{f(\pi \cdot Y_1, \pi \cdot X_1) \approx? f(W_1, X_1)\},$$

$$\sigma = \{X \mapsto f(Y_1, X_1), W \mapsto f(Z_1, W_1), Y \mapsto f(\pi^{-1} \cdot Z_1, \pi^{-1} \cdot Y_1)\}$$

# The Loop

The problem before and after are respectively:

$$P = \{f(X, W) \approx^? f(\pi \cdot X, \pi \cdot Y)\}$$
$$P_1 = \{f(X_1, W_1) \approx^? f(\pi \cdot X_1, \pi \cdot Y_1)\}$$

# Characterizing Problematic Equations

If  $f(t_1, \dots, t_m) \approx^? f(s_1, \dots, s_n)$  and  $\pi_1 \cdot X \in \{t_1, \dots, t_m\}$  and  $\pi_2 \cdot X \in \{s_1, \dots, s_n\}$ , we may get a loop.

# Solutions in the nominal setting

**Nominal Unification** - Algorithm output is a pair  $\langle \nabla, \sigma \rangle$ .

**Nominal C-Unification** - Fixed-point equations (e.g.  $\pi \cdot X \approx^? X$ ) are included in the algorithm output, which is a set of triples  $S = \{ \langle \nabla_1, \sigma_1, FP_1 \rangle, \dots, \langle \nabla_n, \sigma_n, FP_n \rangle \}$ .

**Nominal AC-Unification** - Should we include equations like  $f(X, W) \approx^? f(\pi X, \pi Y)$  as part of the algorithm output?

## Sketch of a Result

We have a sketch that shows that **for the particular problem**  $f(X, W) \approx^? f(\pi \cdot X, \pi \cdot Y)$  we do not need to loop forever and we can obtain a complete set of triples.

This is connected to the fact that an arbitrary permutation  $\pi$  has an order  $k$  such that  $\pi^k = Id$ .



## Detour through C-Unification and C-Matching

In the beginning of my PhD I worked in generalising a nominal C-unification algorithm to also handle C-matching, as described in [AdCSF<sup>+</sup>21].

# Conclusion

After 3 years of PhD (1 year remains):

1. Give the first formalisation of a first-order AC-unification algorithm - Done
2. Extend the algorithm to the nominal setting - TO DO

## Paths of Future Work

- ▶ Formalise more efficient first-order AC-unification algorithms.
- ▶ Investigate nominal unification with other equational theories.
- ▶ Investigate the relationship between higher-order pattern AC-unification and nominal AC-unification (Thanks to David Cerna!)

## Tools we are using

We use the **PVS** proof assistant to do the formalisation.

We found writing structured proofs filled with details useful before proceeding to formalise a proof in PVS and for that we are using Leslie Lamport's **pf2** LaTeX package.



Figure: PVS logo

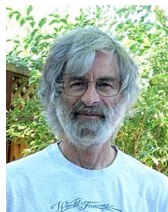


Figure: Leslie Lamport




## We welcome suggestions!

- ▶ Is there any other tool we should be using?
- ▶ Some research direction that we may have missed?
- ▶ And other suggestions of course.

Thank You

**Thank you! Any comments/suggestions/doubts?**

# References I

-  Mauricio Ayala-Rincón, Washington de Carvalho Segundo, Maribel Fernández, Gabriel Ferreira Silva, and Daniele Nantes-Sobrinho, *Formalising nominal C-unification generalised with protected variables*, Math. Struct. Comput. Sci. (2021).
-  Mauricio Ayala-Rincón, Maribel Fernández, Gabriel Ferreira Silva, and Daniele Nantes Sobrinho, *A Certified Algorithm for AC-Unification*, Formal Structures for Computation and Deduction, FSCD 2022 (2022).
-  C. Urban, A. M. Pitts, and M. J. Gabbay, *Nominal Unification*, Theor. Comput. Sci. (2004).

# Input and Output of Algorithm

The algorithm `ACUnif` is recursive and receives as input a triple  $(P, \sigma, V)$ , where  $P$  is the current unification problem,  $\sigma$  is the substitution computed so far and  $V$  are the variables that are/were in the problem.

To unify two terms  $t$  and  $s$  the initial call is with  $P = \{t \approx^? s\}$ ,  $\sigma = Id$  and  $V = Vars(t, s)$ .

The output is a list of substitutions, where each substitution  $\delta$  in this list is an AC-unifier of  $P$ .



# Termination

Almost a decade after Stickel's paper on AC-unification, Fages discovered and fixed a flaw in the proof of termination. Our proof of termination is based on Fages' proof.

# Soundness

The proof of soundness was straightforward.

Notation:  $\delta \subseteq V$

Let  $V$  be a set of variables and  $\delta$  a substitution. We say that  $\delta \subseteq V$  if  $\text{dom}(\delta) \subseteq V$  and  $\text{Vars}(\text{im}(\delta)) \subseteq V$ .

# Completeness

We first had to prove Theorem 1 and then use it to prove Theorem 2.

Theorem 1 (Completeness of ACUnif with  $\delta \subseteq V$ )

Let  $V$  be a set of variables such that  $\delta \subseteq V$  and  $\text{Vars}(t, s) \subseteq V$ . If  $\delta$  unifies  $t \approx^? s$ , then there is a substitution  $\gamma \in \text{ACUnif}(\{t \approx^? s\}, \text{Id}, V)$  such that  $\gamma \leq_V \delta$ .

Theorem 2 (Completeness of ACUnif)

If  $\delta$  unifies  $t \approx^? s$ , then there is a substitution  $\gamma \in \text{ACUnif}(\{t \approx^? s\}, \text{Id}, \text{Vars}(t, s))$  such that  $\gamma \leq_{\text{Vars}(t, s)} \delta$ .

$$\delta \subseteq V$$

The hypothesis  $\delta \subseteq V$  in the theorems of completeness guarantees that the new variables introduced by ACUnif do not clash with the variables in  $\text{dom}(\delta)$  or the variables in the terms in  $\text{im}(\delta)$ .

The loop in  $f(X, W) \approx? f(\pi \cdot X, \pi \cdot Y)$

We found a loop while solving  $f(X, W) \approx? f(\pi \cdot X, \pi \cdot Y)$ .

## Table of Solutions

The table with the solutions of the Diophantine equations is shown below. The name of the new variables was chosen to make clearer the loop we will fall into.

The Diophantine equation associated<sup>1</sup> is  $U_1 + U_2 = V_1 + V_2$  and the table of solutions is:

Table: Solutions for the Equation  $U_1 + U_2 = V_1 + V_2$

$U_1$	$U_2$	$V_1$	$V_2$	$U_1 + U_2$	$V_1 + V_2$	New Variables
0	1	0	1	1	1	$Z_1$
0	1	1	0	1	1	$W_1$
1	0	0	1	1	1	$Y_1$
1	0	1	0	1	1	$X_1$

---

<sup>1</sup>variable  $U_1$  is associated with argument  $X$ , variable  $U_2$  is associated with argument  $W$ , variable  $V_1$  is associated with argument  $\pi \cdot X$  and variable  $V_2$  is associated with argument  $\pi \cdot Y$ .

## After solveAC

$$\{X \approx? X_1, W \approx? Z_1, \pi \cdot X \approx? X_1, \pi \cdot Y \approx? Z_1\}$$

$$\{X \approx? Y_1, W \approx? W_1, \pi \cdot X \approx? W_1, \pi \cdot Y \approx? Y_1\}$$

$$\{X \approx? Y_1 + X_1, W \approx? W_1, \pi \cdot X \approx? W_1 + X_1, \pi \cdot Y \approx? Y_1\}$$

$$\{X \approx? Y_1 + X_1, W \approx? Z_1, \pi \cdot X \approx? X_1, \pi \cdot Y \approx? Z_1 + Y_1\}$$

$$\{X \approx? X_1, W \approx? Z_1 + W_1, \pi \cdot X \approx? W_1 + X_1, \pi \cdot Y \approx? Z_1\}$$

$$\{X \approx? Y_1, W \approx? Z_1 + W_1, \pi \cdot X \approx? W_1, \pi \cdot Y \approx? Z_1 + Y_1\}$$

$$\{X \approx? Y_1 + X_1, W \approx? Z_1 + W_1, \pi \cdot X \approx? W_1 + X_1, \pi \cdot Y \approx? Z_1 + Y_1\}$$



## After instantiating the variables

7 branches are generated:

$$B1 - \{\pi \cdot X \approx^? X\}, \sigma = \{W \mapsto \pi \cdot Y\}$$

$$B2 - \sigma = \{W \mapsto \pi^2 \cdot Y, X \mapsto \pi \cdot Y\}$$

$$B3 - \{f(\pi^2 \cdot Y, \pi \cdot X_1) \approx^? f(W, X_1)\}, \sigma = \{X \mapsto f(\pi \cdot Y, X_1)\}$$

*B4 - No solution*

*B5 - No solution*

$$B6 - \sigma = \{W \mapsto f(Z_1, \pi \cdot X), Y \mapsto f(\pi^{-1} \cdot Z_1, \pi^{-1} \cdot X)\}$$

$$B7 - \{f(\pi \cdot Y_1, \pi \cdot X_1) \approx^? f(W_1, X_1)\},$$

$$\sigma = \{X \mapsto f(Y_1, X_1), W \mapsto f(Z_1, W_1), Y \mapsto f(\pi^{-1} \cdot Z_1, \pi^{-1} \cdot Y_1)\}$$

# The Loop

Focusing on *Branch7*, notice that the problem before `solveAC` and the problem after `solveAC` and instantiating the variables are:

$$P = \{f(X, W) \approx^? f(\pi \cdot X, \pi \cdot Y)\}$$
$$P_1 = \{f(X_1, W_1) \approx^? f(\pi \cdot X_1, \pi \cdot Y_1)\}$$