Formalisation of nominal equational reasoning in PVS†

nominal anti-unification (the library nasa/pvslib/nominal)

Mauricio Ayala-Rincón

Mathematics and Computer Science Departments

Universidade de Brasília

† Research supported by the Brazilian agencies CAPES, CNPq, and FAPDF

Hausdorff Trimester Program Prospects of Formal Mathematics
Hausdorff Institute for Mathematics, Bonn, August 5th, 2024

Joint Work With

Maria Júlia Dias Lima

Mariano Miguel Moscato

Thaynara Arielly de Lima

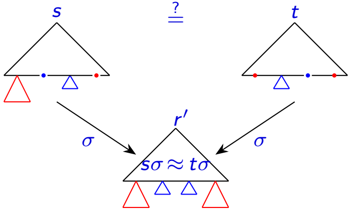
Temur Kutsia

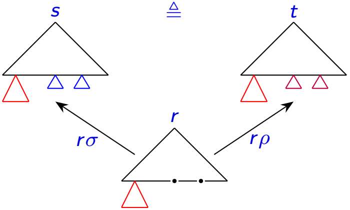
Outline

1. Motivation
2. Formal verification
3. Anti-unification modulo
4. A sound Algorithm for $(\mathfrak{a})(\mathsf{A})(\mathsf{C})(\mathfrak{a}\mathsf{A})(\mathfrak{a}\mathsf{C})$-theories
5. Future Work

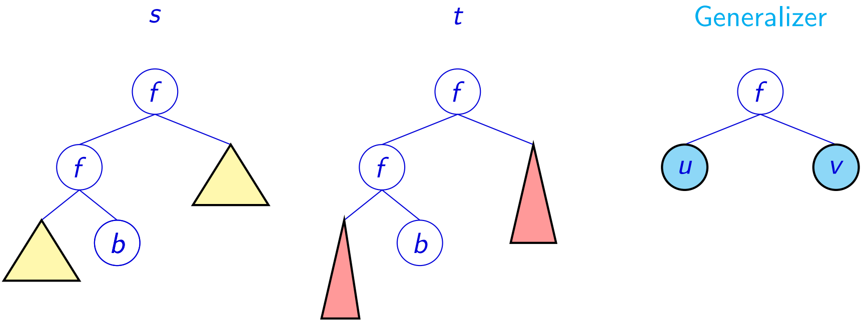
Unification Vs Anti-unification

Unification	Anti-unification
Goal: find a substitution that identifies two expressions.	Goal: find the commonalities between two expressions.

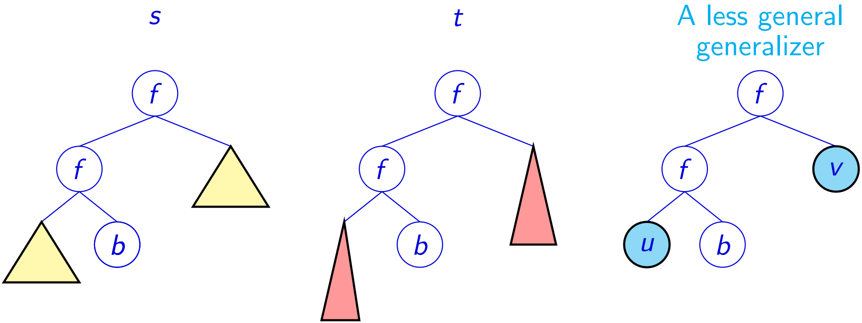




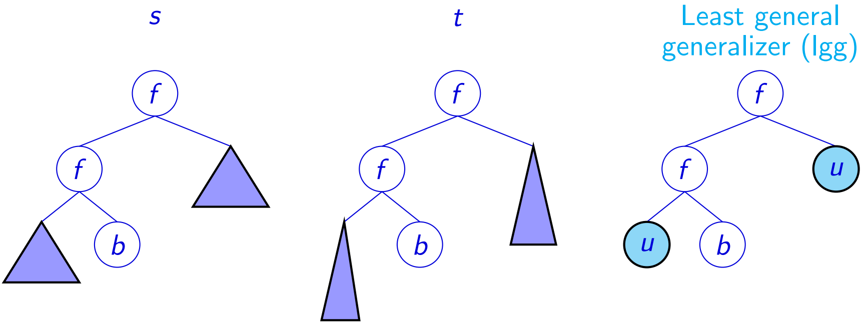
Anti-Unification



Anti-Unification



Anti-Unification



History

- Introduced by Gordon Plotkin [Plo70] and John Reynolds [Rey70]
- First-order: syntactic [Baa91]; C, A, and AC [AEEM14]; idempotent [CK20b], unital [CK20c], semirings [Cer20], absorption [ARCBK24]
- Higher-Order: patterns [BKLV17], top maximal and shallow generalizations variants [CK20a], equational patterns [CK19], modulo [CK20a]
- See david Cerna and Temur Kutsia survey [CK23].

Applications

Applications of anti-unification include:

- searching a large hypothesis space in inductive logic programming (ILP) for logic-based machine learning [CDEM22];
- preventing bugs and misconfigurations in software [MBK+20];
- detecting code clones [VY19];
- searching recursion schemes for efficient parallel compilation [BBH18].

Formal verification - Syntactical case

- terms $t ::= x \mid \langle\rangle \mid \langle t, t\rangle \mid f\ t$
- Labelled equations $E = \{s_i \underset{x_i}{\triangleq} t_i \mid i \leq n\}$

Configurations: $\langle E_U; E_S; \sigma\rangle$

$$\left\langle \underbrace{E_U}_{\text{Unsolved equations}} \quad \underbrace{E_S}_{\text{Solved equations}} \quad \underbrace{\sigma}_{\text{Substitution}} \right\rangle$$

Configuration constraints

- All labels in $E_U \cup E_S$ are different,
- no *redundant* equations appear in E_S, and
- no label in $E_U \cup E_S$ belongs to $dom(\sigma)$.

Inference Rules

$$\text{(Decompose Function)}\ \frac{\langle \{ f\ s \overset{\triangle}{\underset{x}{=}} f\ t \} \cup E, S, \sigma \rangle}{\langle \{ s \overset{\triangle}{\underset{y}{=}} t \} \cup E, S, \{ x \mapsto f\ y \} \circ \sigma \rangle}$$

$$\text{(Decompose Pair)}\ \frac{\langle \{ \langle s, u \rangle \overset{\triangle}{\underset{x}{=}} \langle t, v \rangle \} \cup E, S, \sigma \rangle}{\langle \{ s \overset{\triangle}{\underset{y}{=}} t, u \overset{\triangle}{\underset{z}{=}} v \} \cup E, S, \{ x \mapsto \langle y, z \rangle \} \circ \sigma \rangle}$$

$$\text{(Solve-Red)}\ \frac{\langle \{ s \overset{\triangle}{\underset{x}{=}} t \} \cup E, S, \sigma \rangle}{\langle E, S, \{ x \mapsto x' \} \circ \sigma \rangle}\ \text{if } s \overset{\triangle}{\underset{x'}{=}} t \in S$$

$$\text{(Solve-No-Red)}\ \frac{\langle \{ s \overset{\triangle}{\underset{x}{=}} t \} \cup E, S, \sigma \rangle}{\langle E, \{ s \overset{\triangle}{\underset{x}{=}} t \} \cup S, \sigma \rangle}\ \text{if there is no } s \overset{\triangle}{\underset{x'}{=}} t \in S$$

$$\text{(Syntactic)}\ \frac{\langle \{ s \overset{\triangle}{\underset{x}{=}} s \} \cup E, S, \sigma \rangle}{\langle E, S, \{ x \mapsto s \} \circ \sigma \rangle}\ \text{if neither decomposable nor solvable}$$

Inference Rules

Example

$$
\begin{array}{rc}
\text{(DecFun)} & \dfrac{\langle \{ f\langle f\langle c, b\rangle, c\rangle \triangleq_{x} f\langle f\langle d, b\rangle, d\rangle \}, \emptyset, id\rangle}{\langle \{ \langle f\langle c, b\rangle, c\rangle \triangleq_{y} \langle f\langle d, b\rangle, d\rangle \}, \emptyset, \{x \mapsto f\ y\}\rangle} \\
\text{(DecPair)} & \dfrac{}{\langle \{ f\langle c, b\rangle \triangleq_{z_1} f\langle d, b\rangle, c \triangleq_{z_2} d \}, \emptyset, \{x \mapsto f\ \langle z_1, z_2\rangle\}\rangle} \\
\text{(DecFun)} & \dfrac{}{\langle \{ \langle c, b\rangle \triangleq_{z_3} \langle d, b\rangle, c \triangleq_{z_2} d \}, \emptyset, \{x \mapsto f\ \langle f\ z_3, z_2\rangle\}\rangle} \\
\text{(DecPair)} & \dfrac{}{\langle \{ c \triangleq_{z} d, b \triangleq_{z_4} b, c \triangleq_{z_2} d \}, \emptyset, \{x \mapsto f\ \langle f\ \langle z, z_4\rangle, z_2\rangle\}\rangle} \\
\text{(SolveNRed)} & \dfrac{}{\langle \{ b \triangleq_{z_4} b, c \triangleq_{z_2} d \}, \{c \triangleq_{z} d\}, \{x \mapsto f\ \langle f\ \langle z, z_4\rangle, z_2\rangle\}\rangle} \\
\text{(Syntactic)} & \dfrac{}{\langle \{ c \triangleq_{z_2} d \}, \{c \triangleq_{z} d\}, \{x \mapsto f\ \langle f\ \langle z, b\rangle, z_2\rangle\}\rangle} \\
\text{(SolRed)} & \dfrac{}{\emptyset, \{c \triangleq_{z} d\}, \{x \mapsto f\ \langle f\ \langle z, b\rangle, z\rangle\}\rangle}
\end{array}
$$

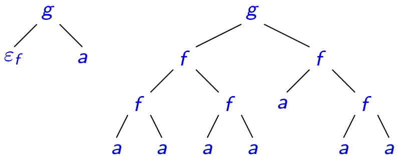
Anti-unification modulo

- Interest on the formalization of anti-unification for theories with Commutative, Associative and Absorption-symbols: C-, A-, and $\mathfrak{a}$-symbols.
- Related $\mathfrak{a}$-symbols are a pair of a function and a constant symbol holding the axioms $f(\varepsilon_f, x) = \varepsilon_f = f(x, \varepsilon_f)$.

Anti-unification in $(\mathfrak{a})(\mathrm{A})(\mathrm{C})(\mathfrak{a}\mathrm{A})(\mathfrak{a}\mathrm{C})$-theories

Example

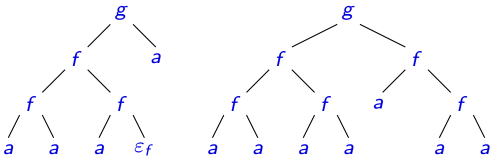
Consider the terms:



An $\mathfrak{a}$-generalization and $\mathfrak{a}$A-generalization will be illustrated.

Anti-unification in $(\mathfrak{a})(A)(C)(\mathfrak{a}A)(\mathfrak{a}C)$-theories

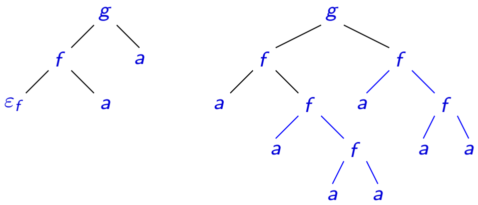
By *expanding* ε_f in $g(\varepsilon_f, a)$, one obtains:



Notice that $g(f(f(a,a), f(a,x)), y)$ is an $\mathfrak{a}$-generalization.

Anti-unification in ($\mathfrak{a}$)(A)(C)($\mathfrak{a}$A)($\mathfrak{a}$C)-theories

Considering the same terms modulo $\mathfrak{a}$A, and by *expanding* ε_f in $g(\varepsilon_f, a)$, one has:



$g(f(x, y), y)$ is an $\mathfrak{a}$A-generalization but not an $\mathfrak{a}$-generalization.

Anti-unification types for

Theory	Anti-unification type	References
Syntactic	1	[Plo70, Rey70]
A	ω	[AEEM14]
C	ω	[AEEM14]
Unital† $(\mathrm{U})^{1}$ / $(\mathrm{U})^{\geq 2}$	ω/nullary	[CK20c]
$\mathfrak{a}$	∞	[ARCBK24]

($\dagger$)Unital: $\{f(i_f, x) = f(x, i_f) = x\}$

Termination and Soundness

Termination

Syntactic anti-unification is terminating.

Soundness

For any valid configuration, syntactic anti-unification computes a least general generalizer.

Completeness

Any generalizer of a given input configuration is equal to or more general than the generalizer computed by syntactic anti-unification.

Conclusions and Future Work

Conclusions

- Although anti-unification has become of increasing interest, the verification of anti-unification algorithms has not been explored.
- The development of procedures to solve anti-unification modulo theories is crucial.
- Only recently, anti-unification modulo $\mathfrak{a}$-, C-, and ($\mathfrak{a}$C)-symbols have been addressed. Procedures combining such properties have been shown to be challenging from theoretical and practical perspectives.

Danke shön

Danke shön!

References I

- María Alpuente, Santiago Escobar, Javier Espert, and José Meseguer, *A modular order-sorted equational generalization algorithm*, Information and Computation **235** (2014), 98–136.
- Mauricio Ayala-Rincón, David M. Cerna, Andrés Felipe Gonzáez Barragán, and Temur Kutsia, *Equational anti-unification over absorption theories*, Automated Reasoning - 12th International Joint Conference, IJCAR 2024, Nancy, France, Proceedings, Lecture Notes in Computer Science, vol. 14740, Springer, 2024, pp. 317–337.
- Franz Baader, *Unification, weak unification, upper bound, lower bound, and generalization problems*, Rewriting Techniques and Applications, 4th International Conference, RTA-91, Como, Italy, April 10-12, 1991, Proceedings, Lecture Notes in Computer Science, vol. 488, Springer, 1991, pp. 86–97.
- Adam D. Barwell, Christopher Brown, and Kevin Hammond, *Finding parallel functional pearls: Automatic parallel recursion scheme detection in haskell functions via anti-unification*, Future Gener. Comput. Syst. **79** (2018), 669–686.
- Alexander Baumgartner, Temur Kutsia, Jordi Levy, and Mateu Villaret, *Higher-order pattern anti-unification in linear time*, J. Autom. Reason. **58** (2017), no. 2, 293–310.

References II

Andrew Cropper, Sebastijan Dumancic, Richard Evans, and Stephen H. Muggleton, *Inductive logic programming at 30*, Mach. Learn. **111** (2022), no. 1, 147–172.

David M. Cerna, *Anti-unification and the theory of semirings*, Theo. Com. Sci. **848** (2020), 133–139.

David M. Cerna and Temur Kutsia, *A generic framework for higher-order generalizations*, 4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany (Herman Geuvers, ed.), LIPIcs, vol. 131, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, pp. 10:1–10:19.

______, *Higher-order pattern generalization modulo equational theories*, Math. Struct. Comput. Sci. **30** (2020), no. 6, 627–663.

______, *Idempotent anti-unification*, ACM Trans. Comput. Log. **21** (2020), no. 2, 10:1–10:32.

______, *Unital anti-unification: type algorithms*, 5th International Conference on Formal Structures for Computation and Deduction, FSCD **167** (2020), no. 6, 26:1–26:20.

References III

——, *Anti-unification and generalization: A survey*, Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China, ijcai.org, 2023, pp. 6563–6573.

Sonu Mehta, Ranjita Bhagwan, Rahul Kumar, Chetan Bansal, Chandra Maddila, B. Ashok, Sumit Asthana, Christian Bird, and Aditya Kumar, *Rex: Preventing bugs and misconfiguration in large services using correlated change analysis*, 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2020, pp. 435–448.

Gordon D. Plotkin, *A note on inductive generalization*, Machine Intelligence 5 **5** (1970), 153–163.

John C. Reynolds, *Transformational system and the algebric structure of atomic formulas*, Machine Intelligence 5 **5** (1970), 135–151.

Wim Vanhoof and Gonzague Yernaux, *Generalization-driven semantic clone detection in CLP*, 29th Int. Symposium on Logic-Based Program Synthesis and Transformation, LOPSTR, LNCS, vol. 12042, 2019, pp. 228–242.