Universidade de Brasília

Anti-Unification on Absorption Theories

Andrés Felipe González Barragan[†] (UnB)
Joint work with Mauricio Ayala-Rincón (UnB), Temur Kutsia (RISC - U. Linz)
David Cerna (CAS ICS)

[†] Author supported by a Brazilian CAPES Scholarship

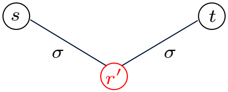
XVI Summer Workshop In Mathematics

Brasilia, January 7th, 2024

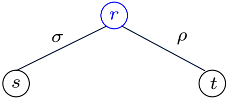
Outline

Unification Vs Anti-unification

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



where $t\sigma \approx r' \approx s\sigma$.



where $r\sigma \approx s$ and $r\rho \approx t$.

Example 1.

Consider the binary symbol $\cdot(x, y)$ as the product over natural numbers, $0, 1, 2$ as constants, and the terms $\cdot(\cdot(1, x), 2)$ and $\cdot(\cdot(1, 1), y)$.

Unification Vs Anti-unification

Unification

The unification is given by the substitution $\sigma = \{x \mapsto 1, y \mapsto 2\}$, because

$$\cdot(\cdot(1, x), 2)\sigma = \cdot(\cdot(1, 1), 2) = \cdot(\cdot(1, 1), y)\sigma$$

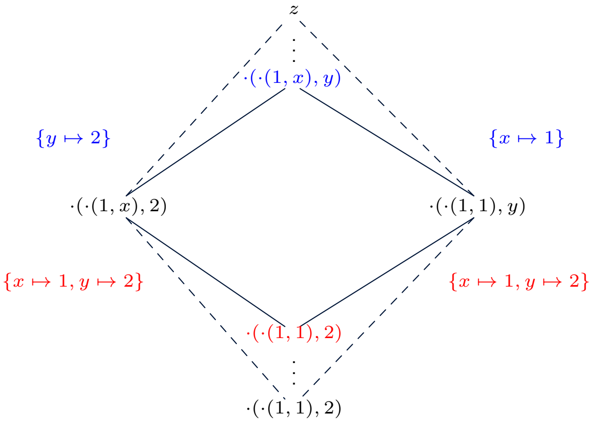
Anti-Unification

A generalization is the term $\cdot(\cdot(1, x), y)$ with the substitutions $\sigma = \{y \mapsto 2\}$ and $\rho = \{x \mapsto 1\}$, because

$$\cdot(\cdot(1, x), y)\sigma = \cdot(\cdot(1, x), 2)$$

$$\cdot(\cdot(1, x), y)\rho = \cdot(\cdot(1, 1), y)$$

Unification Vs Anti-unification



Applications

One interesting application is preventing bugs and misconfigurations in software (Mehta et al. (MEHTA et al., 2020)):

- given a version of an application code configuration,
- verify an updated version.

Applications

For example, the next fragment of code generates the next environment of icons of an application on Swift:

```
ScrollView {
    LazyVGrid(columns: columns) {
        ForEach(symbolNames, id: \.self) {
         symbolItem in
            Button {
                event.symbol = symbolItem
            } label: {
                Image(systemName: symbolItem)
                    .imageScale(.large)
                    .foregroundColor
                     (selectedColor)
                    .padding(5)
            }
            .buttonStyle(.plain)
        }
    }
    .drawingGroup()
}
```

Applications

Then, if we update the general code in different parts, getting:

Original Version	Update Version

Original Version:

```
ScrollView {
   LazyVGrid(columns: columns) {
     ForEach(SymbolNames, id: \.self) {
        SymbolItem in
          Button {
            event.symbol = symbolItem
          } (...)
```

Update Version:

```
ScrollView {
   LazyVGrid(columns: columns) {
     ForEach(SymbolNames, id: \.1) {
        SymbolItem in
          Button {
            event.symbol = symbolItem
          } (...)
```

Applications

The updated version has an error, and using anti-unification we can detect that the next fragment of code that is a generalization of the two codes:

```
ScrollView {
   LazyVGrid(columns: columns) {
     ForEach(SymbolNames, id: x ) {
        SymbolItem in
          Button {
            event.symbol = symbolItem
          } (...)
```

With substitutions $\sigma = \{x \mapsto \backslash\texttt{.self}\}$ and $\rho = \{x \mapsto \backslash\texttt{.1}\}$.

Applications

Applications of anti-unification include:

- searching parallel recursion schemes to transform sequential algorithms into parallel algorithms (Barwell et al. (BARWELL; BROWN; HAMMOND, 2018));
- preventing bugs and misconfigurations in software (Mehta et al. (MEHTA et al., 2020));
- finding duplicate code and similarities;
- detecting code clones (i.e., plagiarism).

Absorption Theory

- The alphabet consists of a countable set of variables $\mathcal{V}$ and set $\mathcal{F}$ of function and with a special constant symbol $\star$ (The wild card).
- Terms over this alphabet, $\mathcal{T}(\mathcal{F}, \mathcal{V})(\mathcal{T})$ and $\mathcal{T}(\mathcal{F} \cup \{\star\}, \mathcal{V})(\mathcal{T}_\star)$, defined as usually:

$$t := x \mid f(t_1, \ldots, t_n)$$

- A finite set E that consists of equations $s \approx t$.
- A preorder $\preceq_E$, which states that $s \preceq_E t$ if there exists a substitution σ such that $s\sigma \approx_E t$.

Absorption Theory

The type of an anti-unification modulo E problem is classified as below.

- *Nullary*(0): if there are terms s and t such that $\mathtt{mcsg}_E(s,t)$ does not exist. Also, called *type zero*.
- *Unitary*(1): if for all s and t, $\mathtt{mcsg}_E(s,t)$ has just one generalization.
- *Finitary*(ω): if for all s and t, $\mathtt{mcsg}_E(s,t)$ has more than one generalization.
- *Infinitary*(∞): there are terms s and t such that $\mathtt{mcsg}_E(s,t)$ is infinite.

Type of some Theories

Theory	Type	Authors and References
Syntactic ($\emptyset$)	1	G. Plotkin and J. Reynolds (PLOTKIN, 1970; REYNOLDS, 1970)
Associativity (A)	ω	M. Alpuente et al. (ALPUENTE et al., 2014)
Commutativity (C)	ω	M. Alpuente et al. (ALPUENTE et al., 2014)
Unital (U)	ω	D. Cerna (CERNA; KUTSIA, 2020a)
Idempotency$_{\geq 1}$ (I)	∞	D. Cerna and T. Kutsia (CERNA; KUTSIA, 2020a)
Unital$_{\geq 2}$ (U_2)	0	D. Cerna and T. Kutsia (CERNA; KUTSIA, 2020b)

Absorption Theory

- An *anti-unification equation* (AUE) between s and t in a normal form is denoted by $s \triangleq_x t$, where x is called as label.
- A *valid set of AUEs* is a set of AUEs where all the labels are different.
- An AUE $s \triangleq_x t$ is *solved* if $head(s)$ and $head(t)$ are not related absorption symbols, where $s, t \in \mathcal{T}$.
- An AUE $s \triangleq_x t$ is *wild* if one of the terms is the wild card and the other belongs to $\mathcal{T}_\star$.

Absorption Theory

Absorption Theory

Absorption is an important algebraic attribute in some magmas: for some function symbol f there is a constant ε_f such that

$$f(x, \varepsilon_f) \approx \varepsilon_f, \text{ or/and } f(\varepsilon_f, x) \approx \varepsilon_f$$

Equational theories with these equations are called absorption theories (`Abs`).

Example 2

Let's find one generalization of the AUE $\varepsilon_f \triangleq f(f(a, b), c)$.

Absorption Theory

The idea of the algorithm is to expand the ε_f to get the generalization:

$\varepsilon_f \triangleq_x f(f(a,b),c)$	x
$f(\varepsilon_f, c) \triangleq_x f(f(a,b),c)$	x
$\varepsilon_f \triangleq_y f(a,b), c \triangleq_z c$	$f(y,z)$
$f(\varepsilon_f, b) \triangleq_y f(a,b)$	$f(y,c)$
$\varepsilon_f \triangleq_u a, b \triangleq_v b$	$f(f(u,v),c)$
$\varepsilon_f \triangleq_u a$	$f(f(u,b),c)$

Notice that the terms

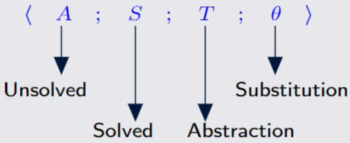
$$f(f(a,u),c), f(f(a,b),u), f(f(u,b),c)$$

are generalizations of the initial terms.

Algorithm `AUnif`

Algorithm for absorption theory

The algorithm `AUnif` is an exhaustive application of inference rules of configurations of the form



Algorithm AUnif

Inference Rules

Then we define the next rules

(Dec): **Decompose**

$$\langle\{f(s_1,\ldots,s_n) \triangleq_x f(t_1,\ldots,t_n)\} \sqcup A; S; \theta\rangle$$

$$\overset{Dec}{\Longrightarrow} \langle\{s_1 \triangleq_{y_1} t_1,\ldots,s_n \triangleq_{y_n} t_n\} \cup A; S; \theta\{x \mapsto f(y_1,\ldots,y_n)\}\rangle$$

For f any function symbol, $n > 0$, and $y_1,\ldots,y_n$ are fresh variables.

Algorithm `AUnif`

Inference Rules

(Solve): **Solve**

$$\langle \{s \triangleq_x t\} \sqcup A; S; T; \theta\rangle \overset{Sol}{\Longrightarrow} \langle A; \{s \triangleq_x t\} \cup S; T; \theta\rangle$$

Where $head(s) \neq head(t)$ are not related absorption symbols.

(Mer): **Merge**

$$\langle \emptyset; \{s \triangleq_x t\} \cup \{s \triangleq_y t\} \cup S; \theta\rangle \overset{Mer}{\Longrightarrow} \langle \emptyset; \{s \triangleq_y t\} \cup S; \theta\{x \mapsto y\}\rangle$$

Algorithm `AUnif`

Inference Rules

(ExpLA1): **Expansion for Absorption, Left 1**

$$\langle \{\varepsilon_f \triangleq_x f(t_1, t_2)\} \sqcup A; S; T; \theta \rangle$$

$$\stackrel{ExpLA1}{\Longrightarrow} \langle \{\varepsilon_f \triangleq_{y_1} t_1\} \cup A; S; \{\star \triangleq_{y_2} t_2\} \cup T; \theta\{x \mapsto f(y_1, y_2)\} \rangle$$

(ExpLA2): **Expansion for Absorption, Left 2**

$$\langle \{\varepsilon_f \triangleq_x f(t_1, t_2)\} \sqcup A; S; T; \theta \rangle$$

$$\stackrel{ExpLA2}{\Longrightarrow} \langle \{\varepsilon_f \triangleq_{y_2} t_2\} \cup A; S; \{\star \triangleq_{y_1} t_1\} \cup T; \theta\{x \mapsto f(y_1, y_2)\} \rangle$$

Algorithm `AUnif`

Problems do not always have a finite number of generalizations. The next example has an infinite set of them!

Example 3

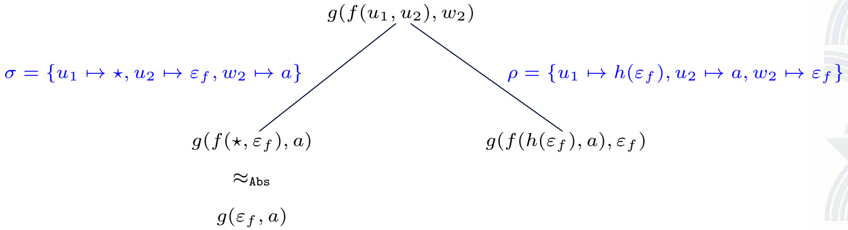
Apply `AUnif` to the anti-unification problem $g(\varepsilon_f, a) \triangleq g(f(h(\varepsilon_f), a), \varepsilon_f)$.

A final configuration given by `AUnif` is

$$\langle \emptyset; \{\varepsilon_f \triangleq_{u_2} a, a \triangleq_{w_2} \varepsilon_f\}; \{\star \triangleq_{u_1} h(\varepsilon_f)\}; \{x \mapsto g(f(u_1, u_2), w_2)\}\rangle$$

Algorithm AUnif

Then $g(f(u_1, u_2), w_2)$ is a generalization with the substitutions σ and ρ.



Notice that any similar term with $h(\varepsilon_f)$ replaced instead u_1 is a generalization too.

Abstraction Set

Termination

AUnif cannot result in an infinite derivation. Also, for a configuration $\mathcal{C}$, the set of final configurations $\mathtt{AUnif}(\mathcal{C})$ is computable in a finite number of steps.

Abstraction Set

Let t be a term in Abs-normal form, and σ be a substitution with images in Abs-normal form. The abstraction of t with respect to σ is the set:

$$\uparrow(t,\sigma) := \{r \mid r\sigma \approx_{\mathtt{Abs}} t,\ r \text{ is an } \mathtt{Abs}\text{-normal form, and } \mathit{Var}(r) \subseteq \mathit{Dom}(\sigma)\}$$

Algorithm AUnif

Example 4

Find the abstraction set of $h(\varepsilon_f)$ with respect to $\rho = \{u_2 \mapsto a, w_2 \mapsto \varepsilon_f\}$:

$$\uparrow (h(\varepsilon_f), \rho) = \{h(\varepsilon_f), h(w_2), h(f(w_2, \_)), h(f(\_, w_2)), h(f(u_2, w_2)), \dots\}$$

Where _ could be replaced by a term whose variables are included in *Dom*(ρ). For example, $h(f(w_2, a))$ and $h(f(w_2, h(g(u_2, w_2))))$ belong to the abstraction set.

Algorithm AUnif

Continue with Example 3:

$$g(\varepsilon_f, a) \triangleq g(f(h(\varepsilon_f), a), \varepsilon_f)$$

The abstraction set give us the "good" terms that can be replaced in the generalization $g(f(u_1, u_2), w_2)$. Hence, other generalizations are

$$g(f(h(\varepsilon_f), u_2), w_2)$$

$$g(f(h(w_2), u_2), w_2)$$

$$g(f(h(f(w_2, a)), u_2), w_2)$$

$$\vdots$$

Soundness and Completeness

Soundness

Let $\langle A_0; S_0; T_0; \theta_0 \rangle \Longrightarrow^* \langle \emptyset; S_n; T_n; \theta_n \rangle$ be a derivation to a final configuration. Then for all $s \triangleq_x t \in A_0 \cup S_0 \cup T_0$, $x\theta_n \in \mathcal{G}_{\mathtt{Abs}}(s, t)$.

Completeness

Let $r \in \mathcal{G}_{\mathtt{Abs}}(t_1, t_2)$. Then for all configurations $\langle A; S; T; \theta \rangle$ such that $t_1 \triangleq_x t_2 \in A$ there exist a final configuration $\langle \emptyset; S'; T'; \theta' \rangle \in \mathtt{AUnif}(\langle A; S; T; \theta \rangle)$ and $\tau \in \Psi(T', S')$ such that $r \preceq_{\mathtt{Abs}} x\theta'\tau$.

The proof of these theorems can be found in (AYALA-RINCÓN et al., 2023).

Type of the Problem

Let s and t be terms and $\texttt{AUnif}(\langle\{s \triangleq_x t\}; \emptyset; \emptyset; \iota\rangle)$ merged. The set of least general generalizations of s and t as

$$\mathcal{C}_{\texttt{AUnif}}(s,t) = \{x\theta\tau \mid \langle\emptyset; S; T; \theta\rangle \in \texttt{AUnif}(\langle\{s \triangleq_x t\}; \emptyset; \emptyset; \iota\rangle) \wedge \tau \in \Psi(T,S)\}.$$

Lemma 1

For all terms s, t, and $g_0, g_1 \in \mathcal{C}_{\texttt{AUnif}}(s,t)$, if $g_0 \neq g_1$ then neither $g_0 \preceq_{\texttt{Abs}} g_1$ nor $g_1 \preceq_{\texttt{Abs}} g_0$ holds.

Type of The Problem

Theorem

Anti-unification modulo Abs theories is of type infinitary.

From Lemma 1 and Completeness Theorem the set $\mathcal{C}_{\texttt{AUnif}}(s,t)$ is the $\texttt{mcsg}(s,t)$, and from Example 3:

$$\mathcal{C}_{\texttt{AUnif}}(g(\varepsilon_f, a); g(f(h(\varepsilon_f)), a), \varepsilon_f)$$

Has infinite $\texttt{lggs}$. Hence, the problem is of type infinitary.

Conclusions and Future Work

Conclusions

- We Introduce a rule-based algorithm that computes generalizations for problems modulo absorption theories and this algorithm is sound and complete.
- Additionally, the algorithm computes least general generalizations (`lggs`), which means that computes a minimal complete set of generalizations. We proved that the problem is of type infinitary.

Conclusions and Future Work

Future Work

- Analyze combinations between absorption theories with Commutative and Associative Theories and build an algorithm that computes the generalizations for this kind of problem.
- Study another Subterm collapse theories, similar to the absorption theories as Absorption theories which can be collapsed for ground terms or terms with variables ($f(x, T) \approx T \approx f(T, x)$, $f(x, t) \approx t \approx f(t, x)$) and unary function symbols that collapse with a ground term ($f(T) \approx T$).

References I

ALPUENTE, M. et al. A modular order-sorted equational generalization algorithm. **Information and Computation**, v. 235, p. 98–136, 2014.

AYALA-RINCÓN, M. et al. Equational anti-unification over absorption theories. **CoRR**, abs/2310.11136, 2023. Disponível em: <https://doi.org/10.48550/arXiv.2310.11136>.

BARWELL, A. D.; BROWN, C.; HAMMOND, K. Finding parallel functional pearls: Automatic parallel recursion scheme detection in haskell functions via anti-unification. **Future Gener. Comput. Syst.**, v. 79, p. 669–686, 2018.

CERNA, D. M.; KUTSIA, T. Idempotent anti-unification. **ACM Trans. Comput. Log.**, v. 21, n. 2, p. 10:1–10:32, 2020.

CERNA, D. M.; KUTSIA, T. Unital anti-unification: type algorithms. **5th International Conference on Formal Structures for Computation and Deduction, FSCD**, v. 167, n. 6, p. 26:1–26:20, 2020.

References II

MEHTA, S. et al. Rex: Preventing bugs and misconfiguration in large services using correlated change analysis. In: **17th USENIX Symposium on Networked Systems Design and Implementation (NSDI)**. [s.n.], 2020. p. 435–448. ISBN 978-1-939133-13-7. Disponível em: <https://www.usenix.org/conference/nsdi20/presentation/mehta>.

PLOTKIN, G. D. A note on inductive generalization. **Machine Intelligence 5**, v. 5, p. 153–163, 1970.

REYNOLDS, J. C. Transformational system and the algebric structure of atomic formulas. **Machine Intelligence 5**, v. 5, p. 135–151, 1970.


THANK YOU