Compressed Term Unification: Results, applications, open problems, and hopes.

Adrià Gascón

Warwick University and The Alan Turing Institute, UK

UNIF'18

Compressed Term Unification: Results, ~~applications~~, open problems, and hopes.

Adrià Gascón

Warwick University and The Alan Turing Institute, UK

UNIF'18

Coauthors & acks

Ashish Tiwari
Guillem Godoy
Sebastian Maneth
Carl Philipp Reh
Lander Ramos

Manfred Schmidt-Schauss
Carles Creus
Markus Lohrey
Kurt Sieber

Temur Kutsia
Jordi Levy
Artur Jez

Mateu Villaret
Albert Rubio

What makes a unification problem?

Term Unification

Solving equations

Solving $s \doteq t$ consists on finding a substitution σ for the variables in s and t such that $\sigma(s) = \sigma(t)$.

Term Unification

Solving equations

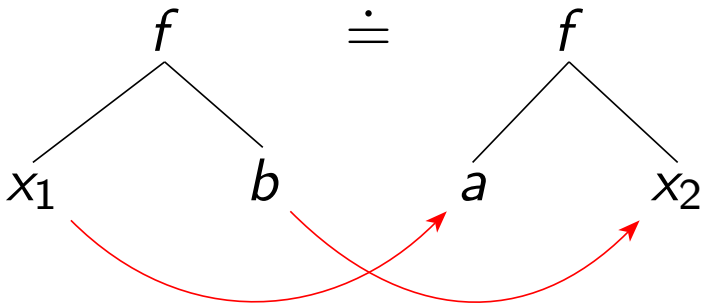
Solving $s \doteq t$ consists on finding a substitution σ for the variables in s and t such that $\sigma(s) = \sigma(t)$.

In the *first-order unification problem*:

- $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}_0)$ are terms with leaf variables $\mathcal{X}_0 = \{x_1, x_2, \ldots\}$ ranging over terms,
- function symbols $\mathcal{F} = \{f, g, h, \ldots\}$ are noninterpreted, and
- $=$ is interpreted as syntactic equality.

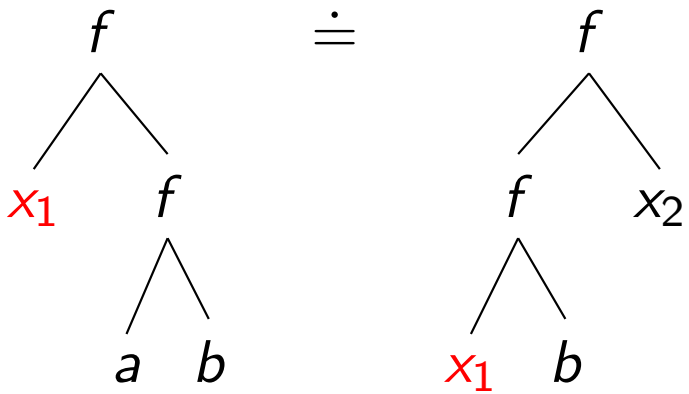
First-order Unification

Example:



$\{x_1 \mapsto a,\ x_2 \mapsto b\}$
is a unifier/solution

Example:



does not have
a solution

Term Unification

Solving equations

Solving $s \doteq t$ consists on finding a substitution σ for the variables in s and t such that $\sigma(s) = \sigma(t)$.

In the *context unification problem*:

- $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}_0 \cup \mathcal{X}_1)$ are terms with leaf variables $\mathcal{X}_0 = \{x_1, x_2, \ldots\}$ ranging over terms and variables of arity one $\mathcal{X}_1 = \{F_1, F_2, \ldots\}$ ranging over contexts,
- function symbols $\mathcal{F} = \{f, g, h, \ldots\}$ are noninterpreted, and
- $=$ is interpreted as syntactic equality.

Context Unification

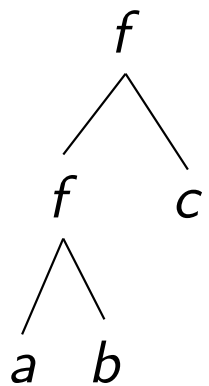
A context is a term with a *single occurrence* of a hole ($\bullet$) in which *terms* may be inserted.

Example (application of a substitution):

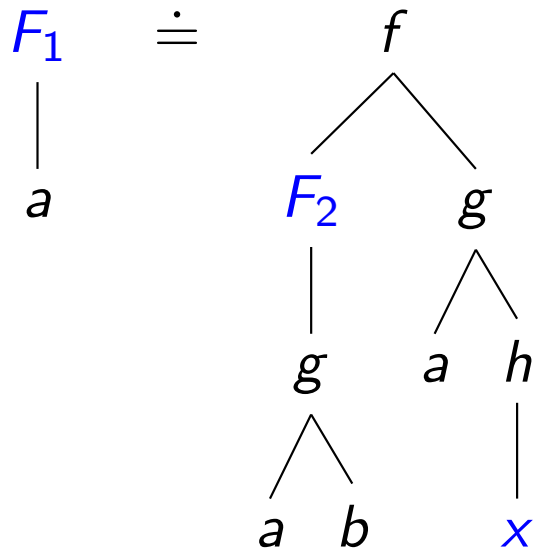
$t =$ 

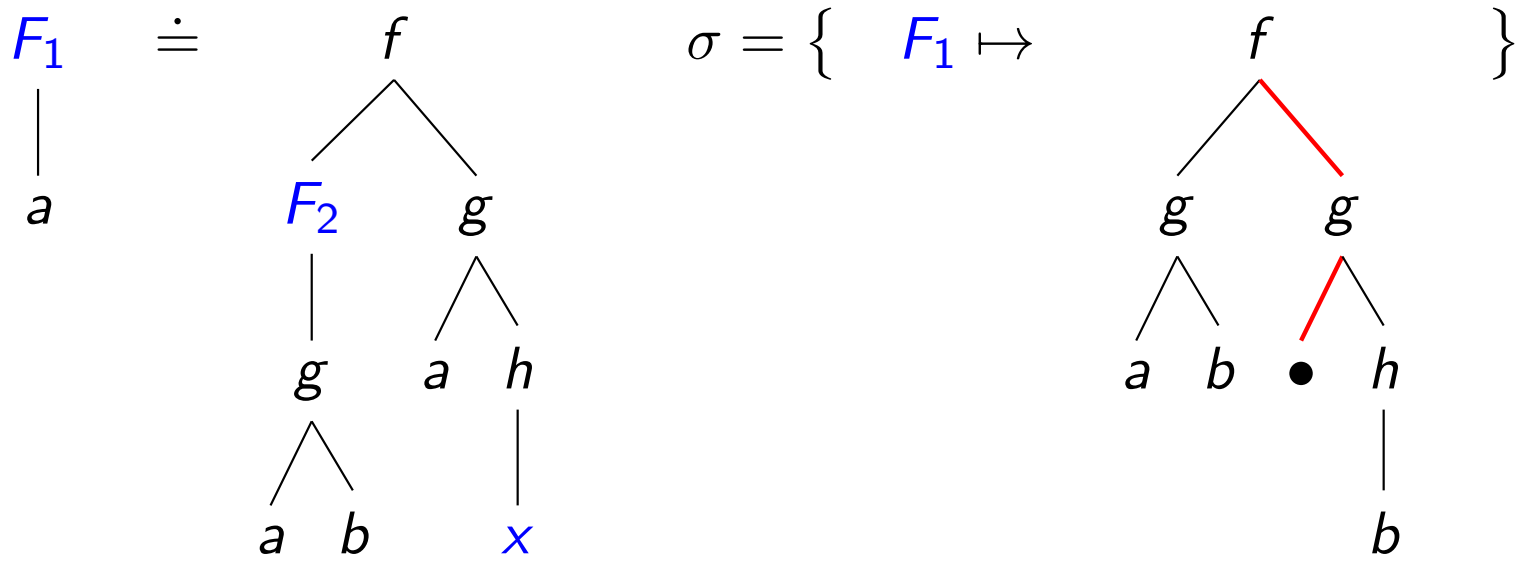
$\sigma = \{ \quad F \mapsto$ 

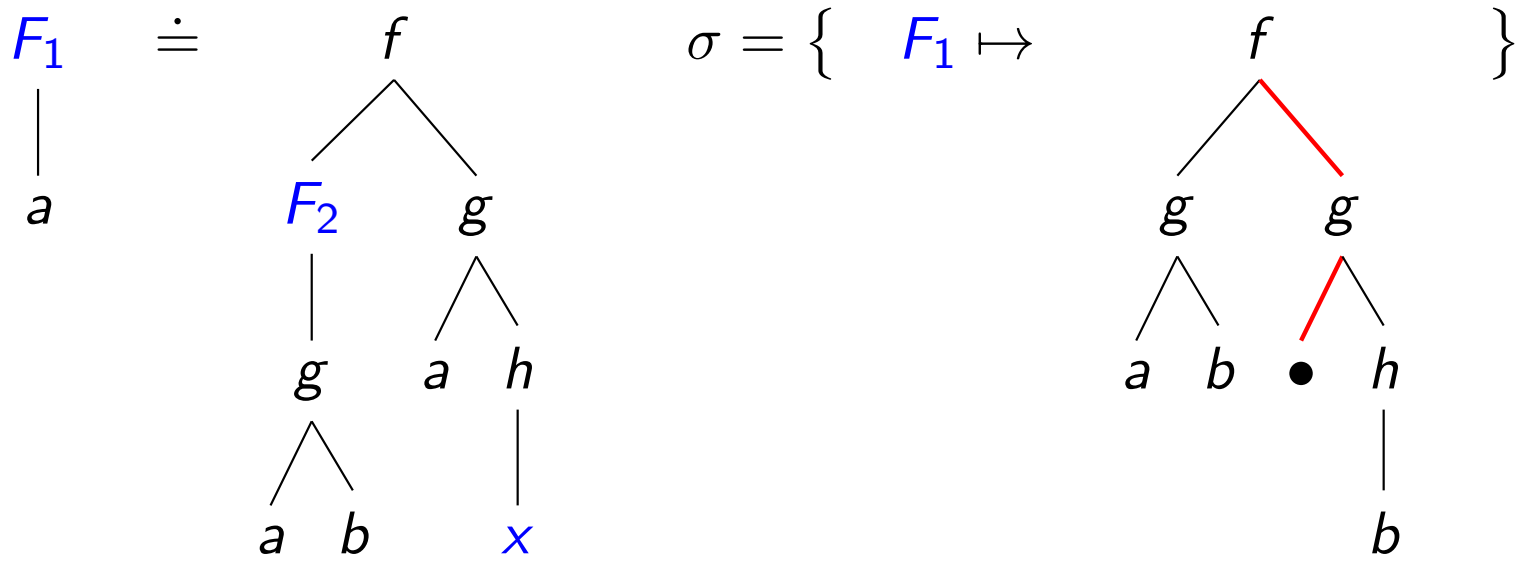
$\}$

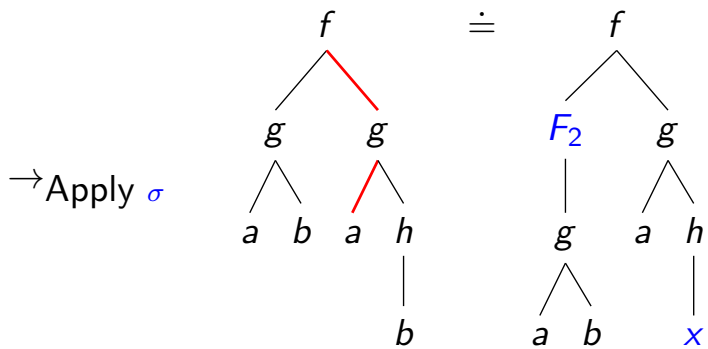
$t\sigma =$ 

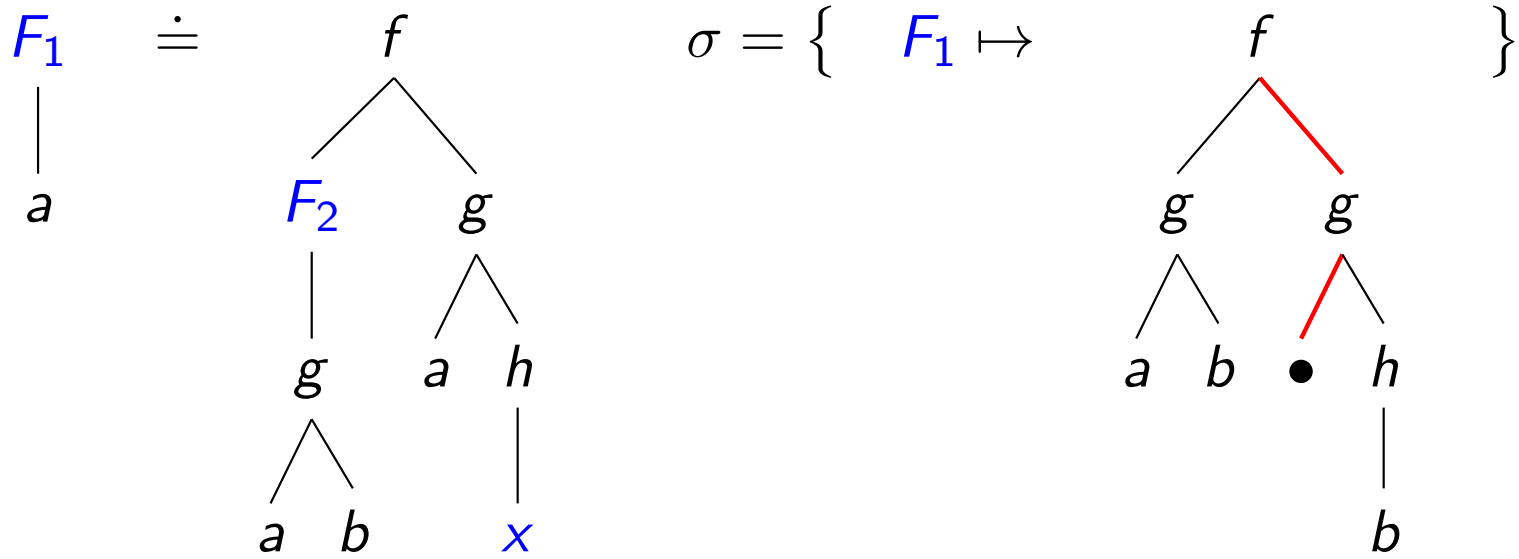
Context unification: an example

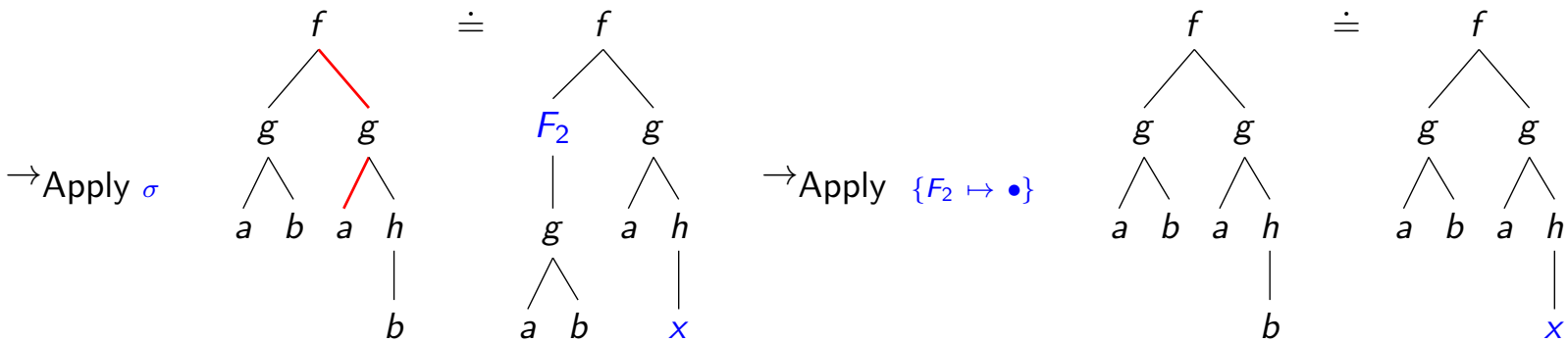



F_1
$\doteq$
f
a
F_2
g
g
a
h
a
b
x
$\sigma = \{$
$F_1 \mapsto$
f
g
g
a
b
$\bullet$
h
b
$\}$

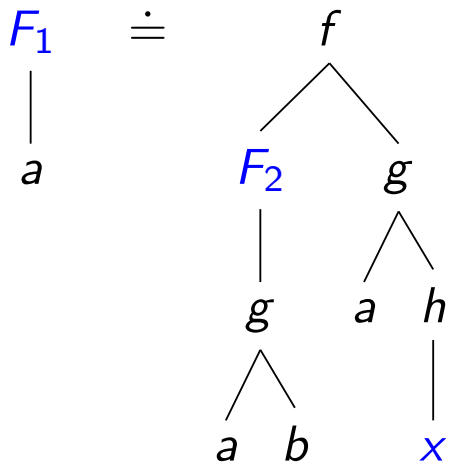

F_1
$\doteq$
f
a
F_2
g
g
a
h
a
b
x
$\sigma = \{$
$F_1 \mapsto$
f
g
g
a
b
$\bullet$
h
b
$\}$


$\rightarrow$Apply σ
f
g
g
a
b
a
h
b
$\doteq$
f
F_2
g
g
a
h
a
b
x

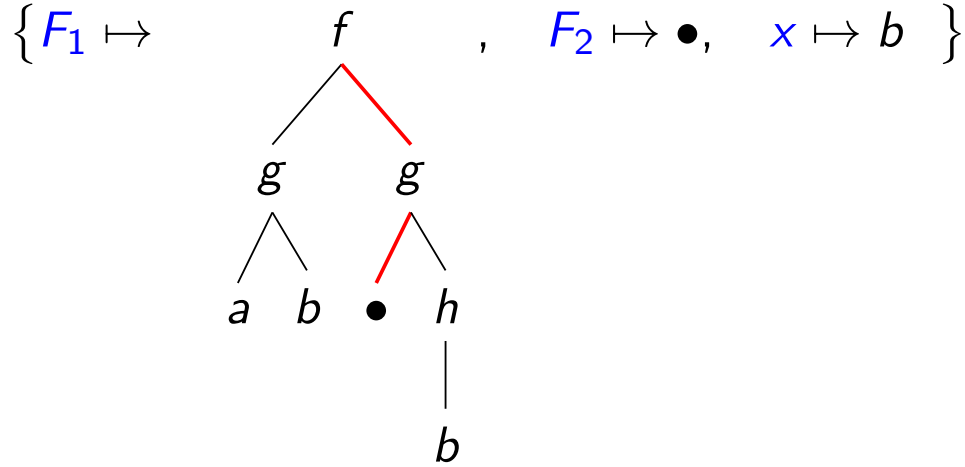

F_1
$\doteq$
f
a
F_2
g
g
a
h
a
b
x
$\sigma = \{$
$F_1 \mapsto$
f
g
g
a
b
$\bullet$
h
b
$\}$


$\rightarrow$Apply σ
f
$\doteq$
f
g
g
F_2
g
a
b
a
h
g
a
h
b
a
b
x
$\rightarrow$Apply $\{F_2 \mapsto \bullet\}$
f
$\doteq$
f
g
g
g
g
a
b
a
h
a
b
a
h
b
x

Instance:



Solution:



Term Matching

Matching equations

Given an equation $s \doteq t$ where *only* s contains variables, find a substitution σ for the variables in s such that $\sigma(s) = t$.

Term Matching

Matching equations

Given an equation $s \doteq t$ where *only* s contains variables, find a substitution σ for the variables in s such that $\sigma(s) = t$.

- First-order unification/matching
- Context unification/matching

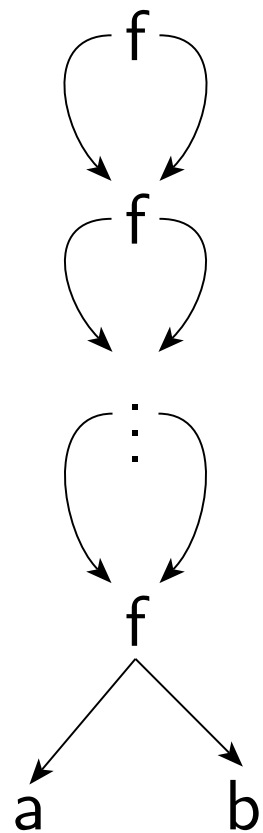
Term Representation

- Explicit
- DAG
- Grammar-based compression (STGs)
- ...

Term representation: DAGs

Definition

A term DAG is an *ordered*, *rooted Directed Acyclic Graph* with nodes labeled with function or variables and the outdegree of every node labeled with a symbol f is equal to the arity of f.



String Compression with SCFGs

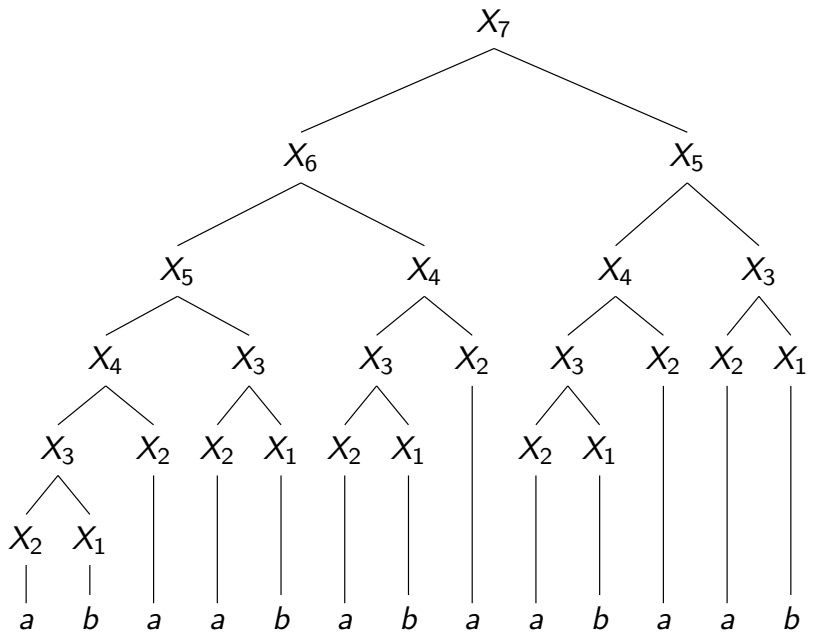
Singleton Context Free Grammars (SCFGs), are essentially *Context Free Grammars (CFGs)* representing just one word. With an SCFG we can succintly represent a word with exponential size.

String Compression with SCFGs

Singleton Context Free Grammars (SCFGs), are essentially *Context Free Grammars (CFGs)* representing just one word. With an SCFG we can succintly represent a word with exponential size.

Example

Consider string $s = abaababaabaab$. It can be generated by the following SCFG: $X_7 \to X_6X_5$, $X_6 \to X_5X_4$, $X_5 \to X_4X_3$, $X_4 \to X_3X_2$, $X_3 \to X_2X_1$, $X_2 \to a$, $X_1 \to b$. We say that X_7 generates s.



Term representation: STGs

Singleton Tree Grammars (STG) are essentially Tree Grammars (TG) representing just one tree. They generalize *Singleton Context Free Grammars (SCFG)* to terms.

STGs: examples

The term $s = f(g(a,a), g(a,h(x)))$ *can be encoded, using STGs, as :*

$$A \to f(A_1, A_2)$$
$$A_1 \to C[A_a]$$
$$C \to g(A_a, \bullet)$$
$$A_a \to a$$
$$A_2 \to C[A_3]$$
$$A_3 \to h(A_x)$$
$$A_x \to x$$

We say that A generates s

With an STG we can succintly represent a term with exponential size and height :

$$A \to C_n[A_a]$$
$$A_a \to a$$
$$C_0 \to f(\bullet)$$
$$C_1 \to C_0[C_0]$$
$$C_2 \to C_1[C_1]$$
$$\vdots$$
$$C_n \to C_{n-1}[C_{n-1}]$$

since A represents the term $f^{2^n}(a)$.

$|t| = 2^n + 1$, $\texttt{height}(t) = 2^n$,

STGs: properties and applications

- Equivalence checking in Ptime [Plandowski], ..., [Lifshits], [Jeż]
- Congruence closure in Ptime [Schmidt-Schauß et al.]
- Pattern submatching [Schmidt-Schauß]
- Membership in several classes of Tree Automata [Lohrey, Maneth], [Lohrey, Maneth et al.]
- STG-compressors available (applications in XML and TRSs) [Bussatto, Maneth], [Lohrey, Maneth et al.], [Bau, Lohrey, et al.]
- Unification theory [Levy, Villaret, Schmidt-Schauß]

Compressed unification

Solving equations

Solving $s \doteq t$ consists on finding a substitution σ for the variables in s and t such that $\sigma(s) = \sigma(t)$.

Term representation

s and t might be given with an explicit, STG, or DAG representation.

Compressed unification

Solving equations

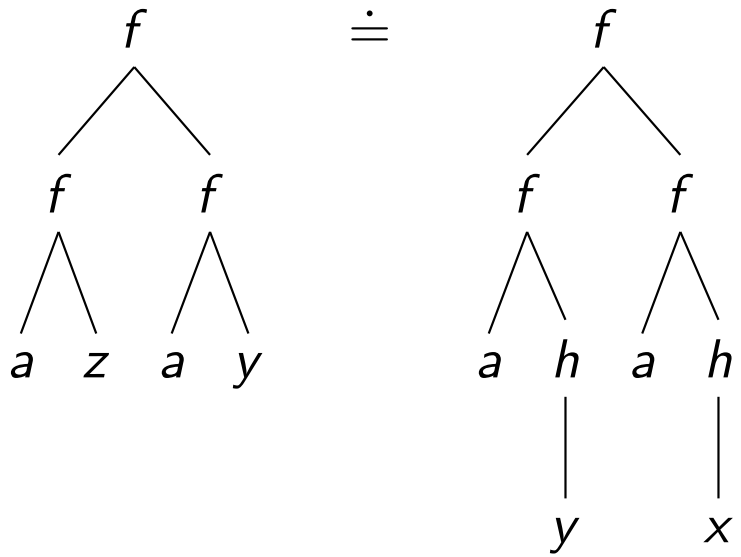
Solving $s \doteq t$ consists on finding a substitution σ for the variables in s and t such that $\sigma(s) = \sigma(t)$.

Term representation

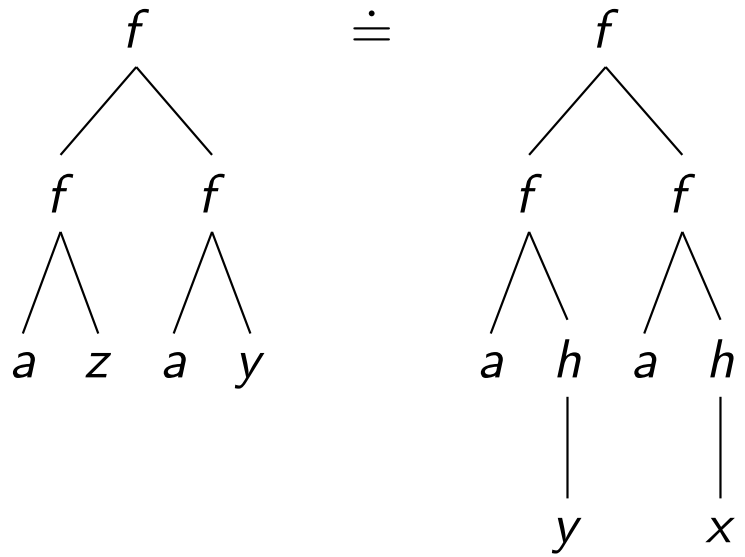
s and t might be given with an explicit, STG, or DAG representation.

- First-order unification/matching with DAGs/STGs/explicit
- Context unification/matching with DAGs/STGs/explicit

Compressed Term Unification



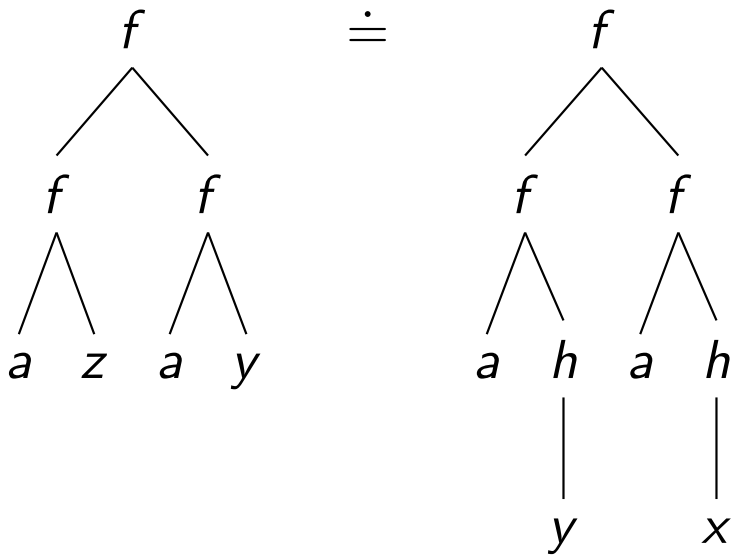
Compressed Term Unification



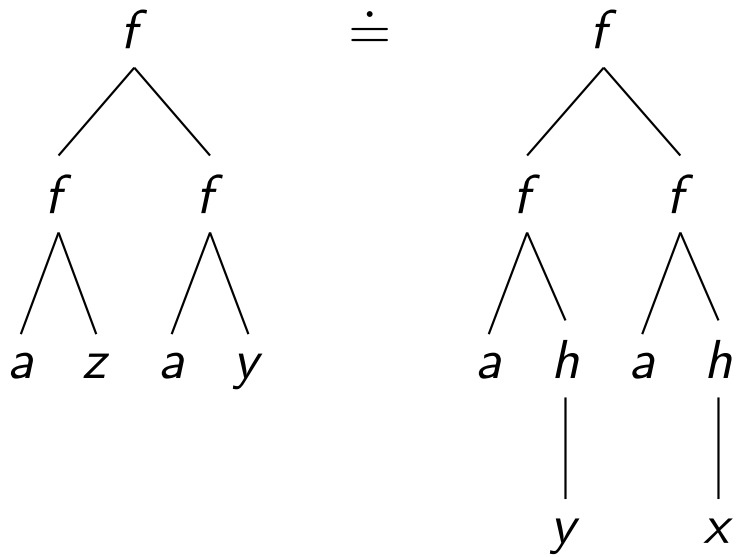
$$A \doteq B$$

DAG $D = \{A \to f(A_1, A_2), A_1 \to f(N_a, N_z), A_2 \to f(N_a, N_y),$
$B \to f(B_1, B_2), B_1 \to f(N_a, B_3), B_3 \to h(N_y),$
$B_2 \to f(N_a, B_4), B_4 \to h(N_x)\}$

Compressed Term Unification



Compressed Term Unification



$$A \doteq B$$

STG $G = \{C_1 \to f(a, \bullet), C_2 \to h(\bullet), A \to f(A_1, A_2),$
$A_1 \to C_1[N_z], A_2 \to C_1[N_y], B \to f(B_1, B_2),$
$B_1 \to C_1[B_3], B_3 \to C_2[N_y], B_2 \to C_1[B_4],$
$B_4 \to C_2[N_x]\}$

Selected Results and Open Problems

Summary of Results

Problem	Term Representation Formalism		
	Explicit	*DAG*	*STG*
Context Unification	PSPACE [J14]	?	?
Context Matching	NP-Complete	NP-Complete	NP-Complete
One-context Unification	NP [GGST10]	NP[CGG12]	NP [CGG12]
Left-linear One-context Unification (and similar variants)	Ptime	Ptime [GTS15]	?
2-*restricted context Unification*	Ptime	Ptime [GST15]	?
k-context Matching	Ptime	Ptime	Ptime [GGS08]
First-order Unification	Ptime	Ptime	Ptime [GGS09, GGS11]
First-order Matching	Ptime	Ptime	Ptime [GGS08]
Equality of unordered ranked trees	Ptime	Ptime	Ptime [LMP15]
Equality of unordered unranked trees	Ptime	Ptime	Ptime [GLMRS18]

Open problem

Is there a NPtime decision procedure for WU? Or is WU PSPACE-hard?

Open problem

Is there a NPtime decision procedure for WU? Or is WU PSPACE-hard?

Open problem

Same question for CU, of course.

One context Unification

An *instance* $\mathcal{I}$ of the 1-CU problem is a set of equations

$$\mathcal{I} = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$$

where $s_i, t_i \in \mathcal{T}(\mathcal{F}, \mathcal{X}_0 \cup \mathcal{X}_1 = \{F\})$.

A solution of $\mathcal{I}$ is a substitution σ such that $\forall i : s_i\sigma = t_i\sigma$.

One context Unification

An *instance* $\mathcal{I}$ of the 1-CU problem is a set of equations

$$\mathcal{I} = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$$

where $s_i, t_i \in \mathcal{T}(\mathcal{F}, \mathcal{X}_0 \cup \mathcal{X}_1 = \{F\})$.

A solution of $\mathcal{I}$ is a substitution σ such that $\forall i : s_i\sigma = t_i\sigma$.

Open problem

Is there a polynomial time decision procedure for 1-CU?

One context Unification

Some of the road has been walked!

One context Unification

Some of the road has been walked!...hopefully

One context Unification

Some of the road has been walked!...hopefully

Definition

An 1-CU instance $\mathcal{I}$ is called *reduced* if it is of the form

$$\{F(u_i) \doteq x_i \mid i = 1, 2, \ldots\} \cup \{F(v_j) \doteq s \mid j = 1, 2, \ldots\} \\ \cup \ \{F(w_k) \doteq t \mid k = 1, 2, \ldots\} \tag{1}$$

where s, t do not contain F; that is, the right hand-side of the equations have at most two non-variable terms.

One context Unification

Some of the road has been walked!...hopefully

Definition

An 1-CU instance $\mathcal{I}$ is called *reduced* if it is of the form

$$\{F(u_i) \doteq x_i \mid i = 1, 2, \ldots\} \cup \{F(v_j) \doteq s \mid j = 1, 2, \ldots\} \\ \cup \ \{F(w_k) \doteq t \mid k = 1, 2, \ldots\} \tag{1}$$

where s, t do not contain F; that is, the right hand-side of the equations have at most two non-variable terms.

Open problem

Is there a polynomial time decision procedure for reduced 1-CU?

The reduction: Subinstances

$$F(r_1) \doteq f(s_1, s_2)$$
$$F(r_2) \doteq f(t_1, t_2)$$
$$F(r_3) \doteq f(u_1, u_2)$$

where $F \notin s_i, t_i, u_i$

$\theta = \{F \mapsto \bullet\}$

$$r_1\theta \doteq f(s_1, s_2)\theta$$
$$r_2\theta \doteq f(t_1, t_2)\theta$$
$$r_3\theta \doteq f(u_1, u_2)\theta$$

FO instance

$\sigma_1 = \mathtt{mgu}(s_2, t_2, u_2)$
$F \mapsto f(F'(\bullet), s_2\sigma_1)$

$$F'(r_1\sigma_1) \doteq s_1\sigma_1$$
$$F'(r_2\sigma_1) \doteq t_1\sigma_1$$
$$F'(r_3\sigma_1) \doteq u_1\sigma_1$$

1-CU instance

$\sigma_2 = \mathtt{mgu}(s_1, t_1, u_1)$
$F \mapsto f(s_1\sigma_2, F'(\bullet))$

$$F'(r_1\sigma_2) \doteq s_2\sigma_2$$
$$F'(r_2\sigma_2) \doteq t_2\sigma_2$$
$$F'(r_3\sigma_2) \doteq u_2\sigma_2$$

1-CU instance

The reduction: solve subinstances together

$$\begin{array}{l} F(r_1) \doteq f(s_1, s_2) \\ F(r_2) \doteq f(t_1, t_2) \\ F(r_3) \doteq f(u_1, u_2) \end{array} \qquad \text{where } F \notin s_i, t_i, u_i$$

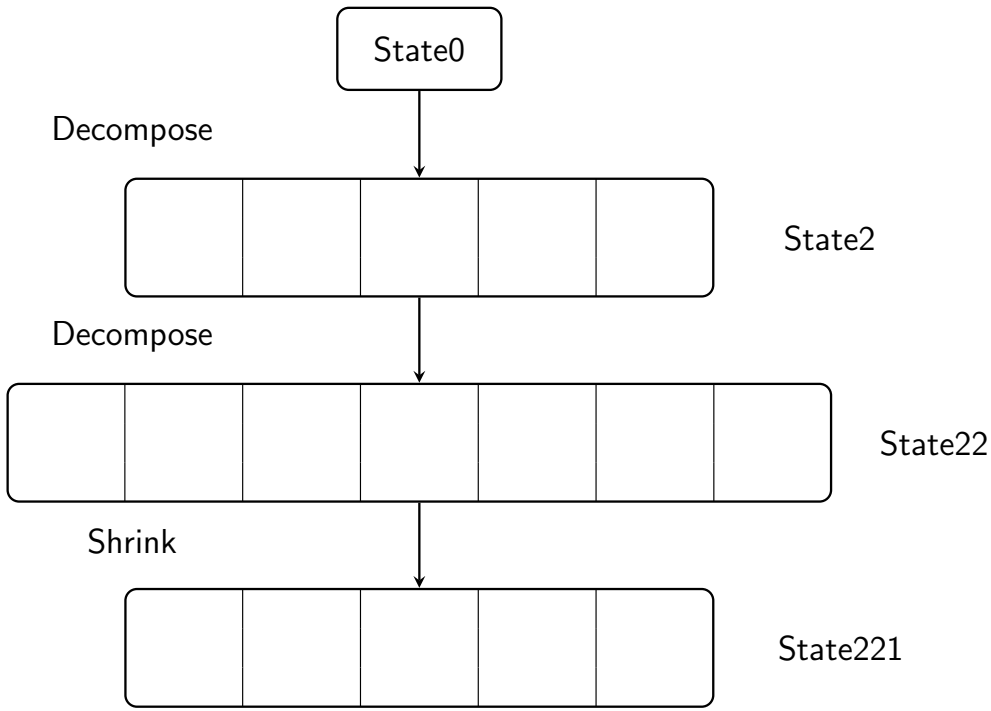
$$S_0 = (\ \langle r_1, r_2, r_3 \rangle \ \ , \ \ \{ f(s_1, s_2) \doteq f(t_1, t_2) \doteq f(u_1, u_2) \}\ )$$

$\Big\downarrow$ **Check** and **Decompose**

$$S_1 = (\ \langle r_1, r_2, r_3 \rangle \ \ , \ \ \{\ s_1 \doteq t_1 \doteq u_1\ ,\ s_2 \doteq t_2 \doteq u_2\ \}\ )$$

Shrinking rules

$$\text{Strategy}: \quad (\texttt{Shrink} - \texttt{i}^{!} \cdot \texttt{Decompose})^{!}$$



Shrinking rules: An observation

$$F(r_1) \doteq f(\mathbf{s}, \mathbf{s})$$
$$F(r_2) \doteq f(\mathbf{t}, \mathbf{t})$$
$$F(r_3) \doteq f(u_1, u_2)$$

where $F \notin s, t, u_i$

$\theta = \{F \mapsto \bullet\}$

$$r_1\theta \doteq f(s, s)\theta$$
$$r_2\theta \doteq f(t, t)\theta$$
$$r_3\theta \doteq f(u_1, u_2)\theta$$

FO instance

$\sigma_1 = \mathtt{mgu}(\mathbf{s}, \mathbf{t}, u_2)$
$F \mapsto f(F'(\bullet), s_2\sigma_1)$

$$F'(r_1\sigma_1) \doteq s\sigma_1$$
$$F'(r_2\sigma_1) \doteq t\sigma_1$$
$$F'(r_3\sigma_1) \doteq u_1\sigma_1$$

1-CU instance

$\sigma_2 = \mathtt{mgu}(\mathbf{s}, \mathbf{t}, u_1)$
$F \mapsto f(s_1\sigma_2, F'(\bullet))$

$$F'(r_1\sigma_2) \doteq s\sigma_2$$
$$F'(r_2\sigma_2) \doteq t\sigma_2$$
$$F'(r_3\sigma_2) \doteq u_2\sigma_2$$

1-CU instance

Shrinking rules: An observation

We can apply $\mathtt{mgu}(s, t)$ **eagerly**:

$$F(r_1) \doteq f(\mathbf{s}, \mathbf{s})$$
$$F(r_2) \doteq f(\mathbf{t}, \mathbf{t})$$
$$F(r_3) \doteq f(u_1, u_2)$$

$\theta = \{F \mapsto \bullet\}$

$\sigma = \mathtt{mgu}(\mathbf{s}, \mathbf{t})$

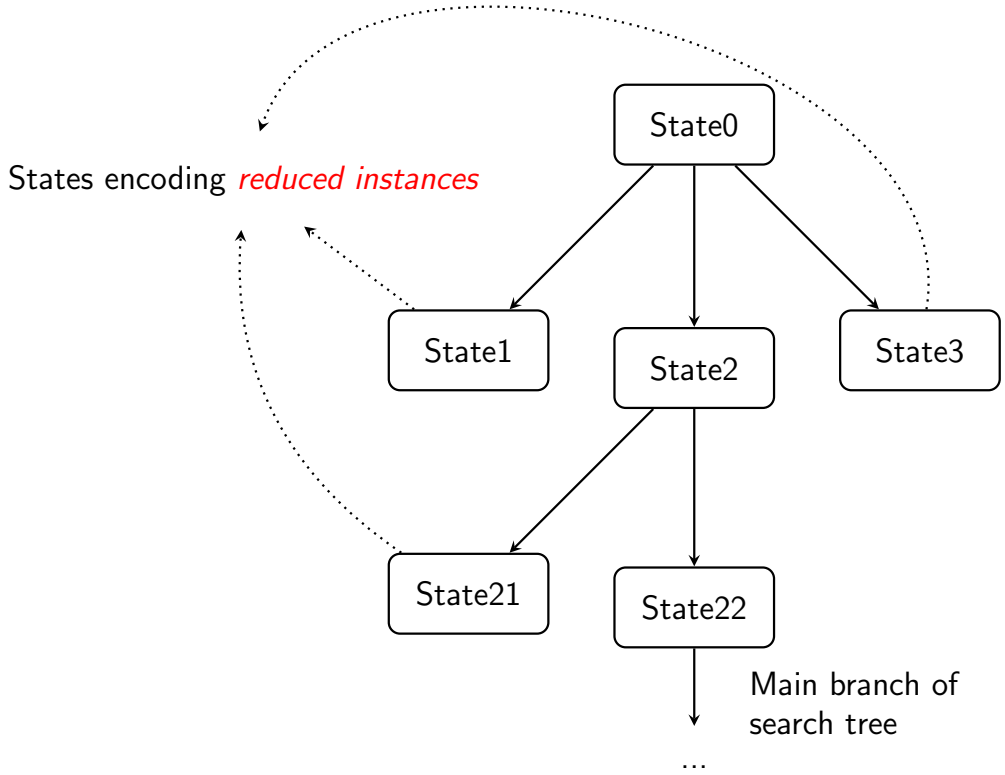
$$r_1\theta \doteq f(s, s)\theta$$
$$r_2\theta \doteq f(t, t)\theta$$
$$r_3\theta \doteq f(u_1, u_2)\theta$$

FO instance

$$F(r_1\sigma) \doteq f(\mathbf{s}\sigma, \mathbf{s}\sigma)$$
$$F(r_2\sigma) \doteq f(\mathbf{t}\sigma, \mathbf{t}\sigma)$$
$$F(r_3\sigma) \doteq f(u_1\sigma, u_2\sigma)$$

1-CU instance

Sometimes guessing is unavoidable



- We in fact give a Turing reduction from 1-CU to *reduced 1-CU*.
- PTIME of some reduced instances relies on our CSL'15 result.

Size of a state: A riddle

Given a set P_r of n *red* nodes, let $G(P_r \uplus P_{bg}, E)$ be a bipartite graph, where P_{bg} is a set of *blue and green nodes*, such that:

(a) G is acyclic, and
(b) each red node has 3 neighbours, and *there is a path from each neighbour to a blue node*.

Question: Find an upper bound $B(n)$ on the size of G.

Summary of Results

Problem	Term Representation Formalism		
	Explicit	*DAG*	*STG*
Context Unification	PSPACE [J14]	?	?
Context Matching	NP-Complete	NP-Complete	NP-Complete
One-context Unification	NP [GGST10]	NP[CGG12]	NP [CGG12]
Left-linear One-context Unification (and similar variants)	Ptime	Ptime [GTS15]	?
2-*restricted context Unification*	Ptime	Ptime [GST15]	?
k-context Matching	Ptime	Ptime	Ptime [GGS08]
First-order Unification	Ptime	Ptime	Ptime [GGS09, GGS11]
First-order Matching	Ptime	Ptime	Ptime [GGS08]
Equality of unordered ranked trees	Ptime	Ptime	Ptime [LMP15]
Equality of unordered unranked trees	Ptime	Ptime	Ptime [GLMRS18]

Comparing compressed terms

Definition

Two-restricted One context Unification (1CU2r) is 1-CU with at most 2 occurrences of F.

Comparing compressed terms

Definition

Two-restricted One context Unification (1CU2r) is 1-CU with at most 2 occurrences of F.

Open problem

Is there a polynomial time decision procedure for 1CU2r over *STG-compressed* terms?

Comparing compressed terms

Definition

Two-restricted One context Unification (1CU2r) is 1-CU with at most 2 occurrences of F.

Open problem

Is there a polynomial time decision procedure for 1CU2r over *STG-compressed* terms?

But in fact subcases are open...

Comparing compressed terms

Definition

Two-restricted One context Unification (1CU2r) is 1-CU with at most 2 occurrences of F.

Open problem

Is there a polynomial time decision procedure for 1CU2r over *STG-compressed* terms?

But in fact subcases are open...

Comparing compressed terms

Open problem

Is there a Ptime procedure for STG-compressed submatching (of nonlinear terms s in t):

$$F(s) = t$$

with $F \notin s, t$.

- This is related with finding a redex of a term rewriting rule in a term [SS2013].

and on a similar note...

Comparing compressed terms

Open problem

Let s and t be STG-compressed terms, and let $<$ be a KBO ordering on terms, and thus parameterized by a weight function and a precedence order on the signature of s and t. Is there a polynomial time decision procedure for $s < t$?

More compression!: non-linear STGs

STGs: examples

The term $s = f(g(a, a), g(a, h(x)))$ *can be encoded, using STGs, as :*

$$A \to f(A_1, A_2)$$
$$A_1 \to C[A_a]$$
$$C \to g(A_a, \bullet)$$
$$A_a \to a$$
$$A_2 \to C[A_3]$$
$$A_3 \to h(A_x)$$
$$A_x \to x$$

We say that A generates s

$$A \to C_n[A_a]$$
$$A_a \to a$$
$$C_0 \to f(\bullet)$$
$$C_1 \to C_0[C_0]$$
$$C_2 \to C_1[C_1]$$
$$\vdots$$
$$C_n \to C_{n-1}[C_{n-1}]$$

since A represents the term $f^{2^n}(a)$.

$|t| = 2^n + 1$, $\texttt{height}(t) = 2^n$,

Open problem

Is there a non-deterministic polynomial time decision procedure for equivalence checking of nonlinear STGs?

Open problem

Is there a non-deterministic polynomial time decision procedure for equivalence checking of nonlinear STGs?

Best know upper bound is PSPACE. My guess is that proving co-NP is "easy"... but hardness is a different story...

And equational theories!

Lemma

STG-Compressed Ranked & Ordered Tree isomorphism is in Ptime.

Lemma

STG-Compressed Ranked & UnOrdered Tree isomorphism is in Ptime.

Lemma

STG-Compressed Unranked & Ordered (forests) Tree isomorphism is in Ptime.

Open problem

We can do STG-Compressed Tree isomorphism module AC in Ptime. Can this be extended to idempotence?

We think the answer is yes. On the other hand adding adding distributivity leads to the polynomial identity testing problem.

Conclusion

Lots of super hard innocent looking problems in compressed unification :)

Conclusion

Lots of super hard innocent looking problems in compressed unification :)
But they can be solved!

Thanks for your attention!

Size of a state

Given a set P_r of n *red* nodes, let $G(P_r \uplus P_{bg}, E)$ be a bipartite graph, where P_{bg} is a set of *blue and green nodes*, such that:

(a) G is acyclic, and
(b) each red node has 3 neighbours, and *there is a path from each neighbour to a blue node*.

Question: Find an upper bound $B(n)$ on the size of G.