

# Introduction to Unification Theory

## Equational Unification

Temur Kutsia

RISC, Johannes Kepler University of Linz, Austria  
kutsia@risc.jku.at

# Overview

Motivation

Equational Theories, Reformulations of Notions

Unification Type, Kinds of Unification

Results for Specific Theories

General Results

# Outline

## Motivation

Equational Theories, Reformulations of Notions

Unification Type, Kinds of Unification

Results for Specific Theories

General Results

# Motivation

- ▶ Unifications algorithms are essential components for deduction systems.
- ▶ Simple integration of axioms that describe the properties of equality often leads to an unacceptable increase of search space.
- ▶ Proposed solution: To build equational axioms into inference, replacing syntactic unification with equational unification.



# Motivation

## Example

Given: AI-theory  $\{f(f(x, y), z) \approx f(x, f(y, z)), f(x, x) \approx x\}$ . Apply idempotence to the term

$$f(x_0, f(x_1, \dots, f(x_{n-1}, f(x_n, f(x_0, \dots, f(x_{n-1}, x_n) \dots)))) \dots)).$$



# Motivation

## Example

Given: AI-theory  $\{f(f(x, y), z) \approx f(x, f(y, z)), f(x, x) \approx x\}$ . Apply idempotence to the term

$$f(x_0, f(x_1, \dots, f(x_{n-1}, f(x_n, f(x_0, \dots, f(x_{n-1}, x_n) \dots)))) \dots)).$$

- ▶ Exponentially many ways of rearranging the parentheses with the help of associativity: Very time consuming if the prover has to search for the right one.



# Motivation

## Example

Given: AI-theory  $\{f(f(x, y), z) \approx f(x, f(y, z)), f(x, x) \approx x\}$ . Apply idempotence to the term

$$f(x_0, f(x_1, \dots, f(x_{n-1}, f(x_n, f(x_0, \dots, f(x_{n-1}, x_n) \dots)))) \dots)).$$

- ▶ Exponentially many ways of rearranging the parentheses with the help of associativity: Very time consuming if the prover has to search for the right one.
- ▶ A human mathematician would use words instead of terms, i.e. would work modulo associativity, and apply idempotence  $xx = x$  to the word  $x_0 \cdots x_n x_0 \cdots x_n$  by unifying  $x$  with  $x_0 \cdots x_n$ .



# Motivation

## Example

Given: AI-theory  $\{f(f(x, y), z) \approx f(x, f(y, z)), f(x, x) \approx x\}$ . Apply idempotence to the term

$$f(x_0, f(x_1, \dots, f(x_{n-1}, f(x_n, f(x_0, \dots, f(x_{n-1}, x_n) \dots)))) \dots)).$$

- ▶ Exponentially many ways of rearranging the parentheses with the help of associativity: Very time consuming if the prover has to search for the right one.
- ▶ A human mathematician would use words instead of terms, i.e. would work modulo associativity, and apply idempotence  $xx = x$  to the word  $x_0 \cdots x_n x_0 \cdots x_n$  by unifying  $x$  with  $x_0 \cdots x_n$ .
- ▶ To adopt this way of proceeding for a prover, we must replace the syntactic unification algorithm in the resolution step by associative unification.





# Outline

Motivation

Equational Theories, Reformulations of Notions

Unification Type, Kinds of Unification

Results for Specific Theories

General Results

# Equational Theory

## Equational Theory

- ▶  $E$ : a set of equations over  $\mathcal{T}(\mathcal{F}, \mathcal{V})$ , called identities.
- ▶ Equational theory  $\doteq_E$  defined by  $E$ : The least congruence relation on  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  closed under substitution and containing  $E$



# Equational Theory

## Equational Theory

- ▶  $E$ : a set of equations over  $\mathcal{T}(\mathcal{F}, \mathcal{V})$ , called identities.
- ▶ Equational theory  $\doteq_E$  defined by  $E$ : The least congruence relation on  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  closed under substitution and containing  $E$   
i.e.,  $\doteq_E$  is the least binary relation on  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  with the properties:
  - ▶  $E \subseteq \doteq_E$ .
  - ▶ Reflexivity:  $s \doteq_E s$  for all  $s$ .
  - ▶ Symmetry: If  $s \doteq_E t$  then  $t \doteq_E s$  for all  $s, t$ .
  - ▶ Transitivity: If  $s \doteq_E t$  and  $t \doteq_E r$  then  $s \doteq_E r$  for all  $s, t, r$ .
  - ▶ Congruence: If  $s_1 \doteq_E t_1, \dots, s_n \doteq_E t_n$  then  $f(s_1, \dots, s_n) \doteq_E f(t_1, \dots, t_n)$  for all  $s, t, n$  and  $n$ -ary  $f$ .
  - ▶ Closure under substitution: If  $s \doteq_E t$  then  $s\sigma \doteq_E t\sigma$  for all  $s, t, \sigma$ .



# Notation, Terminology

- ▶ Identities:  $s \approx t$ .
- ▶  $s \doteq_E t$ : The term  $s$  is equal modulo  $E$  to the term  $t$ .
- ▶  $E$  will be called an equational theory as well (abuse of the terminology).
- ▶  $\text{sig}(E)$ : The set of function symbols that occur in  $E$ .

## Example

- ▶  $C := \{f(x, y) \approx f(y, x)\}$ :  $f$  is commutative.  $\text{sig}(C) = f$ .
- ▶  $f(f(a, b), c) \doteq_C f(c, f(b, a))$ .



# Notation, Terminology

- ▶ Identities:  $s \approx t$ .
- ▶  $s \doteq_E t$ : The term  $s$  is equal modulo  $E$  to the term  $t$ .
- ▶  $E$  will be called an equational theory as well (abuse of the terminology).
- ▶  $\text{sig}(E)$ : The set of function symbols that occur in  $E$ .

## Example

- ▶  $C := \{f(x, y) \approx f(y, x)\}$ :  $f$  is commutative.  $\text{sig}(C) = f$ .
- ▶  $f(f(a, b), c) \doteq_C f(c, f(b, a))$ .
- ▶  $AU := \{f(f(x, y), z) \approx f(x, f(y, z)), f(x, e) \approx x, f(e, x) \approx x\}$ :  
 $f$  is associative,  $e$  is unit.  $\text{sig}(AU) = \{f, e\}$
- ▶  $f(a, f(x, f(e, a))) \doteq_{AU} f(f(a, x), a)$ .



# Notation, Terminology

## *E*-Unification Problem, *E*-Unifier, *E*-Unifiability

- ▶ *E*: equational theory.  
 $\mathcal{F}$ : set of function symbols.  
 $\mathcal{V}$ : countable set of variables.
- ▶ *E*-Unification problem over  $\mathcal{F}$ : a finite set of equations

$$\Gamma = \{s_1 \doteq_E^? t_1, \dots, s_n \doteq_E^? t_n\},$$

where  $s_i, t_i \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ .

- ▶ *E*-Unifier of  $\Gamma$ : a substitution  $\sigma$  such that

$$s_1\sigma \doteq_E t_1\sigma, \dots, s_n\sigma \doteq_E t_n\sigma.$$

- ▶  $u_E(\Gamma)$ : the set of *E*-unifiers of  $\Gamma$ .  $\Gamma$  is *E*-unifiable iff  $u_E(\Gamma) \neq \emptyset$ .



# $E$ -Unification vs Syntactic Unification

- ▶ Syntactic unification: a special case of  $E$ -unif. with  $E = \emptyset$ .
- ▶ Any syntactic unifier of an  $E$ -unification problem  $\Gamma$  is also an  $E$ -unifier of  $\Gamma$ .
- ▶ For  $E \neq \emptyset$ ,  $u_E(\Gamma)$  may contain a unifier that is not a syntactic unifier.



# $E$ -Unification vs Syntactic Unification

- ▶ Syntactic unification: a special case of  $E$ -unif. with  $E = \emptyset$ .
- ▶ Any syntactic unifier of an  $E$ -unification problem  $\Gamma$  is also an  $E$ -unifier of  $\Gamma$ .
- ▶ For  $E \neq \emptyset$ ,  $u_E(\Gamma)$  may contain a unifier that is not a syntactic unifier.

## Example

- ▶ Terms  $f(a, x)$  and  $f(b, y)$ :
  - ▶ Not syntactically unifiable.
  - ▶ Unifiable module commutativity of  $f$ .  
 $C$ -unifier:  $\{x \mapsto b, y \mapsto a\}$





# $E$ -Unification vs Syntactic Unification

- ▶ Syntactic unification: a special case of  $E$ -unif. with  $E = \emptyset$ .
- ▶ Any syntactic unifier of an  $E$ -unification problem  $\Gamma$  is also an  $E$ -unifier of  $\Gamma$ .
- ▶ For  $E \neq \emptyset$ ,  $u_E(\Gamma)$  may contain a unifier that is not a syntactic unifier.

## Example

- ▶ Terms  $f(a, x)$  and  $f(b, y)$ :
  - ▶ Not syntactically unifiable.
  - ▶ Unifiable module commutativity of  $f$ .  
 $C$ -unifier:  $\{x \mapsto b, y \mapsto a\}$
- ▶ Terms  $f(a, x)$  and  $f(y, b)$ :
  - ▶ Have the most general syntactic unifier  $\{x \mapsto b, y \mapsto a\}$ .
  - ▶ If  $f$  is associative, then  $u_A(\{f(a, x) \stackrel{?}{\doteq}_A f(y, b)\})$  contains additional  $A$ -unifiers, e.g.  $\{x \mapsto f(z, b), y \mapsto f(a, z)\}$ .



# Notions Adapted

## Instantiation Quasi-Ordering (Modified)

- ▶  $E$ : equational theory.  $\mathcal{X}$ : set of variables.
- ▶ A substitution  $\sigma$  is *more general modulo  $E$  on  $\mathcal{X}$*  than  $\vartheta$ , written  $\sigma \leq_E^{\mathcal{X}} \vartheta$ , if there exists  $\eta$  such that  $x\sigma\eta \doteq_E x\vartheta$  for all  $x \in \mathcal{X}$ .
- ▶  $\vartheta$  is called an  *$E$ -instance* of  $\sigma$  modulo  $E$  on  $\mathcal{X}$ .
- ▶ The relation  $\leq_E^{\mathcal{X}}$  is quasi-ordering, called *instantiation quasi-ordering*.
- ▶  $\equiv_E^{\mathcal{X}}$  is the equivalence relation corresponding to  $\leq_E^{\mathcal{X}}$ .



# No Single MGU

- ▶ When comparing unifiers of  $\Gamma$ , the set  $\mathcal{X}$  is  $vars(\Gamma)$ .
- ▶ Unifiable  $E$ -unification problems might not have an mgu.

## Example

- ▶  $f$  is commutative.
- ▶  $\Gamma = \{f(x, y) \stackrel{?}{\doteq}_C f(a, b)\}$  has two  $C$ -unifiers:

$$\sigma_1 = \{x \mapsto a, y \mapsto b\}$$

$$\sigma_2 = \{x \mapsto b, y \mapsto a\}.$$

- ▶ On  $vars(\Gamma) = \{x, y\}$ , any unifier is equal to either  $\sigma_1$  or  $\sigma_2$ .
- ▶  $\sigma_1$  and  $\sigma_2$  are not comparable wrt  $\leq_C^{\{x,y\}}$ .
- ▶ Hence, no mgu for  $\Gamma$ .



# MCSU vs MGU

In  $E$ -unification, the role of mgu is taken on by a complete set of  $E$ -unifiers.

## Complete and Minimal Complete Sets of $E$ -Unifiers

- ▶  $\Gamma$ :  $E$ -unification problem over  $\mathcal{F}$ .
- ▶  $\mathcal{X} = \text{vars}(\Gamma)$ .
- ▶  $\mathcal{C}$  is a *complete set of  $E$ -unifiers* of  $\Gamma$  iff
  1.  $\mathcal{C} \subseteq u_E(\Gamma)$ :  $\mathcal{C}$ 's elements are  $E$ -unifiers of  $\Gamma$ , and
  2. For each  $\vartheta \in u_E(\Gamma)$  there exists  $\sigma \in \mathcal{C}$  such that  $\sigma \leq_E^{\mathcal{X}} \vartheta$ .
- ▶  $\mathcal{C}$  is a *minimal complete set of  $E$ -unifiers* ( $mcsu_E$ ) of  $\Gamma$  if it is a complete set of  $E$ -unifiers of  $\Gamma$  and
  3. two distinct elements of  $\mathcal{C}$  are not comparable wrt  $\leq_E^{\mathcal{X}}$ .
- ▶  $\sigma$  is an mgu of  $\Gamma$  iff  $mcsu_E(\Gamma) = \{\sigma\}$ .



# MCSU's

- ▶  $mcsu_E(\Gamma) = \emptyset$  if  $\Gamma$  is not  $E$ -unifiable.
- ▶ Minimal complete sets of unifiers do not always exist.
- ▶ When they exist, they may be infinite.
- ▶ When they exist, they are unique up to  $\equiv_E$ .



# Outline

Motivation

Equational Theories, Reformulations of Notions

**Unification Type, Kinds of Unification**

Results for Specific Theories

General Results

# Unification Type

## Unification Type of a Problem, Theory.

- ▶  $E$ : equational theory.
- ▶  $\Gamma$ :  $E$ -unification problem over  $\mathcal{F}$ .
- ▶  $\Gamma$  has *unification type*
  - ▶ *unitary*, if  $mcsu(\Gamma)$  has cardinality at most one,
  - ▶ *finitary*, if  $mcsu(\Gamma)$  has finite cardinality,
  - ▶ *infinitary*, if  $mcsu(\Gamma)$  has infinite cardinality,
  - ▶ *zero*, if  $mcsu(\Gamma)$  does not exist.
- ▶ Abbreviation: type unitary - 1, finitary -  $\omega$ , infinitary -  $\infty$ , zero - 0.
- ▶ Ordering:  $1 < \omega < \infty < 0$ .
- ▶ *Unification type* of  $E$  wrt  $\mathcal{F}$ : the maximal type of an  $E$ -unification problem over  $\mathcal{F}$ .



# Unification Type

The unification type of an  $E$ -equational problem over  $\mathcal{F}$  depends both

- ▶ on  $E$ , and
- ▶ on  $\mathcal{F}$ .

Examples and more details will follow.





# Unification Type

## Example (Type Unitary)

Syntactic unification.

- ▶ The empty equational theory  $\emptyset$ : Syntactic unification.
- ▶ Unitary wrt any  $\mathcal{F}$  because any unifiable syntactic unification problem has an mgu.



# Unification Type

## Example (Type Finitary)

Commutative unification:  $\{f(x, y) \approx f(y, x)\}$

- ▶  $\{f(x, y) \stackrel{?}{\underset{C}{\doteq}} f(a, b)\}$  does not have an mgu. C-unification is not unitary.
- ▶ Show that it is finitary for any  $\mathcal{F}$ :



# Unification Type

## Example (Type Finitary)

Commutative unification:  $\{f(x, y) \approx f(y, x)\}$

- ▶  $\{f(x, y) \stackrel{?}{\dot{=}}_C f(a, b)\}$  does not have an mgu.  $C$ -unification is not unitary.
- ▶ Show that it is finitary for any  $\mathcal{F}$ :
  - ▶ Let  $\Gamma = \{s_1 \stackrel{?}{\dot{=}}_C t_1, \dots, s_n \stackrel{?}{\dot{=}}_C t_n\}$  be a  $C$ -unification problem.



# Unification Type

## Example (Type Finitary)

Commutative unification:  $\{f(x, y) \approx f(y, x)\}$

- ▶  $\{f(x, y) \doteq_C^? f(a, b)\}$  does not have an mgu.  $C$ -unification is not unitary.
- ▶ Show that it is finitary for any  $\mathcal{F}$ :
  - ▶ Let  $\Gamma = \{s_1 \doteq_C^? t_1, \dots, s_n \doteq_C^? t_n\}$  be a  $C$ -unification problem.
  - ▶ Consider all possible syntactic unification problems  $\Gamma' = \{s'_1 \doteq^? t'_1, \dots, s'_n \doteq^? t'_n\}$ , where  $s'_i \doteq_C s_i$  and  $t'_i \doteq_C t_i$  for each  $1 \leq i \leq n$ .



# Unification Type

## Example (Type Finitary)

Commutative unification:  $\{f(x, y) \approx f(y, x)\}$

- ▶  $\{f(x, y) \doteq_C^? f(a, b)\}$  does not have an mgu.  $C$ -unification is not unitary.
- ▶ Show that it is finitary for any  $\mathcal{F}$ :
  - ▶ Let  $\Gamma = \{s_1 \doteq_C^? t_1, \dots, s_n \doteq_C^? t_n\}$  be a  $C$ -unification problem.
  - ▶ Consider all possible syntactic unification problems  $\Gamma' = \{s'_1 \doteq^? t'_1, \dots, s'_n \doteq^? t'_n\}$ , where  $s'_i \doteq_C s_i$  and  $t'_i \doteq_C t_i$  for each  $1 \leq i \leq n$ .
  - ▶ There are only finitely many such  $\Gamma'$ 's, because the  $C$ -equivalence class for a given term  $t$  is finite.



# Unification Type

## Example (Type Finitary)

Commutative unification:  $\{f(x, y) \approx f(y, x)\}$

- ▶  $\{f(x, y) \doteq_C^? f(a, b)\}$  does not have an mgu.  $C$ -unification is not unitary.
- ▶ Show that it is finitary for any  $\mathcal{F}$ :
  - ▶ Let  $\Gamma = \{s_1 \doteq_C^? t_1, \dots, s_n \doteq_C^? t_n\}$  be a  $C$ -unification problem.
  - ▶ Consider all possible syntactic unification problems  $\Gamma' = \{s'_1 \doteq^? t'_1, \dots, s'_n \doteq^? t'_n\}$ , where  $s'_i \doteq_C s_i$  and  $t'_i \doteq_C t_i$  for each  $1 \leq i \leq n$ .
  - ▶ There are only finitely many such  $\Gamma'$ 's, because the  $C$ -equivalence class for a given term  $t$  is finite.
  - ▶ It can be shown that collection of all mgu's of  $\Gamma'$ 's is a complete set of  $C$ -unifiers of  $\Gamma$ . This set is finite.



# Unification Type

## Example (Type Finitary)

Commutative unification:  $\{f(x, y) \approx f(y, x)\}$

- ▶  $\{f(x, y) \doteq_C^? f(a, b)\}$  does not have an mgu.  $C$ -unification is not unitary.
- ▶ Show that it is finitary for any  $\mathcal{F}$ :
  - ▶ Let  $\Gamma = \{s_1 \doteq_C^? t_1, \dots, s_n \doteq_C^? t_n\}$  be a  $C$ -unification problem.
  - ▶ Consider all possible syntactic unification problems  $\Gamma' = \{s'_1 \doteq^? t'_1, \dots, s'_n \doteq^? t'_n\}$ , where  $s'_i \doteq_C s_i$  and  $t'_i \doteq_C t_i$  for each  $1 \leq i \leq n$ .
  - ▶ There are only finitely many such  $\Gamma'$ 's, because the  $C$ -equivalence class for a given term  $t$  is finite.
  - ▶ It can be shown that collection of all mgu's of  $\Gamma'$ 's is a complete set of  $C$ -unifiers of  $\Gamma$ . This set is finite.
  - ▶ If this set is not minimal (often the case), it can be minimized by removing redundant  $C$ -unifiers.



# Unification Type

## Example (Type Infinitary)

Associative unification:  $\{f(f(x, y), z) \approx f(x, f(y, z))\}$ .

- ▶  $\{f(x, a) \doteq_A^? f(a, x)\}$  has an infinite *mcsu*:  
 $\{\{x \mapsto a\}, \{x \mapsto f(a, a)\}, \{x \mapsto f(a, f(a, a))\}, \dots\}$
- ▶ Hence, *A*-unification can not be unitary or finitary.
- ▶ It is not of type zero because any *A*-unification problem has an *mcsu* that can be enumerated by the procedure from



G. Plotkin.

Building in equational theories.

In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 7, pages 73–90. Edinburgh University Press, 1972.

- ▶ *A*-unification is infinitary for any  $\mathcal{F}$ .





# Unification Type

## Example (Type Zero)

Associative-Idempotent unification:

$$\{f(f(x, y), z) \approx f(x, f(y, z)), f(x, x) \approx x\}.$$

- ▶  $\{f(x, f(y, x)) \stackrel{?}{\doteq}_{AI} f(x, f(z, x))\}$  does not have a minimal complete set of unifiers, see



F. Baader.

Unification in idempotent semigroups is of type zero.

*J. Automated Reasoning*, 2(3):283–286, 1986.

- ▶ AI-unification is of type zero.



# Unification Type. Signature Matters

Associative-commutative unification with unit:

$$ACU = \{f(f(x, y), z) \approx f(x, f(y, z)), f(x, y) \approx f(y, x), f(x, e) \approx x\}.$$

- ▶ Any *ACU* problem built using only *f* and variables has an mgu (i.e. is unitary).
- ▶ There are *ACU* problems that contain function symbols other than *f* and *e*, which are finitary, not unitary.  
For instance,  $mcsu(\{f(x, y) \doteq_{ACU}^? f(a, b)\})$  consists of four unifiers (which ones?).

Kinds of *E*-unification.



# Kinds of $E$ -Unification

One may distinguish three kinds of  $E$ -unification problems, depending on the function symbols that are allowed to occur in them.

## $E$ -Unification Problems: Elementary, with Constants, General.

- ▶  $E$ : an equational Theory.  
 $\Gamma$ : an  $E$ -unification problem over  $\mathcal{F}$ .
- ▶  $\Gamma$  is an elementary  $E$ -unification problem iff  $\mathcal{F} = \text{sig}(E)$ .
- ▶  $\Gamma$  is an  $E$ -unification problem with constants iff  $\mathcal{F} \setminus \text{sig}(E)$  consists of constants.
- ▶  $\Gamma$  is a general  $E$ -unification problem iff  $\mathcal{F} \setminus \text{sig}(E)$  may contain arbitrary function symbols.



# Unification Types of Theories wrt Kinds

- ▶ Unification type of  $E$  wrt elementary unification:  
Maximal unification type of  $E$  wrt all  $\mathcal{F}$  such that  $\mathcal{F} = sig(E)$ .
- ▶ Unification type of  $E$  wrt unification with constants:  
Maximal unification type of  $E$  wrt all  $\mathcal{F}$  such that  $\mathcal{F} \setminus sig(E)$  is a set of constants.
- ▶ Unification type of  $E$  wrt general unification: Maximal unification type of  $E$  wrt all  $\mathcal{F}$  such that  $\mathcal{F} \setminus sig(E)$  is a set of arbitrary function symbols.



# Unification Types of Theories wrt Kinds

The same equational theory can have different unification types for different kinds. Examples:

- ▶ *ACU* (Abelian monoids): Unitary wrt elementary unification, finitary wrt unification with constants and general unification.
- ▶ *AG* (Abelian groups): Unitary wrt elementary unification and unification with constants, finitary wrt general unification.



# Decision and Unification Procedures

- ▶ **Decision procedure** for an equational theory  $E$  (wrt  $\mathcal{F}$ ):  
An algorithm that for each  $E$ -unification problem  $\Gamma$  (wrt  $\mathcal{F}$ ) returns *success* if  $\Gamma$  is  $E$ -unifiable, and *failure* otherwise.



# Decision and Unification Procedures

- ▶ **Decision procedure** for an equational theory  $E$  (wrt  $\mathcal{F}$ ):  
An algorithm that for each  $E$ -unification problem  $\Gamma$  (wrt  $\mathcal{F}$ ) returns *success* if  $\Gamma$  is  $E$ -unifiable, and *failure* otherwise.
- ▶  $E$  is **decidable** if it admits a decision procedure.



# Decision and Unification Procedures

- ▶ **Decision procedure** for an equational theory  $E$  (wrt  $\mathcal{F}$ ): An algorithm that for each  $E$ -unification problem  $\Gamma$  (wrt  $\mathcal{F}$ ) returns *success* if  $\Gamma$  is  $E$ -unifiable, and *failure* otherwise.
- ▶  $E$  is **decidable** if it admits a decision procedure.
- ▶ (Minimal)  **$E$ -unification algorithm** (wrt  $\mathcal{F}$ ): An algorithm that computes a (minimal) finite complete set of  $E$ -unifiers for all  $E$ -unification problems over  $\mathcal{F}$ .





# Decision and Unification Procedures

- ▶ **Decision procedure** for an equational theory  $E$  (wrt  $\mathcal{F}$ ): An algorithm that for each  $E$ -unification problem  $\Gamma$  (wrt  $\mathcal{F}$ ) returns *success* if  $\Gamma$  is  $E$ -unifiable, and *failure* otherwise.
- ▶  $E$  is **decidable** if it admits a decision procedure.
- ▶ (Minimal)  **$E$ -unification algorithm** (wrt  $\mathcal{F}$ ): An algorithm that computes a (minimal) finite complete set of  $E$ -unifiers for all  $E$ -unification problems over  $\mathcal{F}$ .
- ▶  $E$ -unification algorithm yields a decision procedure for  $E$ .



# Decision and Unification Procedures

- ▶ **Decision procedure** for an equational theory  $E$  (wrt  $\mathcal{F}$ ): An algorithm that for each  $E$ -unification problem  $\Gamma$  (wrt  $\mathcal{F}$ ) returns *success* if  $\Gamma$  is  $E$ -unifiable, and *failure* otherwise.
- ▶  $E$  is **decidable** if it admits a decision procedure.
- ▶ (Minimal)  **$E$ -unification algorithm** (wrt  $\mathcal{F}$ ): An algorithm that computes a (minimal) finite complete set of  $E$ -unifiers for all  $E$ -unification problems over  $\mathcal{F}$ .
- ▶  $E$ -unification algorithm yields a decision procedure for  $E$ .
- ▶ (Minimal)  **$E$ -unification procedure**: A procedure that enumerates a possible infinite (minimal) complete set of  $E$ -unifiers.



# Decision and Unification Procedures

- ▶ **Decision procedure** for an equational theory  $E$  (wrt  $\mathcal{F}$ ): An algorithm that for each  $E$ -unification problem  $\Gamma$  (wrt  $\mathcal{F}$ ) returns *success* if  $\Gamma$  is  $E$ -unifiable, and *failure* otherwise.
- ▶  $E$  is **decidable** if it admits a decision procedure.
- ▶ (Minimal)  **$E$ -unification algorithm** (wrt  $\mathcal{F}$ ): An algorithm that computes a (minimal) finite complete set of  $E$ -unifiers for all  $E$ -unification problems over  $\mathcal{F}$ .
- ▶  $E$ -unification algorithm yields a decision procedure for  $E$ .
- ▶ (Minimal)  **$E$ -unification procedure**: A procedure that enumerates a possible infinite (minimal) complete set of  $E$ -unifiers.
- ▶  $E$ -unification procedure does not yield a decision procedure for  $E$ .



# Decidability wrt Kinds

Decidability of an equational theory might depend on the kinds of  $E$ -unification.

- ▶ There exists an equational theory for which elementary unification is decidable, but unification with constants is undecidable:



H.-J. Bürckert.

Some relationships between unification, restricted unification, and matching.

In J. Siekmann, editor, *Proc. 8th Int. Conference on Automated Deduction*, volume 230 of *LNCS*. Springer, 1986.



# Decidability wrt Kinds

Decidability of an equational theory might depend on the kinds of  $E$ -unification.

- ▶ There exists an equational theory for which unification with constants is decidable, but general unification is undecidable:



J. Otop.

E-unification with constants vs. general E-unification.

*Journal of Automated Reasoning*, 48(3):363–390,  
2012.



# Single Equation vs Systems of Equations

- ▶ In syntactic unification, solving systems of equations can be reduced to solving a single equation.
- ▶ For equational unification, the same holds only for general unification.
- ▶ For elementary unification and for unification with constants it is not the case.



# Single Equation vs Systems of Equations

There exists an equational theory  $E$  such that

- ▶ all elementary  $E$ -unification problems of cardinality 1 (single equations) have minimal complete sets of  $E$ -unifiers, but
- ▶  $E$  is of type zero wrt to elementary unification: There exists an elementary  $E$ -unification problem of cardinality  $> 1$  that does not have a minimal complete set of unifiers.



H.-J. Bürckert, A. Herold, and M. Schmidt-Schauß.  
On equational theories, unification, and decidability.  
*J. Symbolic Computation* 8(3,4), 3–49. 1989.



# Single Equation vs Systems of Equations

There exists an equational theory  $E$  such that

- ▶ unifiability of elementary  $E$ -unification problems of cardinality 1 (single equations) is decidable, but
- ▶ for elementary problems of larger cardinality it is undecidable.



P. Narendran and H. Otto.

Some results on equational unification.

In M. E. Stickel, editor, *Proc. 10th Int. Conference on Automated Deduction*, volume 449 of *LNAI*. Springer, 1990.





# Three Main Questions in Unification Theory

For a given  $E$ , unification theory is mainly concerned with finding answers to the following three questions:

**Decidability:** Is it decidable whether an  $E$ -unification problem is solvable? If yes, what is the complexity of this decision problem?

**Unification type:** What is the unification type of the theory  $E$ ?

**Unification algorithm:** How can we obtain an (efficient)  $E$ -unification algorithm, or a (preferably minimal)  $E$ -unification procedure?



# Three Main Questions in Unification Theory

- ▶ Unification type depends on
  - ▶ equational theory,
  - ▶ signature (kinds),
  - ▶ cardinality of unification problems.



# Three Main Questions in Unification Theory

- ▶ Unification type depends on
  - ▶ equational theory,
  - ▶ signature (kinds),
  - ▶ cardinality of unification problems.
- ▶ Decidability depends on
  - ▶ equational theory,
  - ▶ signature (kinds),
  - ▶ cardinality of unification problems.



# Outline

Motivation

Equational Theories, Reformulations of Notions

Unification Type, Kinds of Unification

**Results for Specific Theories**

General Results

# Summary of Results for Specific Theories

General unification:

Theory	Decidability	Type	Algorithm/Procedure
$\emptyset$ , BR	Yes	1	Yes
A, AU	Yes	$\infty$	Yes
C, AC, ACU	Yes	$\omega$	Yes
I, CI, ACI	Yes	$\omega$	Yes
AI	Yes	0	?
$D_{\{f,g\}}A_g$	No	$\infty$	?
AG	Yes	$\omega$	Yes
CRU	No	? ( $\infty$ or 0)	?

BR - Boolean ring, D - distributivity, CRU - commutative ring with unit.



# Commutative Unification and Matching

- ▶ C-unification inference system  $\mathcal{U}_C$  can be obtained from the  $\mathcal{U}$  by adding the C-Decomposition rule:

**C-Decomposition:**  $\{f(s_1, s_2) \doteq_C^? f(t_1, t_2)\} \uplus P'; S \Longrightarrow$   
 $\{s_1 \doteq_C^? t_2, s_2 \doteq_C^? t_1\} \cup P'; S,$   
if  $f$  is commutative.

- ▶ **C-Decomposition** and **Decomposition** transform the same system in different ways.



# Commutative Unification and Matching

- ▶ C-unification inference system  $\mathcal{U}_C$  can be obtained from the  $\mathcal{U}$  by adding the C-Decomposition rule:

$$\begin{aligned} \mathbf{C\text{-Decomposition:}} \quad \{f(s_1, s_2) \doteq_C^? f(t_1, t_2)\} \uplus P'; S \implies \\ \{s_1 \doteq_C^? t_2, s_2 \doteq_C^? t_1\} \cup P'; S, \\ \text{if } f \text{ is commutative.} \end{aligned}$$

- ▶ **C-Decomposition** and **Decomposition** transform the same system in different ways.
- ▶ C-matching algorithm  $\mathcal{M}_C$  is obtained analogously from  $\mathcal{M}$ .



# C-Unification

In order to C-unify  $s$  and  $t$ :

1. Create an initial system  $\{s \stackrel{?}{\underset{C}{\equiv}} t\}; \emptyset$ .
2. Apply successively rules from  $\mathcal{U}_C$ , building a complete tree of derivations. **C-Decomposition** and **Decomposition** rules have to be applied concurrently and form branching points in the derivation tree.





## Example. C-Unification

C-unify  $g(f(x,y),z)$  and  $g(f(f(a,b),f(b,a)),c)$ , commutative  $f$ .

$$\{g(f(x,y),z) \doteq_C^? g(f(f(a,b),f(b,a))),c)\}; \emptyset$$



## Example. C-Unification

C-unify  $g(f(x, y), z)$  and  $g(f(f(a, b), f(b, a)), c)$ , commutative  $f$ .

$$\{g(f(x, y), z) \doteq_C^? g(f(f(a, b), f(b, a)), c)\}; \emptyset$$

↓

$$\{f(x, y) \doteq_C^? f(f(a, b), f(b, a)), z \doteq_C^? c\}; \emptyset$$











# Example. C-Unification

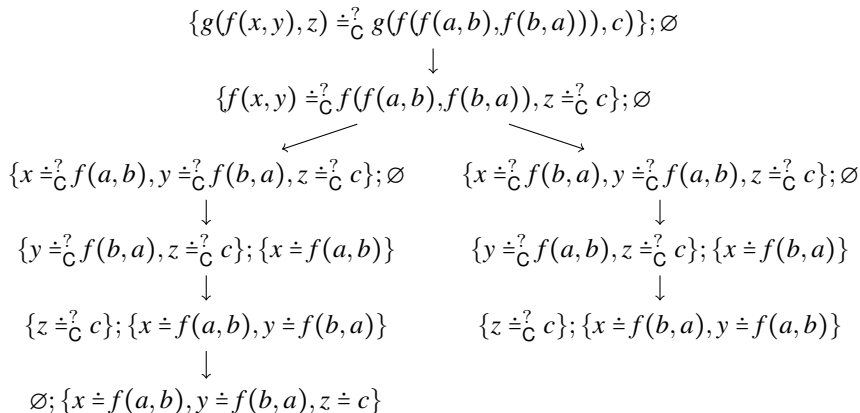
C-unify  $g(f(x, y), z)$  and  $g(f(f(a, b), f(b, a)), c)$ , commutative  $f$ .

$$\begin{array}{c} \{g(f(x, y), z) \doteq_C^? g(f(f(a, b), f(b, a)), c)\}; \emptyset \\ \downarrow \\ \{f(x, y) \doteq_C^? f(f(a, b), f(b, a)), z \doteq_C^? c\}; \emptyset \\ \swarrow \quad \searrow \\ \{x \doteq_C^? f(a, b), y \doteq_C^? f(b, a), z \doteq_C^? c\}; \emptyset \quad \{x \doteq_C^? f(b, a), y \doteq_C^? f(a, b), z \doteq_C^? c\}; \emptyset \\ \downarrow \quad \quad \quad \downarrow \\ \{y \doteq_C^? f(b, a), z \doteq_C^? c\}; \{x \doteq_C^? f(a, b)\} \quad \{y \doteq_C^? f(a, b), z \doteq_C^? c\}; \{x \doteq_C^? f(b, a)\} \\ \downarrow \quad \quad \quad \downarrow \\ \{z \doteq_C^? c\}; \{x \doteq_C^? f(a, b), y \doteq_C^? f(b, a)\} \\ \downarrow \\ \emptyset; \{x \doteq_C^? f(a, b), y \doteq_C^? f(b, a), z \doteq_C^? c\} \end{array}$$



# Example. C-Unification

C-unify  $g(f(x, y), z)$  and  $g(f(f(a, b), f(b, a)), c)$ , commutative  $f$ .





# Example. C-Unification

C-unify  $g(f(x, y), z)$  and  $g(f(f(a, b), f(b, a)), c)$ , commutative  $f$ .

$$\{g(f(x, y), z) \doteq_C^? g(f(f(a, b), f(b, a)), c)\}; \emptyset$$



$$\{f(x, y) \doteq_C^? f(f(a, b), f(b, a)), z \doteq_C^? c\}; \emptyset$$



$$\{x \doteq_C^? f(a, b), y \doteq_C^? f(b, a), z \doteq_C^? c\}; \emptyset$$

$$\{x \doteq_C^? f(b, a), y \doteq_C^? f(a, b), z \doteq_C^? c\}; \emptyset$$



$$\{y \doteq_C^? f(b, a), z \doteq_C^? c\}; \{x \doteq f(a, b)\}$$

$$\{y \doteq_C^? f(a, b), z \doteq_C^? c\}; \{x \doteq f(b, a)\}$$



$$\{z \doteq_C^? c\}; \{x \doteq f(a, b), y \doteq f(b, a)\}$$

$$\{z \doteq_C^? c\}; \{x \doteq f(b, a), y \doteq f(a, b)\}$$



$$\emptyset; \{x \doteq f(a, b), y \doteq f(b, a), z \doteq c\}$$

$$\emptyset; \{x \doteq f(b, a), y \doteq f(a, b), z \doteq c\}$$



# Example. C-Unification

C-unify  $g(f(x, y), z)$  and  $g(f(f(a, b), f(b, a)), c)$ , commutative  $f$ .

$$\{g(f(x, y), z) \doteq_C^? g(f(f(a, b), f(b, a)), c)\}; \emptyset$$



$$\{f(x, y) \doteq_C^? f(f(a, b), f(b, a)), z \doteq_C^? c\}; \emptyset$$



$$\{x \doteq_C^? f(a, b), y \doteq_C^? f(b, a), z \doteq_C^? c\}; \emptyset$$

$$\{x \doteq_C^? f(b, a), y \doteq_C^? f(a, b), z \doteq_C^? c\}; \emptyset$$



$$\{y \doteq_C^? f(b, a), z \doteq_C^? c\}; \{x \doteq f(a, b)\}$$

$$\{y \doteq_C^? f(a, b), z \doteq_C^? c\}; \{x \doteq f(b, a)\}$$



$$\{z \doteq_C^? c\}; \{x \doteq f(a, b), y \doteq f(b, a)\}$$

$$\{z \doteq_C^? c\}; \{x \doteq f(b, a), y \doteq f(a, b)\}$$



$$\emptyset; \{x \doteq f(a, b), y \doteq f(b, a), z \doteq c\}$$

$$\emptyset; \{x \doteq f(b, a), y \doteq f(a, b), z \doteq c\}$$

Not minimal.



# Properties of the C-Unification Algorithm

## Theorem

*Applied to a C-unification problem  $P$ , the C-unification algorithm terminates and computes a complete set of C-unifiers of  $P$ .*

# Properties of the C-Unification Algorithm

## Theorem

*Applied to a C-unification problem  $P$ , the C-unification algorithm terminates and computes a complete set of C-unifiers of  $P$ .*

## Proof.

- ▶ Termination is proved using the same measure as for syntactic unification.
- ▶ Completeness is based on the following two facts:
  - ▶ If  $\Gamma$  is transformed by only one rule of  $\mathcal{U}_C$  into  $\Gamma'$ , then  $u_C(\Gamma) = u_C(\Gamma')$ .
  - ▶ If  $\Gamma$  is transformed by two rules of  $\mathcal{U}_C$  into  $\Gamma_1$  and  $\Gamma_2$ , then  $u_C(\Gamma) = u_C(\Gamma_1) \cup u_C(\Gamma_2)$ .



# MCSU for C-Unification/Matching Problems Can Be Large

## Example

- ▶ Problem:  $f(f(x_1, x_2), f(x_3, x_4)) \stackrel{?}{\doteq}_C f(f(a, b), f(c, d))$ .
- ▶ *mcsu* contains 4! substitutions.



# Properties of the C-Unification Algorithm

- ▶ The algorithm, in general, does not return a minimal complete set of C-unifiers.
- ▶ The obtained complete set can be further minimized, removing redundant unifiers.
- ▶ Not clear how to design a C-unification algorithm that computes a minimal complete set of unifiers directly.



# Properties of the C-Unification Algorithm

## Theorem

*The decision problem of C-matching and unification is NP-complete.*

## Proof.

Exercise. □



# ACU-Unification

$$\text{ACU} = \{f(f(x, y), z) \approx f(x, f(y, z)), f(x, y) \approx f(y, x), f(x, e) \approx x\}$$

1. Associativity, commutativity, unit element.
2.  $f$  is associative and commutative,  $e$  is the unit element.





# Example: Elementary ACU-Unification

Elementary ACU-unification problem:

$$\Gamma = \{f(x, f(x, y)) \stackrel{?}{\doteq}_{\text{ACU}} f(z, f(z, z))\}$$

Solving idea:

1. Associate with the equation in  $\Gamma$  a homogeneous linear Diophantine equation  $2x + y = 3z$ .
2. The equation states that the number of new variables introduced by a unifier  $\sigma$  in both sides of  $\Gamma\sigma$  must be the same.

(Continues on the next slide.)



## Example. Elementary ACU-Unification (Cont.)

3. Solve  $2x + y = 3z$  over nonnegative integers. Three minimal solutions:

$$x = 1, y = 1, z = 1$$

$$x = 0, y = 3, z = 1$$

$$x = 3, y = 0, z = 2$$

Any other solution of the equation can be obtained as a nonnegative linear combination of these three solutions.

(Continues on the next slide.)



## Example. Elementary ACU-Unification (Cont.)

4. Introduce new variables  $v_1, v_2, v_3$  for each solution of the Diophantine equation:

	$x$	$y$	$z$
$v_1$	1	1	1
$v_2$	0	3	1
$v_3$	3	0	2

5. Each row corresponds to a unifier of  $\Gamma$ :

$$\sigma_1 = \{x \mapsto v_1, y \mapsto v_1, z \mapsto v_1\}$$

$$\sigma_2 = \{x \mapsto e, y \mapsto f(v_2, f(v_2, v_2)), z \mapsto v_2\}$$

$$\sigma_3 = \{x \mapsto f(v_3, f(v_3, v_3)), y \mapsto e, z \mapsto f(v_3, v_3)\}$$

However, none of them is an mgu.



## Example. Elementary ACU-Unification (Cont.)

6. To obtain an mgu, we should combine all three solutions:

	$x$	$y$	$z$
$v_1$	1	1	1
$v_2$	0	3	1
$v_3$	3	0	2

The columns indicate that the mgu we are looking for should have

- ▶ in the binding for  $x$  one  $v_1$ , zero  $v_2$ , and three  $v_3$ 's,
- ▶ in the binding for  $y$  one  $v_1$ , three  $v_2$ 's, and zero  $v_3$ ,
- ▶ in the binding for  $z$  one  $v_1$ , one  $v_2$ , and two  $v_3$ 's

7. Hence, we can construct an mgu:

$$\sigma = \{x \mapsto f(v_1, f(v_3, f(v_3, v_3))), y \mapsto f(v_1, f(v_2, f(v_2, v_2))), \\ z \mapsto f(v_1, f(v_2, f(v_3, v_3)))\}$$





## Example: ACU-Unification with constants

- ▶ ACU-unification problem with constants

$$\Gamma = \{f(x, f(x, y)) \doteq_{\text{ACU}}^? f(a, f(z, f(z, z)))\}$$

reduces to inhomogeneous linear Diophantine equation

$$S = \{2x + y = 3z + 1\}.$$

- ▶ Every natural solution of  $S$  is obtained as the sum of one of its minimal solutions and a solution of the corresponding homogeneous LDE  $2x + y = 3z$ .
- ▶ One element of the minimal complete set of unifiers of  $\Gamma$  is obtained from the combination of one minimal solution of  $S$  with the set of all minimal solutions of  $2x + y = 3z$ .



## Example: ACU-Unification with constants

- ▶ ACU-unification problem with constants

$$\Gamma = \{f(x, f(x, y)) \doteq_{\text{ACU}}^? f(a, f(z, f(z, z)))\}$$

reduces to inhomogeneous linear Diophantine equation

$$S = \{2x + y = 3z + 1\}.$$

- ▶ The minimal complete set of unifiers of  $\Gamma$  is  $\{\sigma_1, \sigma_2\}$ , where

$$\begin{aligned}\sigma_1 = \{ &x \mapsto f(v_1, f(v_3, f(v_3, v_3))), \\ &y \mapsto f(a, f(v_1, f(v_2, f(v_2, v_2))), \\ &z \mapsto f(v_1, f(v_2, f(v_3, v_3)))\}\end{aligned}$$

$$\begin{aligned}\sigma_2 = \{ &x \mapsto f(a, f(a, f(v_1, f(v_3, f(v_3, v_3))))), \\ &y \mapsto f(v_1, f(v_2, f(v_2, v_2))), \\ &z \mapsto f(a, f(v_1, f(v_2, f(v_3, v_3))))\}\end{aligned}$$



# ACU-Unification with constants

- ▶ If an ACU-unification problem contains more than one constant, solve the corresponding inhomogeneous LDE for each constant.
- ▶ The unifiers in the minimal complete set correspond to all possible combinations of the minimal solutions of these inhomogeneous equations.





# ACU-Unification with constants

## Example

$xy \stackrel{?}{\text{ACU}} aabbb$ :

- ▶ Equation for  $a$ :  $2x + y = 2$ . Minimal solutions:  $(1, 0)$  and  $(0, 2)$ .
- ▶ Corresponding unifiers:  $\{x \mapsto a, y \mapsto e\}$ ,  $\{x \mapsto e, y \mapsto aa\}$
- ▶ Equation for  $b$ :  $2x + y = 3$ . Minimal solutions:  $(0, 3)$  and  $(1, 1)$ .
- ▶ Corresponding unifiers:  $\{x \mapsto e, y \mapsto bbb\}$ ,  $\{x \mapsto b, y \mapsto b\}$
- ▶ Unifiers in the minimal complete set:  $\{x \mapsto a, y \mapsto bbb\}$ ,  $\{x \mapsto ab, y \mapsto b\}$ ,  $\{x \mapsto e, y \mapsto aabbb\}$ ,  $\{x \mapsto b, y \mapsto aab\}$ .



# From ACU to AC

## Example

- ▶ How to solve  $\Gamma_1 = \{f(x, f(x, y)) \doteq_{AC}^? f(z, f(z, z))\}$ ?
- ▶ We “know” how to solve  $\Gamma_2 = \{f(x, f(x, y)) \doteq_{ACU}^? f(z, f(z, z))\}$ , but its mgu is not an mgu for  $\Gamma_1$ .
- ▶ Mgu of  $\Gamma_2$ :

$$\sigma = \{x \mapsto f(v_1, f(v_3, f(v_3, v_3))), y \mapsto f(v_1, f(v_2, f(v_2, v_2))), \\ z \mapsto f(v_1, f(v_2, f(v_3, v_3)))\}$$

- ▶ Unifier of  $\Gamma_1$ :  $\vartheta = \{x \mapsto v_1, y \mapsto v_1, z \mapsto v_1\}$ .
- ▶  $\sigma$  is not more general modulo AC than  $\vartheta$ .



# From ACU to AC

## Example

- ▶ Idea: Take the mgu of  $\Gamma_2$ .
- ▶ Compose it with all possible erasing substitutions that map a subset of  $\{v_1, v_2, v_3\}$  to the unit element.
- ▶ Restriction: The result of the composition should not map  $x$ ,  $y$ , and  $z$  to the unit element.



# From ACU to AC

## Example

Minimal complete set of unifiers for  $\Gamma_1$ :

$$\sigma_1 = \{x \mapsto f(v_1, f(v_3, f(v_3, v_3))), y \mapsto f(v_1, f(v_2, f(v_2, v_2))), \\ z \mapsto f(v_1, f(v_2, f(v_3, v_3)))\}$$

$$\sigma_2 = \{x \mapsto f(v_3, f(v_3, v_3)), y \mapsto f(v_2, f(v_2, v_2)), \\ z \mapsto f(v_2, f(v_3, v_3))\}$$

$$\sigma_3 = \{x \mapsto f(v_1, f(v_3, f(v_3, v_3))), y \mapsto v_1, z \mapsto f(v_1, f(v_3, v_3))\}$$

$$\sigma_4 = \{x \mapsto v_1, y \mapsto f(v_1, f(v_2, f(v_2, v_2))), z \mapsto f(v_1, v_2)\}$$

$$\sigma_5 = \{x \mapsto v_1, y \mapsto v_1, z \mapsto v_1\}$$



# How to Solve Systems of LDEs over Naturals?

Contejean-Devie Algorithm:



[Evelyne Contejean and Hervé Devie.](#)

An Efficient Incremental Algorithm for Solving Systems of Linear Diophantine Equations.

[Information and Computation 113\(1\): 143–172 \(1994\).](#)



# How to Solve Systems of LDEs over Naturals?

Contejean-Devie Algorithm:



[Evelyne Contejean and Hervé Devie.](#)

An Efficient Incremental Algorithm for Solving Systems of Linear Diophantine Equations.

[Information and Computation 113\(1\): 143–172 \(1994\).](#)

Generalizes Fortenbacher's Algorithm for solving a single equation:



[Michael Clausen and Albrecht Fortenbacher.](#)

Efficient Solution of Linear Diophantine Equations.

[J. Symbolic Computation 8\(1,2\): 201–216 \(1989\).](#)



# How to Solve Systems of LDEs over Naturals?

Contejean-Devie Algorithm:



[Evelyne Contejean and Hervé Devie.](#)

An Efficient Incremental Algorithm for Solving Systems of Linear Diophantine Equations.

[Information and Computation 113\(1\): 143–172 \(1994\).](#)

Generalizes Fortenbacher's Algorithm for solving a single equation:



[Michael Clausen and Albrecht Fortenbacher.](#)

Efficient Solution of Linear Diophantine Equations.

[J. Symbolic Computation 8\(1,2\): 201–216 \(1989\).](#)

Will be discussed in the next lecture.



# Example. $E$ -Unification of Type 0

## Example

- ▶ Equational theory:  $E = \{f(e, x) \approx x, g(f(x, y)) \approx g(y)\}$ .
- ▶  $E$ -unification problem:  $\Gamma = \{g(x) \doteq_E^? g(e)\}$ .





# Example. $E$ -Unification of Type 0

## Example

- ▶ Equational theory:  $E = \{f(e, x) \approx x, g(f(x, y)) \approx g(y)\}$ .
- ▶  $E$ -unification problem:  $\Gamma = \{g(x) \stackrel{?}{\approx}_E g(e)\}$ .
- ▶ Complete (why?) set of solutions:

$$\sigma_0 = \{x \mapsto e\}$$

$$\sigma_1 = \{x \mapsto f(x_0, e)\}$$

$$\sigma_2 = \{x \mapsto f(x_1, f(x_0, e))\}$$

...

$$\sigma_n = \{x \mapsto f(x_{n-1}, x\sigma_{n-1})\}$$

...



# Example. $E$ -Unification of Type 0

## Example

- ▶ Equational theory:  $E = \{f(e, x) \approx x, g(f(x, y)) \approx g(y)\}$ .
- ▶  $E$ -unification problem:  $\Gamma = \{g(x) \doteq_E^? g(e)\}$ .
- ▶ Complete (why?) set of solutions:

$$\sigma_0 = \{x \mapsto e\}$$

$$\sigma_1 = \{x \mapsto f(x_0, e)\}$$

$$\sigma_2 = \{x \mapsto f(x_1, f(x_0, e))\}$$

...

$$\sigma_n = \{x \mapsto f(x_{n-1}, x\sigma_{n-1})\}$$

...

- ▶ No *mcsu*.  $\sigma_i = \{x \mapsto \sigma_{i+1}(x_i \mapsto e)\}$ .  $\sigma_i \not\leq_E^{\{x\}} \sigma_j$  for  $i > j$ .

Infinite descending chain:  $\sigma_0 \not\leq_E^{\{x\}} \sigma_1 \not\leq_E^{\{x\}} \sigma_2 \not\leq_E^{\{x\}} \dots$



# Example. $E$ -Unification of Type 0

## Example (Cont.)

Why does  $\sigma_0 \succ_E^{\{x\}} \sigma_1 \succ_E^{\{x\}} \sigma_2 \succ_E^{\{x\}} \dots$  imply that there is no *mcsu*?

- ▶ Let  $S = \{\sigma_0, \sigma_1, \dots\}$ .



# Example. $E$ -Unification of Type 0

## Example (Cont.)

Why does  $\sigma_0 \succ_E^{\{x\}} \sigma_1 \succ_E^{\{x\}} \sigma_2 \succ_E^{\{x\}} \dots$  imply that there is no *mcsu*?

- ▶ Let  $S = \{\sigma_0, \sigma_1, \dots\}$ .
- ▶ Let  $S'$  be an arbitrary complete set of unifiers of  $\Gamma$ .



# Example. $E$ -Unification of Type 0

## Example (Cont.)

Why does  $\sigma_0 \succ_E^{\{x\}} \sigma_1 \succ_E^{\{x\}} \sigma_2 \succ_E^{\{x\}} \dots$  imply that there is no *mcsu*?

- ▶ Let  $S = \{\sigma_0, \sigma_1, \dots\}$ .
- ▶ Let  $S'$  be an arbitrary complete set of unifiers of  $\Gamma$ .
- ▶ Since  $S$  is complete, for any  $\vartheta \in S'$  there exists  $\sigma_i \in S$  such that  $\sigma_i \leq_E^{\{x\}} \vartheta$ .



# Example. $E$ -Unification of Type 0

## Example (Cont.)

Why does  $\sigma_0 \succ_E^{\{x\}} \sigma_1 \succ_E^{\{x\}} \sigma_2 \succ_E^{\{x\}} \dots$  imply that there is no *mcsu*?

- ▶ Let  $S = \{\sigma_0, \sigma_1, \dots\}$ .
- ▶ Let  $S'$  be an arbitrary complete set of unifiers of  $\Gamma$ .
- ▶ Since  $S$  is complete, for any  $\vartheta \in S'$  there exists  $\sigma_i \in S$  such that  $\sigma_i \leq_E^{\{x\}} \vartheta$ .
- ▶ Since  $\sigma_{i+1} \prec_E^{\{x\}} \sigma_i$ , we get  $\sigma_{i+1} \prec_E^{\{x\}} \vartheta$ .



# Example. $E$ -Unification of Type 0

## Example (Cont.)

Why does  $\sigma_0 \succ_E^{\{x\}} \sigma_1 \succ_E^{\{x\}} \sigma_2 \succ_E^{\{x\}} \dots$  imply that there is no *mcsu*?

- ▶ Let  $S = \{\sigma_0, \sigma_1, \dots\}$ .
- ▶ Let  $S'$  be an arbitrary complete set of unifiers of  $\Gamma$ .
- ▶ Since  $S$  is complete, for any  $\vartheta \in S'$  there exists  $\sigma_i \in S$  such that  $\sigma_i \leq_E^{\{x\}} \vartheta$ .
- ▶ Since  $\sigma_{i+1} \leq_E^{\{x\}} \sigma_i$ , we get  $\sigma_{i+1} \leq_E^{\{x\}} \vartheta$ .
- ▶ On the other hand, since  $S'$  is complete, there exists  $\eta \in S'$  such that  $\eta \leq_E^{\{x\}} \sigma_{i+1}$ .



# Example. $E$ -Unification of Type 0

## Example (Cont.)

Why does  $\sigma_0 \succ_E^{\{x\}} \sigma_1 \succ_E^{\{x\}} \sigma_2 \succ_E^{\{x\}} \dots$  imply that there is no *mcsu*?

- ▶ Let  $S = \{\sigma_0, \sigma_1, \dots\}$ .
- ▶ Let  $S'$  be an arbitrary complete set of unifiers of  $\Gamma$ .
- ▶ Since  $S$  is complete, for any  $\vartheta \in S'$  there exists  $\sigma_i \in S$  such that  $\sigma_i \leq_E^{\{x\}} \vartheta$ .
- ▶ Since  $\sigma_{i+1} \prec_E^{\{x\}} \sigma_i$ , we get  $\sigma_{i+1} \prec_E^{\{x\}} \vartheta$ .
- ▶ On the other hand, since  $S'$  is complete, there exists  $\eta \in S'$  such that  $\eta \leq_E^{\{x\}} \sigma_{i+1}$ .
- ▶ Hence,  $\eta \prec_E^{\{x\}} \vartheta$  which implies that  $S'$  is not minimal.





# Specific vs General Results

For each specific equational theory separately studying

- ▶ decidability,
- ▶ unification type,
- ▶ unification algorithm/procedure.

Can one study these problems for bigger classes of equational theories?



# Outline

Motivation

Equational Theories, Reformulations of Notions

Unification Type, Kinds of Unification

Results for Specific Theories

**General Results**

# Specific vs General Results

For each specific equational theory separately studying

- ▶ decidability,
- ▶ unification type,
- ▶ unification algorithm/procedure.

Can one study these problems for bigger classes of equational theories?



# General Results

In general, unification modulo equational theories

- ▶ is undecidable,
- ▶ unification type of a given theory is undecidable,
- ▶ admits a complete unification procedure (Gallier & Snyder, called an universal  $E$ -unification procedure).



# General Results

Universal  $E$ -unification procedure  $\mathcal{U}_E$ .

Rules:

- ▶ **Trivial, Orient, Decomposition, Variable Elimination** from  $\mathcal{U}$ , plus
- ▶ **Lazy Paramodulation:**

$$\{e[u]\} \cup P'; S \Longrightarrow \{l \doteq^? u, e[r]\} \cup P'; S,$$

for a fresh variant of the identity  $l \approx r$  from  $E \cup E^{-1}$ , where

- ▶  $e[u]$  is an equation where the term  $u$  occurs,
- ▶  $u$  is not a variable,
- ▶ if  $l$  is not a variable, then the top symbol of  $l$  and  $u$  are the same.



# General Results

Universal  $E$ -unification procedure. Control.

In order to solve a unification problem  $\Gamma$  modulo a given  $E$ :

- ▶ Create an initial system  $\Gamma; \emptyset$ .
- ▶ Apply successively rules from  $\mathcal{U}_E$ , building a complete tree of derivations.
- ▶ No other inference rule may be applied to the equation  $l \doteq^? u$  that is generated by the Lazy Paramodulation rule before it is subjected to a Decomposition step.



# General Results

Universal  $E$ -unification procedure.

## Example

$$E = \{f(a, b) \approx a, a \approx b\}.$$

Unification problem:  $\{f(x, x) \doteq_E^? x\}$ .

Computing a unifier  $\{x \mapsto a\}$  by the universal procedure:

$$\begin{aligned} \{f(x, x) \doteq_E^? x\}; \emptyset &\Longrightarrow_{LP} \{f(a, b) \doteq_E^? f(x, x), a \doteq_E^? x\}; \emptyset \\ &\Longrightarrow_D \{a \doteq_E^? x, b \doteq_E^? x, a \doteq_E^? x\}; \emptyset \\ &\Longrightarrow_O \{x \doteq_E^? a, b \doteq_E^? x, a \doteq_E^? x\}; \emptyset \\ &\Longrightarrow_S \{b \doteq_E^? a, a \doteq_E^? a\}; \{x \doteq a\} \\ &\Longrightarrow_{LP} \{a \doteq_E^? a, b \doteq_E^? b, a \doteq_E^? a\}; \{x \doteq a\} \\ &\Longrightarrow_T^+ \emptyset; \{x \doteq a\} \end{aligned}$$



# General Results

Pros and cons of the universal procedure:

- ▶ Pros: Is sound and complete. Can be used for any  $E$ .
- ▶ Cons: Very inefficient. Usually does not yield a decision procedure or a (minimal)  $E$ -unification algorithm even for unitary or finitary theories with decidable unification.





# General Results

More useful results can be obtained by imposing additional restrictions on equational theories:

- ▶ Syntactic approaches: Restricting syntactic form of the identities defining equational theories.
- ▶ Semantic approaches: Depend on properties of the free algebras defined by the equational theory.

