# 15 Foundation of Programming Languages and Software Engineering: *The Word Problem*

Summer Term 2010

Robert Elsässer

Albert-Ludwigs-Universität Freiburg

# Central Problems of Equational Reasoning

**Definition (Validity)**

$s \approx t$ is valid in $\mathcal{E}$ iff $s \approx_{\mathcal{E}} t$

**Definition (Satisfiability)**

$s \approx t$ is satisfiable in $\mathcal{E}$ if there exists a substitution $\sigma$ such that $\sigma s \approx_{\mathcal{E}} \sigma t$.

# The Word Problem

## Definition

Suppose $\Sigma$ is a signature and $X$ a set of variables disjoint from $\Sigma$.

- The word problem for $\mathcal{E}$ is the problem of deciding $s \approx_{\mathcal{E}} t$ for arbitrary $s, t \in T(\Sigma, X)$.
- The ground word problem for $\mathcal{E}$ is the problem of deciding $s \approx_{\mathcal{E}} t$ for arbitrary $s, t \in T(\Sigma, \emptyset)$.

# Solving the Word Problem

## A Sample Problem

Given $\Sigma_{int} = \{\mathtt{zero}^{(0)}, \mathtt{pred}^{(1)}, \mathtt{succ}^{(1)}\}$ and
$\mathcal{E}_{int} = \{\mathtt{pred}(\mathtt{succ}(x)) = x, \mathtt{succ}(\mathtt{pred}(x)) = x\}$
we would like to decide whether
$\mathtt{succ}(\mathtt{zero}) \approx_{\mathcal{E}_{int}} \mathtt{succ}(\mathtt{succ}(\mathtt{pred}(\mathtt{zero})))$

## A solution

- Use identities as reduction rules:
  $\mathtt{pred}(\mathtt{succ}(x)) \rightarrow_{\mathcal{E}_{int}} x, \mathtt{succ}(\mathtt{pred}(x)) \rightarrow_{\mathcal{E}_{int}} x$
- Apply reduction rules to both terms:
  - $\mathtt{succ}(\underline{\mathtt{succ}(\mathtt{pred}(\mathtt{zero}))}) \rightarrow_{\mathcal{E}_{int}} \mathtt{succ}(\underline{\mathtt{zero}})$
- Check whether the resulting terms are identical.

Problem: Applying the reduction rules might not terminate.

# An Undecidable Word Problem

## Combinatory Logic

$\Sigma_C = \{S^{(0)}, I^{(0)}, K^{(0)}, \cdot^{(2)}\}$

$$\begin{aligned}\mathcal{E}_C &= \{((S \cdot x) \cdot y) \cdot z = (x \cdot z) \cdot (y \cdot z), \\ &\quad (K \cdot x) \cdot y = x, I \cdot x = x\}\end{aligned}$$

Look at the following reduction sequence:

$$\begin{aligned} & \underline{((S \cdot I) \cdot I) \cdot ((S \cdot I) \cdot I)} \\ \to_{\mathcal{E}_C}\ & (\underline{I \cdot ((S \cdot I) \cdot I)}) \cdot (I \cdot ((S \cdot I) \cdot I)) \\ \to_{\mathcal{E}_C}\ & ((S \cdot I) \cdot I) \cdot (\underline{I \cdot ((S \cdot I) \cdot I)}) \\ \to_{\mathcal{E}_C}\ & ((S \cdot I) \cdot I) \cdot ((S \cdot I) \cdot I) \end{aligned}$$

In general: All computable functions can be encoded as ground terms over $\Sigma_C \Rightarrow$ the word problem for $\mathcal{E}_C$ is undecidable.

# The Reduction Relation Generated by $\Sigma$-Identities

## Definition

Let $\mathcal{E}$ be a set of $\Sigma$-identities.
The reduction relation $\rightarrow_{\mathcal{E}} \subseteq T(\Sigma, X) \times T(\Sigma, X)$ is defined as

$$s \rightarrow_{\mathcal{E}} t \text{ iff}$$

there exists $(l, r) \in \mathcal{E}$, $p \in \mathcal{P}os(s)$, and a substitution $\sigma$ with $s|_p = \sigma(l)$ and $t = s[\sigma(r)]_p$.

# Example

## Computing with Groups

$$\Sigma_G = \{e^{(0)}, i^{(1)}, f^{(2)}\}$$
$$\begin{aligned}\mathcal{E}_G = \{& f(x, f(y, z)) = f(f(x, y), z),\\ & f(e, x) = x,\\ & f(i(x), x) = e\}\end{aligned}$$

$$\begin{array}{llr} & f(i(e), f(e, e)) & \sigma_1 = \{x \mapsto i(e), y \mapsto e, z \mapsto e\}, 1^{st}\ \text{id} \\ \to_{\mathcal{E}_G} & f(f(i(e), e), e) & \sigma_2 = \{x \mapsto e\}, 3^{rd}\ \text{id} \\ \to_{\mathcal{E}_G} & f(e, e) & \sigma_3 = \{x \mapsto e\}, 2^{nd}\ \text{id} \\ \to_{\mathcal{E}_G} & e & \end{array}$$

# Composing Relations

## Definition

Given two relations $R \subseteq A \times B$ and $S \subseteq B \times C$, their composition is defined by

$$R \circ S := \{(x, z) \in A \times C \mid \text{there exists some } y \in B \text{ with } (x, y) \in R \text{ and } (y, z) \in S\}$$

## Example

Suppose $R = \{\mathrm{FR} \to \mathrm{OG}, \mathrm{OG} \to \mathrm{KA}, \mathrm{KA} \to \mathrm{MA}\}$.
Then $R \circ R = \{\mathrm{FR} \to \mathrm{KA}, \mathrm{OG} \to \mathrm{MA}\}$.

# Notations for Reduction Relations

Suppose $\rightarrow$ is a binary relation on $M$.

| | |
|---|---|
| $\xrightarrow{0} := \{(x,x) \mid x \in M\}$ | identity |
| $\xrightarrow{i+1} := \xrightarrow{i} \circ \rightarrow$ | $(i+1)$-fold composition, $i \geq 0$ |
| $\xrightarrow{+} := \bigcup_{i>0} \xrightarrow{i}$ | transitive closure |
| $\xrightarrow{*} := \xrightarrow{+} \cup \xrightarrow{0}$ | reflexive transitive closure |
| $\xrightarrow{=} := \rightarrow \cup \xrightarrow{0}$ | reflexive closure |
| $\leftarrow := \{(y,x) \mid x \rightarrow y\}$ | inverse |
| $\leftrightarrow := \leftarrow \cup \rightarrow$ | symmetric closure |
| $\xleftrightarrow{+} := (\leftrightarrow)^{+}$ | transitive symmetric closure |
| $\xleftrightarrow{*} := (\leftrightarrow)^{*}$ | reflexive transitive symmetric closure |

# Terminology for Reduction Relations (1)

Suppose $\rightarrow$ is a binary relation on $M$ and $x, y \in M$.

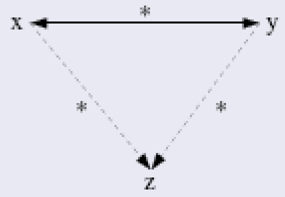
- $x$ is reducible iff there is a $z \in M$ with $x \rightarrow z$.
- $x$ is in normal form iff it is not reducible.
- $y$ is a normal form of $x$ iff $x \xrightarrow{*} y$ and $y$ is in normal form.
- if $x$ has a unique normal form, it is denoted by $x\downarrow$.
- $x$ and $y$ are joinable iff there is a $z \in M$ such that $x \xrightarrow{*} z \xleftarrow{*} y$. We then write $x \downarrow y$.

# Terminology for Reduction Relations (2)

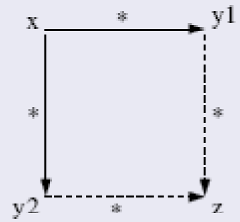
## Definition

A reduction $\longrightarrow$ is called

- Church-Rosser iff $x \overset{*}{\longleftrightarrow} y$ implies $x \downarrow y$,




- confluent iff $y_1 \overset{*}{\longleftarrow} x \overset{*}{\longrightarrow} y_2$ implies $y_1 \downarrow y_2$,




- semi-Confluent iff $y_1 \longleftarrow x \overset{*}{\longrightarrow} y_2$ implies $y_1 \downarrow y_2$,
- terminating iff there is no infinite chain $x_0 \longrightarrow x_1 \longrightarrow x_2 \longrightarrow \ldots$.

# Deciding the Word Problem

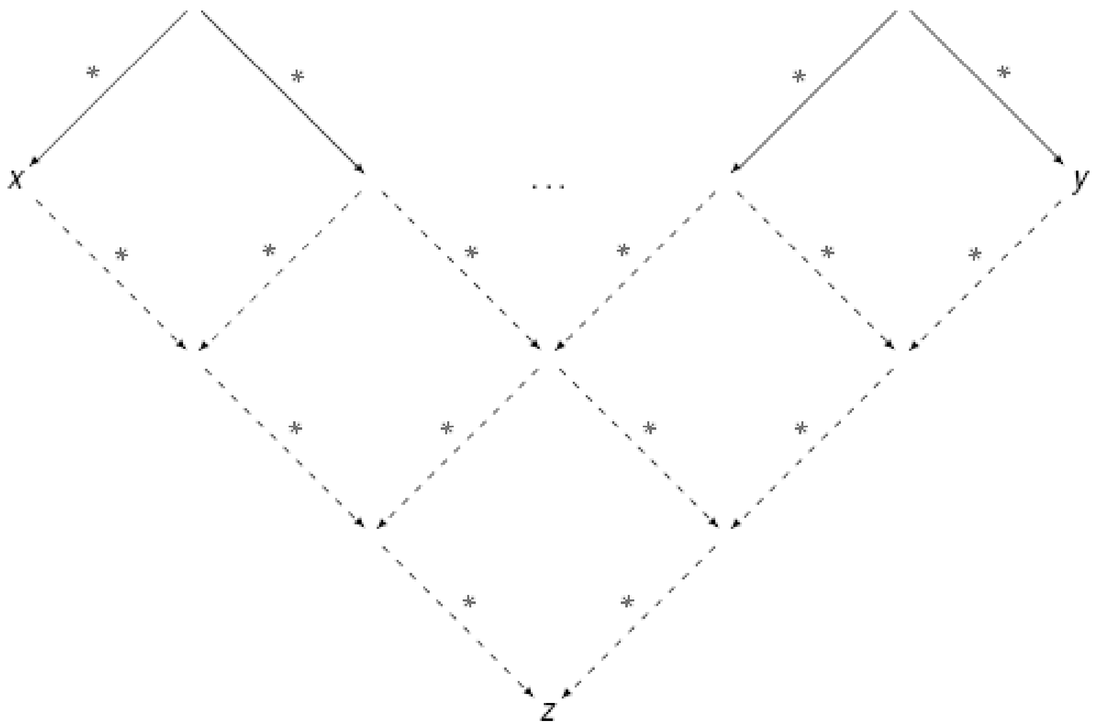
**Theorem (Deciding the word problem for $\mathcal{E}$)**

If $\mathcal{E}$ is finite and $\rightarrow_{\mathcal{E}}$ is confluent and terminating, then the word problem for $\mathcal{E}$ is decidable.

- Plan: To decide whether $s \approx_{\mathcal{E}} t$ holds, compare $s{\downarrow_{\mathcal{E}}}$ and $t{\downarrow_{\mathcal{E}}}$ for syntactic equality.
- Caveat:
  - $s{\downarrow_{\mathcal{E}}}$ and $t{\downarrow_{\mathcal{E}}}$ must exist
  - $s{\downarrow_{\mathcal{E}}}$ and $t{\downarrow_{\mathcal{E}}}$ must be computable
- Before proving the theorem, we need to establish some lemmas and facts.

# Church-Rosser and Confluence are Equivalent

- It is easy to see that any Church-Rosser relation is confluent.
- If $\rightarrow$ is confluent and $x \overset{*}{\longleftrightarrow} y$, then we can visualize the proof of $x \downarrow y$ as follows:



# Church-Rosser and Confluence are Equivalent

**Lemma**

The following conditions are equivalent:

1. $\rightarrow$ has the Church-Rosser property.
2. $\rightarrow$ is confluent.
3. $\rightarrow$ is semi-confluent.

*Proof.* We show that the implications $1 \Rightarrow 2 \Rightarrow 3 \Rightarrow 1$ hold

$1 \Rightarrow 2$ If $\rightarrow$ has the Church-Rosser property and $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$, then $y_1 \overset{*}{\leftrightarrow} y_2$. Hence, by the Church-Rosser property, $y_1 \downarrow y_2$, i.e. $\rightarrow$ is confluent.

$2 \Rightarrow 3$ Obviously any confluent relation is semi-confluent.

# Proof (cont.)

$3 \Rightarrow 1$ If $\rightarrow$ is semi-confluent and $x \overset{*}{\leftrightarrow} y$, then we show $x \downarrow y$ by induction on the length of the chain $x \overset{*}{\leftrightarrow} y$.

- $x = y$, trivial.
- If $x \overset{*}{\leftrightarrow} y' \leftrightarrow y$, we know $x \downarrow y'$ by IH. We show $x \downarrow y$ by case distinction:
  - $y' \leftarrow y$: $x \downarrow y$ follows directly from $x \downarrow y'$.
  - $y' \rightarrow y$ : from the IH, we get $x \overset{*}{\rightarrow} z$ and $z \overset{*}{\leftarrow} y'$ for some z. Semi-confluence implies $z \downarrow y$, hence $x \downarrow y$.

# Existence and Uniqueness of Normal Forms

- If $\rightarrow$ is confluent, every element has at most one normal form.
- If $\rightarrow$ is terminating, every element has at least one normal form.
- If $\rightarrow$ is confluent and terminating, every element has a unique normal form.

# Another Lemma

**Lemma**

If $\rightarrow$ is confluent and terminating, then $x \stackrel{*}{\longleftrightarrow} y$ iff $x\downarrow = y\downarrow$.

*Proof.*

"$\Leftarrow$": Trivial.

"$\Rightarrow$": Suppose $x \stackrel{*}{\longleftrightarrow} y$.

- Because $\rightarrow$ is confluent and terminating, $x$ and $y$ have unique normal forms $x\downarrow$ and $y\downarrow$, respectively.
- Clearly, $x\downarrow \stackrel{*}{\longleftrightarrow} y\downarrow$.
- Because $\rightarrow$ is Church-Rosser, there exists some $z$ such that $x\downarrow \stackrel{*}{\rightarrow} z \stackrel{*}{\leftarrow} y\downarrow$.
- But $x\downarrow$ and $y\downarrow$ are normal forms, so $x\downarrow = z = y\downarrow$.

# Relating $\leftrightarrow_\mathcal{E}$ and $\approx_\mathcal{E}$: Auxiliary Lemma

**Lemma**

Suppose $s, s' \in T(\Sigma, X)$ and $s \rightarrow_\mathcal{E} s'$.

1. Then $\sigma(s) \rightarrow_\mathcal{E} \sigma(s')$ for any substitution $\sigma$ on $T(\Sigma, X)$.
   ($\rightarrow_\mathcal{E}$ is closed under substitution)
2. Then $f(t_1, \ldots, s, \ldots, t_n) \rightarrow_\mathcal{E} f(t_1, \ldots, s', \ldots, t_n)$ for any $n \geq 0$, $f \in \Sigma^{(n)}$, and $t_1, \ldots, t_n \in T(\Sigma, X)$.
   ($\rightarrow_\mathcal{E}$ is closed under $\Sigma$-operations)
3. Then $s \approx_\mathcal{E} s'$.

Proof of (1) and (2): Exercise

# Proof of (3)

We have $s \rightarrow_{\mathcal{E}} s'$. Hence, there exists

- an identity $(l, r) \in \mathcal{E}$,
- a position $p \in \mathcal{P}os(s)$, and
- a substitution $\sigma$

such that $s|_p = \sigma(l)$ and $s' = s[\sigma(r)]_p$.
We have $\sigma(l) \approx_{\mathcal{E}} \sigma(r)$ because $\approx_{\mathcal{E}}$ contains $\mathcal{E}$ and is closed under substitution.
By the following lemma, we finally get $s \approx_{\mathcal{E}} s'$.

**Lemma**

Suppose $R$ is a congruence relation and $s, t \in T(\Sigma, X)$. If $p \in \mathcal{P}os(s)$ and $s|_p \; R \; t$ then $s \; R \; s[t]_p$.

*Proof.* By induction on the length of $p$.

# Relating $\leftrightarrow\varepsilon$ and $\approx\varepsilon$

**Lemma**

$\stackrel{*}{\longleftrightarrow}_{\mathcal{E}} \;=\; \approx_{\mathcal{E}}$

*Proof.*

- "$\stackrel{*}{\longleftrightarrow}_{\mathcal{E}} \;\subseteq\; \approx_{\mathcal{E}}$".
  Suppose $s \stackrel{*}{\longleftrightarrow}_{\mathcal{E}} t$. We show by induction on the length of the chain $s \stackrel{*}{\longleftrightarrow}_{\mathcal{E}} t$ that $s \approx_{\mathcal{E}} t$.
  - length = 0, hence $s = t$.
  - length $> 0$, hence $s \stackrel{*}{\longleftrightarrow}_{\mathcal{E}} s' \longleftrightarrow_{\mathcal{E}} t$. By using our auxiliary lemma, we get $s' \approx_{\mathcal{E}} t$. The IH gives use $s \approx_{\mathcal{E}} s'$. Hence, $s \approx_{\mathcal{E}} t$.

# Proof (cont.)

- "$\approx_{\mathcal{E}} \;\subseteq\; \stackrel{*}{\longleftrightarrow}_{\mathcal{E}}$".
  - By definition, $\stackrel{*}{\longleftrightarrow}_{\mathcal{E}}$ is an equivalence relation and contains $\mathcal{E}$.
  - By using our auxiliary lemma, we show that $\stackrel{*}{\longleftrightarrow}_{\mathcal{E}}$ is closed under substitution and $\Sigma$-operations.
    (The proof is, again, by induction on the length of the chain $s \stackrel{*}{\longleftrightarrow}_{\mathcal{E}} t$.)
  - But $\approx_{\mathcal{E}}$ is defined as the least relation satisfying these properties, so $\approx_{\mathcal{E}} \;\subseteq\; \stackrel{*}{\longleftrightarrow}_{\mathcal{E}}$ must hold.

# Deciding the Word Problem

**Theorem (Deciding the word problem for $\mathcal{E}$)**

If $\mathcal{E}$ is finite and $\rightarrow_{\mathcal{E}}$ is confluent and terminating, then the word problem for $\mathcal{E}$ is decidable.

*Proof.* Suppose $s, t \in T(\Sigma, X)$. We must give an algorithm that decides $s \approx_{\mathcal{E}} t$. Because $s \approx_{\mathcal{E}} t$ and $s \overset{*}{\longleftrightarrow}_{\mathcal{E}} t$ and $s\downarrow_{\mathcal{E}} = t\downarrow_{\mathcal{E}}$ are all equivalent, we only need to give an algorithm for computing the normal form $u\downarrow_{\mathcal{E}}$ for any term $u$.

# Computing Normal Forms (1)

Suppose $\mathcal{E}$ is finite and $\to_{\mathcal{E}}$ is confluent and terminating. Given a term $u \in T(\Sigma, X)$, we can compute the normal form $u{\downarrow_{\mathcal{E}}}$ using the following iteration:

1. Decide if $u$ is already in normal form w.r.t $\to_{\mathcal{E}}$. If yes, stop. Otherwise, continue with step (2).
2. Find some $u'$ such that $u \to_{\mathcal{E}} u'$ (if $u$ is not in normal form). Then continue with step (1), setting $u = u'$.

This iteration terminates because $\to_{\mathcal{E}}$ is terminating.

# Computing Normal Forms (2)

Here is how we decide whether $u$ is in normal form:

- For all identities $(l, r) \in \mathcal{E}$ (only finitely many), and
- all positions $p \in \mathcal{P}os(u)$ (only finitely many)
- check whether there exists a substitution $\sigma$ such that $u|_p = \sigma(l)$. If yes, then we can reduce $u$ to $u[\sigma(r)]_p$. If not, $u$ is already in normal form.

We will see later that finding a substitution $\sigma$ such that $u|_p = \sigma(l)$ is also decidable.