Foundations of Programming Languages and Software Engineering

Universität Freiburg

June 2012

- The Word Problem

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 σ such that $\sigma s \approx_{\mathcal{E}} \sigma t$.

The Word Problem

Definition

Suppose Σ is a signature and X a set of variables disjoint from Σ.

- 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
$\mathcal{E}_{int} = \{\mathtt{pred}(\mathtt{succ}(x)) \approx x, \mathtt{succ}(\mathtt{pred}(x)) \approx x\}$
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:
  - $\mathtt{succ}(\underline{\mathtt{succ}(\mathtt{pred}(\mathtt{zero}))}) \rightarrow_{\mathcal{E}_{int}} \mathtt{succ}(\mathtt{zero})$
- Check whether the resulting terms are identical.

Problem: Applying the reduction rules might not terminate.

An Undecidable Word Problem

Combinatory Logic

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

$\mathcal{E}_C = \{((S \cdot x) \cdot y) \cdot z = (x \cdot z) \cdot (y \cdot z),$
$(K \cdot x) \cdot y = x, I \cdot x = x\}$

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 Σ_C $\Rightarrow$ the word problem for $\mathcal{E}_C$ is undecidable.

Undecidability

The computation of any Turing machine can be simulated by an appropriate signature Σ and set of identities $\mathcal{E} \Rightarrow$ the word problem in general is undecidable.

The Reduction Relation Generated by Σ-Identities

Definition

Let $\mathcal{E}$ be a set of Σ-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 $(l, r) \in \mathcal{E}$, $p \in Pos(s)$, and a substitution σ with $s|_p = \sigma(l)$ and $t = s[\sigma(r)]_p$.

Example

Computing with Groups

$$\Sigma_G = \{e^{(0)}, i^{(1)}, f^{(2)}\}$$

$$\begin{aligned} \mathcal{E}_G = \{ & f(x, f(y, z)) \approx f(f(x, y), z), \\ & f(e, x) \approx x, \\ & f(i(x), x) \approx e\} \end{aligned}$$

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

Composing Relations

Definition

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

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

Example

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

Notations for Reduction Relations

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

$\xrightarrow{0} := \{(x,x) \mid x \in M\}$	identity
$\xrightarrow{i+1} := \rightarrow \circ \xrightarrow{i}$	$(i+1)$-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
$\overset{+}{\leftrightarrow} := (\leftrightarrow)^+$	transitive symmetric closure
$\overset{*}{\leftrightarrow} := (\leftrightarrow)^*$	reflexive transitive symmetric closure

Terminology for Reduction Relations (1)

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

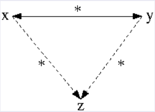
- x is reducible iff there is a $z \in M$ with $x \rightarrow z$.
- x is in normal form iff it is not reducible.
- y is a normal form of x iff $x \xrightarrow{*} y$ and y is in normal form.
- if x has a unique normal form, it is denoted by $x\!\downarrow$.
- x and y are joinable iff there is a $z \in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

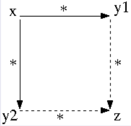
Terminology for Reduction Relations (2)

Definition

A reduction $\rightarrow$ is called

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

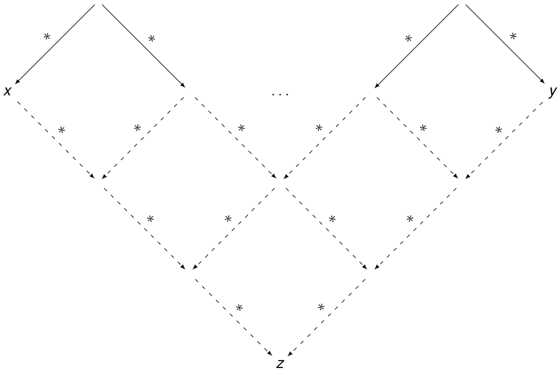




- confluent iff $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$ implies $y_1 \downarrow y_2$,
- semi-confluent iff $y_1 \leftarrow x \overset{*}{\rightarrow} y_2$ implies $y_1 \downarrow y_2$,
- terminating iff there is no infinite chain
  $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \ldots$.
- Normalizing iff every element has a normal form.
- Convergent iff it is both confluent and terminating.

Church-Rosser and Confluence are Equivalent

- It is easy to see that any Church-Rosser relation is confluent.
- If $\rightarrow$ is confluent and $x \overset{*}{\leftrightarrow} y$, 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 $x \downarrow y$.

Auxiliary Lemmas

Lemma

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

Lemma

$\overset{*}{\leftrightarrow}_{\mathcal{E}} \;=\; \approx_{\mathcal{E}}$

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
- We do not give the proof details here, but some important facts are . . .

Existence and Uniqueness of Normal Forms

- If $\to$ is confluent, every element has at most one normal form.
- If $\to$ is terminating, every element has at least one normal form.

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.

$\Rightarrow$ If $\rightarrow$ is confluent and terminating, every element has a unique normal form.

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 $s \approx_{\mathcal{E}} t$. Since $s \approx_{\mathcal{E}} t$ and $s{\downarrow_{\mathcal{E}}} = t{\downarrow_{\mathcal{E}}}$ are equivalent (proof omitted), 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 $\rightarrow_{\mathcal{E}}$ 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 $\rightarrow_{\mathcal{E}}$. If yes, stop. Otherwise, continue with step (2).
2. Find some u' such that $u \rightarrow_{\mathcal{E}} u'$ (if u is not in normal form). Then continue with step (1), setting $u = u'$.

This iteration terminates because $\rightarrow_{\mathcal{E}}$ 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 $p \in Pos(u)$ (only finitely many)
- check whether there exists a substitution σ 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 σ such that $u|_p = \sigma(l)$ is also decidable.