Conditional Rewriting

Bernhard Gramlich

ISR 2009, Brasilia, Brazil, June 22-26, 2009
Outline

- Introduction
- Basics in Conditional Rewriting
- Termination
- Confluence
- Transforming CTRSs Into TRSs
- Systems with Extra Variables (3-Ctrss)
- Transforming CTRSs Into TRSs
- Further Topics (expressive power, modularity, ...)
Lecture 1
Introduction

Examples

- $x \leq z = \text{true} \iff x \leq y = \text{true}, y \leq z = \text{true}$

- $a = b \iff a \neq c, a = c \iff a \neq b$ ($a = b \lor a = c$)

  Hence: minimal models non-unique. Consequence (here): no negation in conditions.

- $f(x, y) \to g(x) \iff P(x, y)$ (built-in predicates, not here)

- $a \to b \iff c = d$

- $a \to b \iff c \iff^* d$

- $a \to b \iff c \downarrow d$

- $a \to b \iff c \to^* d$

- $a \to b \iff a \to^* c$

- $x \leq z \to \text{true} \iff x \leq y \to^* \text{true}, y \leq z \to^* \text{true}$

- $s(x) + y \to s(z) \iff x + y \to^* z$

- $s(x) + y \to s(z) \iff x + y \to^* z$
Basics in Conditional Rewriting

Format of CESs / CTRSs

- **conditional (term) equational system (CES) $R$:** axioms of shape $l = r \iff s_1 = t_1, \ldots, s_n = t_n$; leads to

- **conditional equational logic (CEL):** $=^R$

- **conditional (term) rewrite system (CTRS) $R$:** rules of shape $l \rightarrow r \iff s_1 \square t_1, \ldots, s_n \square t_n$; interpretation of equality $\square$ in conditions:
  - $\square = \leftrightarrow^*$: semi-equational system
  - $\square = \downarrow$: join system
  - $\square = \rightarrow^*$: oriented system
  - $\square = \rightarrow^*$, all $t_1$ $R_u$-irreducible ground terms: **normal** (join) system (where $R_u = \{ l \rightarrow r \mid l \rightarrow r \implies c \in R \}$)
Basics in Conditional Rewriting

Join CTRSs as non-left-linear normal (join) CTRSs

- replace every \( l \rightarrow r \leftarrow s_1 \downarrow t_1, \ldots, s_n \downarrow t_n \) by
  \[ l \rightarrow r \leftarrow eq(s_1, t_2) \rightarrow^* true, \ldots, eq(s_n, t_n) \rightarrow^* true \]
- add \( eq(x, x) \rightarrow true \)
- \( true \) fresh boolean constant, \( eq \) fresh binary predicate
- \( s \rightarrow_R t \) iff \( eq(s, t) \rightarrow_R^* true \) (in many-sorted setting)

Syntactical properties of CTRSs \( R \) (via \( R_u \))

- \( R_u = \{ l \rightarrow r \mid l \rightarrow r \implies c \in R \} \)
- \( R \) left-linear, right-linear, linear, (non-)collapsing,
  (non-)duplicating, non-overlapping, orthogonal, overlaying,
  \ldots, if \( u \) is left-linear, \ldots, respectively
- advantage of these syntactic definitions: easily decidable!
- disadvantage (partially): semantically misleading (Ex.: \( f(x, y) \rightarrow a \leftarrow x \leftrightarrow^* y \))
Basics in Conditional Rewriting

Classification of CTRSs according to extra variables

$VAR(s)$ set of all variables occurring in $s$, $l \rightarrow r \iff c \in R$

- type 1: $VAR(l) \supseteq VAR(r) \cup VAR(c)$ (no extra variables)
- type 2: $VAR(l) \supseteq VAR(r)$ (extra variables at most in cond.)
- type 3: $VAR(l) \cup VAR(c) \supseteq VAR(r)$ (extra variables in right-hand sides must appear in conditions)
- type 4: no restrictions.

- $R_n$-CTRS if all its rules are of type $n$

- type 2 involves existential search in conditions
- type 3 involves existential search in conditions and rhs’s and may cause infinite branching
- type 4 very hard to understand / handle computationally (almost no results known for systems of this type)
Basics in Conditional Rewriting

Induced equality for CES \( E \)

Inference system for conditional equational logic (CEL): Given CES \( E \), CEL defines conditional equational derivability \( \models_E \):

- **Reflexivity:** \( t = t \)
- **Symmetry:** \( s = t \) \( \quad t = s \)
- **Transitivity:** \( s = t, t = u \) \( \quad s = u \)
- **Congruence:** \( f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n) \)
\[ s_1\sigma = t_1\sigma, \ldots, s_n\sigma = t_n\sigma \]
- **Application:** \( l\sigma = r\sigma \)
\[
\text{if } l = r \iff s_1 = t_1, \ldots, s_n = t_n \in E
\]
Basics in Conditional Rewriting

Induced rewrite relation for CTRS $R$
for $R$ consisting of rules of the form $l \rightarrow r \iff s_1 \square t_1, \ldots, s_n \square t_n$
with $\square \in \{\leftrightarrow^*, \downarrow, \rightarrow^*\}$:

- $R_0 = \emptyset$
- $R_{k+1} = \{l \sigma \rightarrow r \sigma \mid l \rightarrow r \iff c \in R, s_i \sigma \square R_k t_i \sigma \text{ for all } s_i \square t_i \text{ in } c\}$.
- $\rightarrow = \rightarrow_R = \bigcup_{i \geq 0} \rightarrow_{R_i}$.
- If $s \rightarrow^*_R t$ with some concrete derivation $s \rightarrow^{R_n}_R t$, the latter has level $n$.
- The depth of a reduction $s \rightarrow^*_R t$ is the minimal (level) $n$ with $s \rightarrow^{R_n}_R t$.

Relationships between different induced rewrite relations

- $\rightarrow, \rightarrow^* \subseteq \rightarrow, \downarrow \subseteq \rightarrow, \leftrightarrow^* \subseteq \equiv_R$
- Logicality: $\leftrightarrow^{*, \leftrightarrow^*} R \sqsupseteq R$ (hence $\leftrightarrow^{*, \leftrightarrow^*} R \sqsupseteq R$)?
Basics in Conditional Rewriting

Examples

For $R = \{a \rightarrow b \iff c \iff^* d; e \rightarrow c; e \rightarrow d\}$:
- $e \rightarrow_{R_1} c$, $e \rightarrow_{R_1} d$
- $a \rightarrow_{R_2} b$, $a \rightarrow_{R_k} b$ for every $k \geq 2$
- $a \not\rightarrow_{R_1} b$, $\text{depth}(a \rightarrow^* b) = 2$

For $R = \{a \rightarrow b \iff c \downarrow d; e \rightarrow c; e \rightarrow d\}$:
- $c, d$ in normal form w.r.t. every $R_k$
- $a \rightarrow_{R_k} b$ for no $k$, hence $a \not\rightarrow_R b$

For $R = \{f(x) \rightarrow a \iff f(x) \downarrow x; b \rightarrow f(b)\}$:
- $b \rightarrow_R f(b) \rightarrow_R a$, since $f(b) \downarrow_R b$ due to $f(b) \leftarrow_R b$
- $f(a) \rightarrow_R a$ iff $f(a) \downarrow_R a$ iff $f(a) \rightarrow_R a$ (since $a$ NF), hence $f(a) \not\rightarrow_R a$
Major Problems and Difficulties

effectiveness in testing conditions

▶ conditions are recursively(!) evaluated
▶ entails in general a possibly non-terminating search process
▶ consequence: basic notions like one-step reduction, one-step reducibility, being a normal form etc. are undecidable in general!

no modular decomposition of steps (non-locality)

▶ $s \rightarrow_R t$ using rule $\rho$ does not imply $s \rightarrow_{\{\rho\}} t$, because verification of conditions may need more rules, possibly the whole system!
▶ major obstacle in extending *modularity results* form TRSs to CTRSs
Major Problems and Difficulties

confluence: variable overlaps may be critical!

- **TRS:** \( l\sigma \rightarrow r\sigma, \sigma \rightarrow \sigma' \implies l\sigma' \rightarrow r\sigma' \) for \( l \rightarrow r \in R \)
- **CTRS:** \( l\sigma \rightarrow r\sigma, \sigma \rightarrow \sigma' \implies l\sigma' \rightarrow r\sigma' \) for \( l \rightarrow r \leftarrow c \in R \)?

In general NO!

- \( s_i\sigma \rightarrow^* t_i\sigma, \sigma \rightarrow \sigma' \), hence: \( s_i\sigma \rightarrow^* s_i\sigma', t_i\sigma \rightarrow^* t_i\sigma' \)
- but: \( s_i\sigma' \downarrow t_i\sigma' \)?

termination: may be non-effective!

- \( R = \{ a \rightarrow b \leftarrow a \rightarrow^* c \} \)
  - terminating (has empty rewrite relation \( \rightarrow_R \))
  - but trying to apply the rule leads to a cycle
- Hence, from a practical point of view,
  - \( R \) should not only be terminating,
  - but also the evaluation of conditions (leading to stronger notions of termination)
Major Problems and Difficulties

realization / implementation of conditional rewriting

- how to realize / implement rewriting with conditional rewrite rules?
- strategies for sequentialization / backtracking?
- how to avoid complicated rewrite machine architecture?

why CTRSs instead of transformed unconditional TRSs?

- conceptually more adequate, intuitive
- any encoding / transformation entails a loss of structural information!
- appropriate encodings / transformations into TRSs are not easy!
More tomorrow!