The Confluence of Ground Term Rewrite Systems is Decidable in Polynomial Time

Hubert Comon

ENS Cachan
France

Guillem Godoy
Robert Nieuwenhuis

Tech. Univ. Catalonia
Barcelona, Spain

Contents

1. Term Rewrite Systems (TRS)
2. Applications
3. Termination and confluence
4. Ground TRS: decision problems
5. What have we solved?
6. How did we solve it? (roughly)
7. Consequences/conclusions

Term Rewrite Systems (TRS)

Sets of rules for replacing (sub)expressions.

Example 1:

$$\begin{array}{rcl} 0 + x & \to & 0 \\ s(x) + y & \to & s(x + y) \\ 0 * x & \to & 0 \\ s(x) * y & \to & y + x * y \end{array}$$

defines $(N, +, *)$. We have:

$$\begin{array}{ll} \underline{s(s(0)) * s(s(0))} & \to \\ s(s(0)) + \underline{s(0) * s(s(0))} & \to \\ s(s(0)) + s(s(0)) + \underline{0 * s(s(0))} & \to^* \\ s(s(s(s(0)))) & \end{array}$$

Example 2:

$$\begin{array}{rcl} a & \to & q_a \\ g(q_a) & \to & q_g \\ g(q_g) & \to & q_g \\ f(q_a, q_a) & \to & q_f \\ f(q_g, q_f) & \to & q_{accept} \end{array}$$

is a tree automaton recognizing $f(g^+(a), f(a, a))$.

Rewriting Theory

Results from and for:

- formal language theory
- algebra
- programming languages:
  - semantics
  - implementation
  - static analysis
  - ...
- automated deduction:
  - decision procedures
  - combination procedures
  - constraint solving
  - symbolic computation
  - ...
- design/verification (e.g., hardware)
- ...

Two basic properties of TRS R:

Termination:
no infinite sequences $t_1 \to_R t_2 \to_R \ldots$

Confluence: (or Church-Rosser p.)
$t_1 \leftarrow_R^* s \to_R^* t_2$ implies $\exists u.\ t_1 \to_R^* u \leftarrow_R^* t_2$

Both are undecidable in general.

Ground Term Rewrite Systems (GTRS)

- GTRS include suffix (or, equivalently, prefix) string rewrite systems
- Word problem PTIME-complete (Kozen '77)
- Termination in PTIME (Plaisted '93)
- Confluence decidable (Dauchet et al '87, Oyamaguchi '87):
  1. build in PTIME tree automaton A_1 accepting $\langle t, u, v \rangle$ such that u and v are reachable from t
  2. A_2 accepting $\langle t, u, v \rangle$ such that u and v are joinable
  3. then R confluent iff $A_1 \subseteq A_2$ (DEXPTIME)

Exact complexity of deciding confluence of GTRS?

Open problem #12 on RTA-LOOP since its creation in '91.

Unknown also for string rewriting.

Between DEXPTIME and P

Important property:

R confluent iff confluent from every lhs l of R:

$$t_1 \leftarrow_R^* l \rightarrow_R^* t_2 \quad \text{implies} \quad \exists u.\ t_1 \rightarrow_R^* u \leftarrow_R^* t_2$$

Some of our failed attempts (since '98):

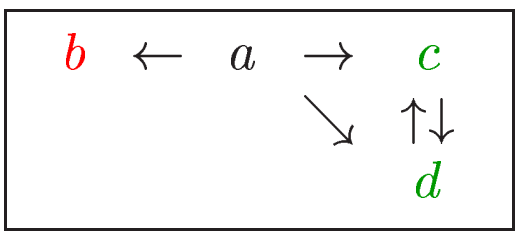
- prove hardness (EXPTIME, (co-)NP, ...)
- build special-purpose automata for the string case
- try to characterize confluence of SRS by properties based on stable prefixes:

  with $b \to ab$, $ab \to b$ and $b \to c$, we have:

  $$acb \to_R acab \to_R acaab \to_R acaac$$

  E.g.: R confluent from l iff cannot pump up two different stable prefixes from l?

This is false:



Last autumn: PTIME algorithm for SRS

- Extremely technical proof
- Based on analysis of stable prefixes. PTIME test for:

  can u become stable in uv by only rewriting on v?
- Apply Plaisted's idea: transform R into

$$\{s \to t \mid s \to_R^* t, \quad s,\ t \text{ subterms of R}\}$$

- then get increasing rewrite derivations: with $a \to b$ (1) and $ab \to ca$ (2)

  $\underline{ab} \to_2 c\underline{a} \to_1 b$ is increasing

  $a\underline{a} \to_1 \underline{ab} \to_2 ca$ is not.
- decide confluence from each l by enumerating enough words reachable from l in increasing derivations.

Now:
PTIME algorithm for full GTRS

First, apply three PTIME confluence-preserving transformations:

- into Curry form, with
  1 single binary symbol f + constants:
  $g(a, h(b), c)$ becomes
  $f(\ f(\ f(g, a),\ f(h, b)\ ),\ \ c\ )$
- into shallow, by adding new constants:
  $f(f(a, b)) \to c$ becomes
  $f(d) \to c, \quad f(a, b) \to d, \quad d \to f(a, b)$
- Plaisted's transformation for terms

Second, give PTIME algorithm for:

can f become stable in $f(s, t)$ by only rewriting on s and t?

(essentially, a reachability problem)

PTIME algorithm for full GTRS (cont.)

Third, check a number of necessary conditions for confluence (this simplifies analysis in fourth step).

Fourth, s and t are deeply joinable (dj) if

$$s \to_R^* s' \text{ and } t \to_R^* t' \quad \text{implies} \quad s' \downarrow t'$$

And note that:

R confluent iff each lhs is dj with itself

Fifth, PTIME test for dj between sides of R

PTIME algorithm for deep joinability

Define DJ-conditions of s and t: necessary for their deep joinability.

Fixpoint construction: (s,t) in some ndj_k if they are not dj

$$ndj_0 = \{\ (s,s') \mid \text{DJ-conditions}(s,s') = \text{false}\ \}$$

$$\begin{aligned} ndj_{i+1} &= ndj_i \\ &\cup \{\ (s,s') \mid l \in le(s),\ \ l' \in le(s'),\ \ (l,l') \in ndj_i\ \} \\ &\cup \{\ (s,s') \mid r \in ri(s),\ r' \in ri(s'),\ (r,r') \in ndj_i\ \} \end{aligned}$$

- where $le(s) \approx$ the set of $f(a,b)$ s.t.
  - $s \to_R^3 f(f(a,b),t)$ and
  - f can become stable in $f(a,b)$.
- similarly for $ri(s)$.

Exact complexity

We prove PTIME-hardness for strings (by LOGSPACE reduction from the alternating graph reachability problem).

Altogether this implies PTIME-completeness for strings and terms.

Conclusions

- picture closed for GTRS
- Surprising: deciding dj is in PTIME, but half deep joinability (hdj):

  given s, t, R, is any term reachable from s joinable with t?

  is EXPTIME-complete!
- confluence $\equiv$ hdj(l, t) for all lhs l and t reachable from l in one step.
  hdj is hard, but the conjunction is not!
- is there a non-trivial class of tree automata for which e.g. inclusion is in P?
- precise degree $|R|^5$
- eliminating transformation to shallow, it could be that:
  - co-NP for left-linear right-ground TRS
  - decidable for right-ground TRS