Complexity Analysis of Precedence Terminating Infinite Graph Rewrite Systems

Naohi Eguchi

Chiba University, Japan
University of Innsbruck, Austria

September 27, 2014
41st TRS meeting, Jozankei, Sapporo, Japan
Introduction

- **Complexity analysis**: Runtime complexity & Normal-form size measured by the *depths* of starting terms.
- **Precedence terminating** TRSs can only express non-recursion schemata.
- **Infinite** precedence terminating TRSs cover all the primitive recursive functions.
  - can be used as a memoisation technique if *graph rewrite systems* (GRSs) are used instead.
Why runtime complexity? Why graph rewriting?

**Derivation height, Runtime complexity**

\[
\text{dh}(t, \rightarrow) = \max\{m \in \mathbb{N} \mid \exists t_1, \ldots, t_m \text{ s.t. } t \rightarrow t_1 \rightarrow \cdots \rightarrow t_m\}
\]

\[
\text{rc}_R(n) = \max\{\text{dh}(t, \rightarrow_R) \mid |t| \leq n \& t = f(s_1, \ldots, s_k)\}
\]

Such a term \(f(s_1, \ldots, s_k)\) is called basic.

**Maxim (Avanzini-Moser RTA ’10, FLOPS ’10)**

Polynomial runtime complexity \(\iff\) polytime computability

follows from: term rewriting \(\iff\) graph rewriting in the simulating GRSs modulo un-sharing + sharing

- The size of a normal form can be exponential since the size of its graph representation is polynomial in the simulating GRS.
- Hence graph rewriting is a more natural framework.
Why infinite graph rewriting? 1/3

\[ g : \text{Btree} \times \text{Btree} \to \text{Btree}, \quad f : \text{Nat} \times \text{Btree} \to \text{Btree} \]

**Example**

\[
\mathcal{R} : \\
g(\epsilon, z) \rightarrow z \\
g(c(x, y), z) \rightarrow c(g(x, z), g(y, z)) \\
f(0, y) \rightarrow \epsilon \\
f(s(x), y) \rightarrow g(y, f(x, y))
\]

Assume a standard interpretation into binary trees.

Let \([x] = \) and \([y] = \)

\[
[g(x, y)] = \\
[f(s^m(0), y)] = (2^{O(m)} \text{ many copies of } [y])
\]
Why infinite graph rewriting? 2/3

Example

\[
\mathcal{R} : \begin{align*}
g(\epsilon, z) & \rightarrow z 
g(c(x, y), z) & \rightarrow c(g(x, z), g(y, z)) 
f(0, y) & \rightarrow \epsilon 
f(s(x), y) & \rightarrow g(y, f(x, y))
\end{align*}
\]

Let \( c^0(\epsilon) = \epsilon, \ c^{n+1}(\epsilon) = c(c^n(\epsilon), c^n(\epsilon)) \).

- \( f(s^m(0), c^n(\epsilon)) \) leads to a normal form of exponential size in a polynomial step, but

- Avanzini-Moser’s maxim still applies if rc\(\mathcal{R}\) is polynomial.

However none of existing methods can show rc\(\mathcal{R}\) is poly. Why?

- \( g \) is duplicated in \( g(c(x, y), z) \rightarrow c(g(x, z), g(y, z)) \).

Hence normalisation of \( f(s^m(0), c^n(\epsilon)) \) needs an exponential step if measured by the depth of \( f(s^m(0), c^n(\epsilon)) \).

- Most methods practically use depth not size.
Why infinite graph rewriting? 3/3

Example

\[ \mathcal{R} : \]

\[
g(\epsilon, z) \rightarrow z \quad g(c(x, y), z) \rightarrow c(g(x, z), g(y, z))
\]

\[
f(0, y) \rightarrow \epsilon \quad f(s(x), y) \rightarrow g(y, f(x, y))
\]

- Difference between depth and size does not make a big change over unary constructors.
- But essential over arbitrary constructors:
  (General primitive recursion)
  \[
f(c(x_1, \ldots, x_k), \bar{y}) \rightarrow h_c(x_1, \ldots, x_k, \bar{y}, f(x_1, \bar{y}), \ldots, f(x_k, \bar{y}))
\]
  The problem cannot be solved by simple sharing.
- A solution: Infinitely instantiation with fully shared graph representation
  \[\Rightarrow\text{ Unfolding graph rewrite rules} \]
  (Dal Lago, Martini and Zorzi '10)
Unfolding graph rewrite rules: Representation of the general primitive recursion with infinite graph rewrite rules (Dal Lago, Martini and Zorzi).

Every unfolding graph rewrite rule is precedence terminating in the sense of Middeldorp, Ohsaki and Zantema.

This work: Precedence termination with argument separation \[\implies\] Polynomial complexity analysis of a subclass of precedence terminating infinite GRSs

which is complete for an extension of usual polytime computable functions to those over arbitrary constructors.
Representing the general primitive recursion with infinite graph rewrite rules.

\[ \begin{align*}
  f(\epsilon, z) & \rightarrow g(z), \\
  f(c(x, y), z) & \rightarrow h(x, y, z, f(x, z), f(y, z))
\end{align*} \]

of general primitive recursion with infinite instances:

\[ \begin{align*}
  f(\epsilon, z) & \rightarrow g(z), \\
  f(c(\epsilon, \epsilon), z) & \rightarrow h(\epsilon, \epsilon, z, g(z), g(z)), \\
  f(c(c(\epsilon, \epsilon), c(\epsilon, \epsilon)), z) & \rightarrow \\
  h(c(\epsilon, \epsilon), c(\epsilon, \epsilon), z, h(\epsilon, \epsilon, z, g(z), g(z)), h(\epsilon, \epsilon, z, g(z), g(z))),& \ldots
\end{align*} \]

- But then graph representation is more convenient.
Note: An infinite set \( G \) of unfolding graph rewrite rules for a fixed signature is uniformly defined,

- independent of each instance and of the underlying TRS.
- Hence \( \exists \) polytime algorithm \( [G] \mapsto [H] \) such that \( G \xrightarrow{\sim} H \).

\[ \begin{array}{c}
 h \\
 c \\
 \varepsilon \\
 z \\
 g \\
 \end{array} \]

**Tiered recursive functions (Dal Lago-Martini-Zorzi)**

Tiered recursive function are an extension of usual polytime functions to those functions over arbitrary constructors.

**Theorem (Dal Lago-Martini-Zorzi ’10)**

\( \forall f: \text{ tiered recursive function} \ \exists G: \text{ infinite GRS defining } f \)

\( \exists p: \text{ poly. } s.t. \ G \xrightarrow{k} H \implies \max\{k, |H|\} \leq p(|G|). \)
Precedence termination

Observe: Unfolding graph rewrite rule is precedence terminating.

Definition (Middeldorp-Ohsaki-Zantema)

Let $\succ$: precedence, well-founded partial order on a signature. A rewrite rule $f(t) \rightarrow r$ is precedence terminating if $f \succ g$ for any $g \in \{ h : \text{function symbol} \mid h \text{ appears in } r \}$.

$f(c(\epsilon, \epsilon), z) \rightarrow h(\epsilon, \epsilon, z, g(z), g(z))$: precedence terminating if $f \succ h$, $f \succ g$ and $f \succ \epsilon$.

- Runtime complexity of precedence terminating finite TRSs is at most exponential.
- Precedence terminating infinite TRSs cover all the primitive recursive functions.
- This work: Restrictive precedence termination which induces polynomial complexity of infinite GRSs.
Separation of argument positions of functions. (Tiered recursion)

**Example**

\[
\mathcal{R} : \\
g(\epsilon; z) \rightarrow z \\
g(c(x, y); z) \rightarrow c(\epsilon; g(x; z), g(y; z)) \\
f(0, y; ) \rightarrow \epsilon \\
f(s(x), y; ) \rightarrow g(y; f(x, y; ))
\]

\[
f(x_1, \ldots, x_k; x_{k+1}, \ldots, x_{k+l}) : \text{called normal arguments of } f.
\]

**Observation.** Starting with a basic term:

- Terms in normal argument positions are always normalised.
- Rewriting occurs only in non-normal positions.
- Note: the argument separation is not always possible.
Definition

Let $>: \text{precedence.}$ A prec. term. graph rewrite rule $L \rightarrow R$ is precedence terminating with argument separation if:

The structure in normal positions unchanged, i.e. $\forall G: \text{normal sub-term graph of } R \exists H: \text{normal sub-term graph of } L \text{ s.t. } G \cong H.$

Theorem

Suppose $\forall L \rightarrow R \in G \text{ GRS prec. terminating with argument sep. :}$

1. $L: \text{maximally shared (modulo argument separation).}$
2. $|R| \leq |L| + m. \ (m: \text{size of subgraphs of } L \text{ connected to normal positions of root}_L)$

Then $\exists p: \text{poly. s.t. } G \rightarrow^k_H \& G: \text{closed basic term graph}$

$$\Rightarrow k \leq p(n) \& |H| \leq p(n) + |G|.$$  

($n: \text{size of subgraphs of } G \text{ connected to normal positions of root}_G$)
Corollary

\( \forall f : \text{tiered recursive function} \ \exists G : \text{infinite GRS defining } f \)

\( \exists p : \text{poly. s.t. } G \xrightarrow{k} H \text{ & } G : \text{closed basic term graph} \)

\[ \Rightarrow k \leq p(n) \& |H| \leq p(n) + |G|. \]

\((n: \text{size of subgraphs of } G \text{ connected to normal positions of root } G)\)

Compare:

Theorem (Dal Lago-Martini-Zorzi ’10)

\( \forall f : \text{tiered recursive function} \ \exists G : \text{infinite GRS defining } f \)

\( \exists p : \text{poly. s.t. } G \xrightarrow{i \leftarrow k} H \rightleftharpoons \max\{k, |H|\} \leq p(|G|). \)

- \(p(n)\) depends only on the size \(n\) of the normal subgraphs of the starting term graph \(G\).
- Innermost rewriting is not necessary as long as \(G\) is a closed basic term graph.
Example

\[ f(x; y) \]
\[ h(x, y; z, u, v) \]
\[ g(; x) \]

- \( L \rightarrow R \): precedence terminating with argument separation if \( f > h, f > g, f > c \) and \( f > \epsilon \).
- \( |L| = 5, |R| = 6, m = 3 \). Hence \( |R| \leq 8 = |L| + m \).

\[ \forall L \rightarrow R \in G, |R| \leq |L| + m \ (m: \text{size of subgraphs of } L \text{ connected to normal positions of root}_L) \]
Conclusion

- Seems not possible to relate the general form of primitive recursion to polynomial runtime complexity.
  \[ f(c(x_1, \ldots, x_k), \vec{y}) \rightarrow h_c(x_1, \ldots, x_k, \vec{y}, f(x_1, \vec{y}), \ldots, f(x_k, \vec{y})) \]

- General primitive recursion can be related to polynomial runtime complexity with an infinite set of unfolding graph rewrite rules.

- This work: Complexity analysis of precedence terminating infinite GRSs based on unfolding graph rewrite rules.

- Seems more appropriate to measure runtime complexity by the depth (not by the size) of a starting term for TRSs over non-simple constructors, e.g. TRSs over tree structures.
References

- **Proving Termination of Unfolding Graph Rewriting for General Safe Recursion**

- **Complexity Analysis of Precedence Terminating Infinite Graph Rewrite Systems**

*Thank you for your listening!*