Left-normal Translation for Applicative Term Rewriting Systems

1910251 Yusuke Suzuki

Term rewriting is a computation model based on directed equations. Sets of such equations are called term rewriting systems (TRSs); especially, TRSs over constant symbols and a single binary function symbol $\circ$, an application symbol, are called applicative term rewriting systems (ATRSs). ATRSs underlie functional programming languages and proof assistants and enable them to model higher-order functions.

Consider the following ATRS $\mathcal{R}$ that computes Fibonacci numbers.

$$
\begin{aligned}
1\colon&\quad + \ \mathsf{0}\ x \to x\\
2\colon&\quad + \ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (+\ x\ y)\\
3\colon&\quad \mathsf{tail}\ (:\ x\ xs) \to xs\\
4\colon&\quad \mathsf{nth}\ (:\ x\ xs)\ \mathsf{0} \to x\\
5\colon&\quad \mathsf{nth}\ (:\ x\ xs)\ (\mathsf{s}\ y) \to \mathsf{nth}\ xs\ y\\
6\colon&\quad \mathsf{zip}\ f\ (:\ x\ xs)\ (:\ y\ ys) \to :\ (f\ x\ y)\ (\mathsf{zip}\ f\ xs\ ys)\\
7\colon&\quad \mathsf{fibs} \to :\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))
\end{aligned}
$$

For instance, the term $\mathsf{nth}\ \mathsf{fibs}\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))$ is rewritten to the second Fibonacci number. However, a naive computation may cause an infinite rewrite sequence like:

$$
\begin{aligned}
&\mathsf{nth}\ \underline{\mathsf{fibs}}\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0})) \to \mathsf{nth}\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \underline{\mathsf{fibs}}\ (\mathsf{tail}\ \mathsf{fibs}))))\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))\\
&\to \mathsf{nth}\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \underline{\mathsf{fibs}}\ (\mathsf{tail}\ \mathsf{fibs}))))\ (\mathsf{tail}\ \mathsf{fibs}))))\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))\\
&\to \cdots
\end{aligned}
$$

whilst another computation yields the finite rewrite sequence:

$$
\begin{aligned}
&\mathsf{nth}\ \underline{\mathsf{fibs}}\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0})) \to \underline{\mathsf{nth}\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))}\\
&\to \underline{\mathsf{nth}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ (\mathsf{s}\ \mathsf{0})} \to \mathsf{nth}\ (\mathsf{zip}\ +\ \underline{\mathsf{fibs}}\ (\mathsf{tail}\ \mathsf{fibs}))\ \mathsf{0}\\
&\to \mathsf{nth}\ (\mathsf{zip}\ +\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))\ (\mathsf{tail}\ \underline{\mathsf{fibs}}))\ \mathsf{0}\\
&\to \mathsf{nth}\ \underline{(\mathsf{zip}\ +\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))}\\
&\qquad \underline{(\mathsf{tail}\ (:\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))))}\ \mathsf{0}\\
&\to \underline{\mathsf{nth}\ (:\ (+\ \mathsf{0}\ (\mathsf{s}\ \mathsf{0}))\ (\mathsf{zip}\ +\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))}\\
&\qquad \underline{(:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))\ \mathsf{0}}\\
&\to \underline{+\ \mathsf{0}(\mathsf{s}\ \mathsf{0})}\\
&\to \mathsf{s}\ \mathsf{0}
\end{aligned}
$$

Underlines indicate rewrite positions. Comparing the two rewrite sequences, they rewrite the same position at their first step, whilst different positions thereafter. In the latter, subterms are *needed* to be rewritten at every underlined position in order to obtain the second Fibonacci number. On the other hand, in the former, subterms are not *needed* to be rewritten at underlined positions but ones at the first step. Such positions that are needed for results of computations are called needed positions, and the strategy that rewrites subterms at needed positions are the needed strategy. However, it is known that needed positions are uncomputable in general [2], and thus we cannot use the strategy readily.

As for another strategy, O'Donnell [3] showed the Normalization Theorem: in left-normal TRSs, the leftmost-outermost strategy always leads a term to a computational result. Left-normality is the property that no function symbols occur on the right of variables in a term. If every left-hand side of a TRS is left-normal, also the TRS is called so. Recalling the ATRS $\mathcal{R}$, the rules 4, 5 and 6 are not left-normal because of the underlined subterms, neither is $\mathcal{R}$.

$$
\begin{aligned}
4\colon&\quad \mathsf{nth}\ (:\ x\ xs)\ \underline{\mathsf{0}} \to x\\
5\colon&\quad \mathsf{nth}\ (:\ x\ xs)\ \underline{(\mathsf{s}\ y)} \to \mathsf{nth}\ xs\ y\\
6\colon&\quad \mathsf{zip}\ f\ (:\ x\ xs)\ \underline{(:\ y\ ys)} \to :\ (f\ x\ y)\ (\mathsf{zip}\ f\ xs\ ys)
\end{aligned}
$$

In this thesis, we propose left-normal translation for ATRSs, which translates non-left-normal ATRSs into left-normal ATRSs and enables us to obtain results of computations with the translated ATRSs. For example, the previous TRS $\mathcal{R}$ is translated into the following left-normal ATRSs.

$$
\begin{aligned}
1\colon&\quad +\ \mathsf{0}\ x \to x\\
2\colon&\quad +\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (+\ x\ y)\\
3\colon&\quad \mathsf{tail}\ (:\ x\ xs) \to xs\\
4\colon&\quad \mathsf{fibs} \to :\ \mathsf{0}\ (:\ (\mathsf{s}\ \mathsf{0})\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\\
5\colon&\quad \mathsf{nth}\ (:\ x\ xs)\ y \to \mathsf{nth}_1\ y\ x\ xs\\
6\colon&\quad \mathsf{zip}\ f\ xs\ ys \to \mathsf{zip}_1\ xs\ f\ ys\\
7\colon&\quad \mathsf{nth}_1\ \mathsf{0}\ x\ xs \to x\\
8\colon&\quad \mathsf{nth}_1\ (\mathsf{s}\ y)\ x\ xs \to \mathsf{nth}\ xs\ y\\
9\colon&\quad \mathsf{zip}_1\ (:\ x\ xs)\ f\ ys \to \mathsf{zip}_2\ ys\ x\ xs\ f\\
10\colon&\quad \mathsf{zip}_2\ (:\ y\ ys)\ x\ xs\ f \to :\ (f\ x\ y)\ (\mathsf{zip}\ f\ xs\ ys)
\end{aligned}
$$

With this left-normal ATRS, we get obtain the term $\mathsf{s}\ \mathsf{0}$ from $\mathsf{nth}\ \mathsf{fibs}\ (\mathsf{s}\ (\mathsf{s}\ \mathsf{0}))$

by the leftmost-outermost strategy.

$$
\begin{aligned}
&\mathsf{nth}\ \underline{\mathsf{fibs}}\ (\mathsf{s}\ (\mathsf{s}\ 0))\\
&\to \underline{\mathsf{nth}\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))\ (\mathsf{s}\ (\mathsf{s}\ 0))}\\
&\to \underline{\mathsf{nth}_1\ (\mathsf{s}\ (\mathsf{s}\ 0))\ 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))}\\
&\to \underline{\mathsf{nth}\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ (\mathsf{s}\ 0)}\\
&\to \underline{\mathsf{nth}_1\ (\mathsf{s}\ 0)\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))}\\
&\to \mathsf{nth}\ (\underline{\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})})\ 0\\
&\to \mathsf{nth}\ (\mathsf{zip}_1\ \underline{\mathsf{fibs}}\ +\ (\mathsf{tail}\ \mathsf{fibs}))\ 0\\
&\to \mathsf{nth}\ (\underline{\mathsf{zip}_1\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))\ +\ (\mathsf{tail}\ \mathsf{fibs})})\ 0\\
&\to \mathsf{nth}\ (\mathsf{zip}_2\ (\mathsf{tail}\ \underline{\mathsf{fibs}})\ 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ +)\ 0\\
&\to \mathsf{nth}\ (\mathsf{zip}_2\ (\underline{\mathsf{tail}\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))})\\
&\qquad 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ +)\ 0\\
&\to \mathsf{nth}\ (\underline{\mathsf{zip}_2\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ 0\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ +})\ 0\\
&\to \underline{\mathsf{nth}\ (:\ (+\ 0\ (\mathsf{s}\ 0))}\\
&\qquad \underline{(\mathsf{zip}\ +\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs}))))\ 0}\\
&\to \underline{\mathsf{nth}_1\ 0\ (+\ 0\ (\mathsf{s}\ 0))}\\
&\qquad \underline{(\mathsf{zip}\ +\ (:\ (\mathsf{s}\ 0)\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))\ (\mathsf{zip}\ +\ \mathsf{fibs}\ (\mathsf{tail}\ \mathsf{fibs})))}\\
&\to \underline{+\ 0\ (\mathsf{s}\ 0)}\\
&\to \mathsf{s}\ 0
\end{aligned}
$$

Meanwhile, this translation moves needed positions to leftmost-outermost positions, which the leftmost-outermost strategy rewrites. Hence, by the translation, we can simulate the needed strategy by the leftmost-outermost strategy.

There is an existing work: left-normal translation was originally developed by Hashida [1]. Hashida's translation translates constructor systems to left-normal constructor systems. This is the pioneering work using an approach that simulates the needed strategy by the leftmost-outermost strategy. Our study is aimed to extend Hashida's translation to ATRSs and to enlarge the class of TRSs that can be left-normal.

Our contribution is two-fold. Firstly, we establish left-normal translation for ATRSs, and then realise simulating the needed strategy by the leftmost-outermost strategy in computations of applicative terms. Secondly, we show that our translation for ATRSs includes Hashida's translation [1]: functional TRSs that can be translated by left-normal translation for functional TRSs

can be translated by left-normal translation for ATRSs after currying, a procedure that translates functional TRSs into ATRSs. The translation also can handle ATRSs including higher-order function. We therefore succeed in extending the class of TRSs with which terms can be computed by the needed strategy.

References

[1] A. Hashida. Transformation-based normalization analysis for term rewriting. Master's thesis, JAIST, 2019.

[2] G. Huet and J.-J. Lévy. Computations in orthogonal rewriting systems, II. In *Computational Logic – Essays in Honor of Alan Robinson*, pages 395–414. The MIT Press, 1991.

[3] M. J. O'Donnell. Computing in systems described by equations. 1977.