Synthesizing programs by Knuth-Bendix completion

Hiromi Yoshimasa

School of Information Science,
Japan Advanced Institute of Science and Technology

February 15, 1999

Keywords: deforestation, fusion rule, term rewriting system, Knuth-Bendix completion, program transformation .

Deforesrtation method proposed by P.Wadler (1990) eliminates useless intermediate data structures with the foldl/build rules in functional programming. As a program transformation method for functional programming languages, deforestation using fusion transformation is effective. F.Bellegarde (1995) indicates that fusion rules for deforestation can be expressed by term rewriting systems. This fusion rule is to perform deforestation on constructor-based system R , it is enough to search for fusion terms in the right-hand side of definition rules, then to build a fusion rule r and finally to synthesize the constructor based system R_h of the fresh symbol h. This system generates a fusion rule $s \to h(x_1, x_2, ..., x_n)$, which is defined by a rewrite rule.

Automating synthesis of programs for deforesrtation proposed by F.Bellegarde introduced fusion algorithm as completion procedure. Completion procedure is good at synthesizing a constructor-based equatinal definition of a function h from a synthesis rule which specifies how h is defined in terms of other functions. h is defined by a composition of other functions that generates intermediate data structures for deforestation. In this method the term $h(x_1, x_2, ..., x_n)$ is put in the left-hand side and the term t to be synthesized in the right-hand side of an equation $h(x_1, ..., x_n) = t$. Let this equation be E. This equation E must be transformed into a rewrite rule R. In this research we collect examples of equations E and good ordering, which can be transformed into rewrite rules. Each successful example produces a rewrite rule, obtained from the equation E by exchanging its both-hand sides, and synthesized rewrite rules produced by Knuth-Bendix completion procedure. Our reseach results indicate that fusion transformation for term rewriting systems can be realized based on completion procedure. Synthesizing programs by Knuth-Bendix completion is illustrated as follows

R : { Term rewriting rule }

$$\begin{array}{c} E : h(x) = t \\ \Downarrow \quad \text{good ordering} \\ \text{Fusion rule } R_h : t \rightarrow h(x) \\ \\ R \cup R_h \\ \\ \Downarrow \quad \text{Knuth-Bendix completion} \\ \begin{cases} h([]) \rightarrow t_1 \\ h(x :: xs) \rightarrow t_2 \end{cases} \end{array}$$

Knuth-Bendix completion procedure is implemented with standard ML of New Jersey(1994). According to this research result using Knuth-Bendix completion, restriction of fusion rule defined by F.Bellegarde can be removed. For example, the fusion term t of a fusion rule R_h must be linear, but it is able to synthesize $x + x \rightarrow d(x)$, which is derived from the equation $d(x) = x + x$, whose right-hand side is nonlinear. Experiments in Knuth-Bendix completion has shown that a fusion rule R_h can be synthesized from a nonlinear fusion term. In this experiment, program synthesis using deforestation method by Knuth-Bendix completion uses functional sets R_{length} , R_+ and $R_\times$. They are represented as a set of recursive programs. Using fresh symbol h with some function, and making equation E, synthesizing these functional programs, then computing experiment in Knuth-Bendix completion procedure with such the set of recursive programs.

This experiment in completion uses lexicographic path ordering for functional symbols. During this experiment, we learn that differnt orderings lead to diffrent results classified into the following three cases : Success after some steps, failure in the middle of completion, and non-termination with endless loop. In order to find when the completion procedure succeeds, we experimented 731 cases with diffrent orderings over different 46 programs. As a result of this experiment, completion succeeded for input data of 121 cases. Each output data gives a generation rule. A result of new generation rules are reported in this thesis.

From the experiment result of program synthesis, we consider orderings. When a ordering of the fresh symbol h is the biggest of functional symbols, these cases apt to succeed in completion. Successful examples of completions are in different complete steps for diffrent orderings. As mentioned before, ordering influences completing synthesis of programs. We need to carefully controll a suitable ordering in order to succeed program synthesis.

For example, Consider an equation $E : h(x, y, z) = length((x@y) :: z)$. This equation E is transformed into a rewrite rule $R_h : length((x@y) :: z) \rightarrow h(x, y, z)$ by Knuth-Bendix completion procedure. Then we synthesize $R_+ \cup R_{length} \cup R_@ \cup R_h$. When ordering is $h > length > @ > + > [] > :: > 0 > s$, program synthesis succeeds by 10 steps completion.

Results of output data are as follows.

- $h(x :: y, z, w) \to length((x :: (y@z)) + w)$
- $h(x, y, z) \to length(s(x@y) + z)$
- $h(x, y, 0) \to length(x@y)$

From the result of experiments we introduced 27 successful examples and 3 failure examples in this thesis.