Compression of Rewriting Systems for Termination Analysis

Johannes Waldmann

1 Introduction

In the context of automated analysis of termination of term rewriting, the following computation is fundamental:

> Given an interpretation of function symbols of a ranked signature by linear functions, compute efficiently the interpretations of a set of terms over that signature.

A k-ary linear function f is given as $f(x_1, \dots, x_k) = F_1 \cdot x_1 + \dots + F_k \cdot x_k + F_0$. We usually consider vector-valued functions, so $F_1, \dots, F_k$ are matrices over some semiring, and F_0 is a vector. The interpretation of a term t containing variables $v_1, \dots, v_k$ is again a k-ary linear function. We are interested in the terms that are the left-hand sides and right-hand sides of the rewrite rules, because we want to compare the coefficients of their interpretations.

The computations are symbolic in the sense that all coefficients are represented by expressions in some constraint program, and a constraint solver is used to compute their actual values. Here, we are not concerned with the solving, but just want to ensure that the constraint program is short.

We apply the following model of computation: a *straight line program* P over a partially ordered set of variables $(V, <)$ is a sequence of statements (definitions) $c = a + b$ or $c = a \cdot b$ where $a, b, c \in V$ and $a < c$ and $b < c$. The minimal elements of $(V, <)$ are called *inputs*.

We say that P *computes an interpretation* if for a signature Σ, and linear interpretation $[\cdot]$, and set of terms T, the inputs of P are the coefficients of the interpretations of function symbols, and for each $t \in T$, each coefficient c of $[t]$ is represented by some variable in P.

Here, "is represented by" means that the corresponding expressions are equivalent modulo the semiring axioms (addition is associative and commutative, multiplication is associative, addition distributes over multiplication). (TODO: special role of Zero and One?)

The *cost* of P is the number of multiplication statements. (TODO: 1. cost of multiplication depends on matrix dimensions, these dimensions depend on sortedness of signature, 2. addition may have nonzero cost)

One solution of the compression problem is implemented in the termination prover Matchbox, (compressor source code is now available separately from `http://dfa.imn.htwk-leipzig.de/cgi-bin/gitweb.cgi?p=tpdb.git;a=blob;f=TPDB/Compress.hs;hb=HEAD` and described in [EWZ08] (Section 7.4). It is equivalent to a greedy construction of a linear acyclic context free tree grammar in Chomsky normal form, cf. [LMSS09].

References

[EWZ08] Jörg Endrullis, Johannes Waldmann, and Hans Zantema. Matrix interpretations for proving termination of term rewriting. *J. Autom. Reasoning*, 40(2-3):195–220, 2008.

[LMSS09] Markus Lohrey, Sebastian Maneth, and Manfred Schmidt-Schauß. Parameter reduction in grammar-compressed trees. In Luca de Alfaro, editor, *FOSSACS*, volume 5504 of *Lecture Notes in Computer Science*, pages 212–226. Springer, 2009.