Synthesizing Matrix Interpretations via Backward Completion[*]

Dieter Hofbauer

ASW – Berufsakademie Saarland, Germany
d.hofbauer@asw-berufsakademie.de

Abstract

Various approaches to automatically synthesizing termination proofs via matrix interpretations are used in state-of-the-art provers, most notably those based on satisfiability solving. In this talk, an alternative idea for the particular case of string rewriting is presented, exploiting the view of matrix interpretations as weighted automata. A demo of a prototype implementation will show the utility of this approach, as well as its limitations.

1998 ACM Subject Classification F.3.1 Specifying and Verifying and Reasoning about Programs

Keywords and phrases Termination, string rewriting, matrix interpretations

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Matrix interpretations for string rewriting interpret the free monoid of strings in a ring structure, where concatenation of factors corresponds to multiplication and replacement of factors corresponds to subtraction. For termination, we use an (infinite) ordered ring, which is well-founded on its *positive cone*, see [1].

▶ **Example 1.** In $(\mathbb{Z}, 0, 1, +, \cdot)$ we can prove termination of $\{aba \to aa\}$ by the interpretation $i : a \mapsto 1, b \mapsto 2$ as $i(aba \to aa) = i(aba) - i(aa) = i(a) \cdot i(b) \cdot i(a) - i(a) \cdot i(a) = 1 > 0$, but neither $\{ab \to ba\}$ nor $\{aa \to aba\}$ can be proven terminating in $\mathbb{Z}$ as multiplication is commutative and due to the totality of the ordering employed, respectively.

For that reason, we use non-commutative rings with a non-total ordering, in this case rings of square matrices over the natural numbers; for definitions and notations we refer to [1].

▶ **Example 2.** For proving termination of $\{ab \to ba\}$ consider the E_2-interpretation $i : a \mapsto \left(\begin{smallmatrix}1&1\\0&1\end{smallmatrix}\right), b \mapsto \left(\begin{smallmatrix}1&0\\0&2\end{smallmatrix}\right)$, where $i(ab \to ba) = i(ab) - i(ba) = i(a) \cdot i(b) - i(b) \cdot i(a) = \left(\begin{smallmatrix}1&2\\0&2\end{smallmatrix}\right) - \left(\begin{smallmatrix}1&1\\0&2\end{smallmatrix}\right) = \left(\begin{smallmatrix}0&1\\0&0\end{smallmatrix}\right) \in P_2$, and for proving termination of $\{aa \to aba\}$ the E_1-interpretation $i : a \mapsto \left(\begin{smallmatrix}1&1\\1&0\end{smallmatrix}\right), b \mapsto \left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)$ gives $i(aa \to aba) = i(aa) - i(aba) = \left(\begin{smallmatrix}2&1\\1&1\end{smallmatrix}\right) - \left(\begin{smallmatrix}1&1\\1&1\end{smallmatrix}\right) = \left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right) \in P_1$.

Various approaches for synthesizing matrix interpretations have been proposed, for instance complete enumeration of restricted matrix shapes, random guesses for small matrix dimensions, evolutionary programming, and, most prominently, constraint solving. In this talk, we present *backward completion* as another approach for the same purpose. This is related to *forward completion* procedures for match-bound termination proofs as in [2].

[*] This work was partially supported by a travel grant from the Japan Advanced Institute of Science and Technology (JAIST).

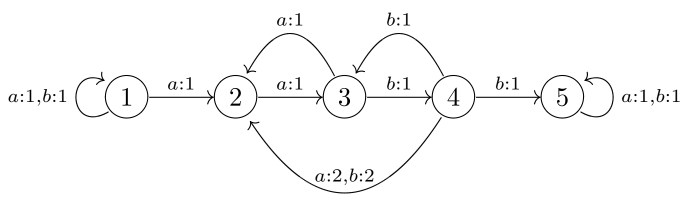
2 Matrix interpretations as weighted automata

Matrix interpretations correspond to *weighted automata* in a direct way. A *weighted automaton* is a mapping $\text{weight} : Q \times \Sigma \times Q \to \mathbb{N}$, where Q is a finite set (of *states*). This mapping is extended to $\text{weight} : Q \times \Sigma^* \times Q \to \mathbb{N}$ by multiplying weights along a single path and adding weights of different paths. A transition from state p to state q with weight n for letter a corresponds to $i(a)_{p,q} = n$ in a matrix interpretation, and vice versa.

▶ **Example 3.** The matrix interpretation

$$a \mapsto \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \qquad b \mapsto \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

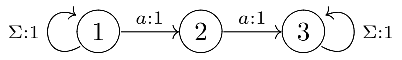
of type $E_{\{1,5\}}$ corresponds to the following weighted automaton, proving termination of $\{aabb \to bbbaaa\}$ (TPDB problem `z001`, see [3], known as *Zantema's problem*):



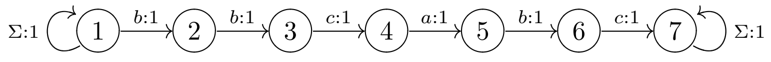
3 Backward completion

Often, automata of a particularly simple form can prove termination of string rewriting systems. These *straight-line automata* essentially consist of a single path that corresponds to the left-hand side of a rule, where each transition has weight 1.

▶ **Example 4.** For $\{aa \to aba\}$, the following automaton proves termination:



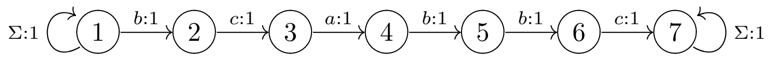
▶ **Example 5.** In the same way, for $\{bbcabc \to abbcbca\}$ (TPDB problem z061) the automaton



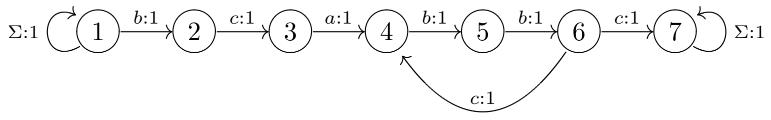
serves as a termination certificate.

Of course, straight-line automata fail in many cases.

▶ **Example 6.** Consider $\{bcabbc \to abcbbca\}$ (TPDB problem z062). For the automaton



we get $\text{weight}(1, bcabbc, 4) = 0 \not\geq 1 = \text{weight}(1, abcbbca, 4)$, and the termination proof fails. However, if we compensate for that defect by adding an edge



we are done: $\text{weight}(1, bcabbc, 4) = 1 = \text{weight}(1, abcbbca, 4)$.

This motivates *backward completion*. Define a pair of states (s,t) with $\mathrm{weight}(s,\ell,t) < \mathrm{weight}(s,r,t)$ to be a *defect* for the rewriting rule $\ell \to r$. Compensate for such a defect by adding a path for ℓ from s to t. In general, for a *backward completion step* choose a defect (s,t) for some rule $\ell \to r$ and a factorization $\ell = \ell_1\ell_2\ell_3$ so that, for some states p, q,

$$\begin{tikzcd} s \arrow[r, "\ell_1"] & p & q \arrow[r, "\ell_3"] & t \end{tikzcd}$$

and add a path of weight 1 from p to q for ℓ_2:

$$\begin{tikzcd} s \arrow[r, "\ell_1"] & p \arrow[r, "\ell_2{:}1"] & q \arrow[r, "\ell_3"] & t \end{tikzcd}$$

4 Implementation

One possiblity is to implement backward completion as a probabilistic algorithm, as follows.

(1) Choose as start automaton $\begin{tikzcd} 0 \arrow[loop left, "\Sigma{:}1"] & 1 \arrow[loop right, "\Sigma{:}1"] \end{tikzcd}$ with just two states.
(2) Randomly choose the left-hand side ℓ of a rule $\ell \to r$ and add a path from 0 to 1 for ℓ, resulting in $\begin{tikzcd} 0 \arrow[loop left, "\Sigma{:}1"] \arrow[r, "\ell{:}1"] & 1 \arrow[loop right, "\Sigma{:}1"] \end{tikzcd}$. Note that now $\mathrm{weight}(0,\ell,1) > \mathrm{weight}(0,r,1)$ unless ℓ is a factor of r, in which case the rewriting system is already non-terminating.
(3) As long as defects exist and $\mathrm{weight}(0,\ell,1) > \mathrm{weight}(0,r,1)$ keeps holding true, randomly perform some backward completion step.

This might fail or go on forever, but in case the procedure stops successfully, the resulting automaton provides a termination certificate.

▶ **Example 7.** The automaton in example 3 can easily be synthesized in this way. Futher examples include rewriting systems from the TPDB that haven't been solved during the 2013 termination competition (see `termination-portal.org/wiki/Termination_Competition/`).

5 Extensions

In the talk, we present various variants of this strategy, in particular a non-probabilistic variant where completion steps are enumerated under a breadth-first strategy. Building on results from [4], a specialized strategy aims at automatically proving polynomial upper bounds on derivation lengths. Backward completion can be easily adapted to the setting of relative termination proofs, thereby considerably strengthening its applicability. A demo implemention shows the utility of the approach, as well as its limitations.

References

1 D. Hofbauer, J. Waldmann. *Termination of string rewriting with matrix interpretations.* Proc. 17th Int. Conf. on Rewriting Techniques and Applications (RTA), pp. 328–342, 2006.
2 A. Geser, D. Hofbauer, J. Waldmann, H. Zantema. *Finding finite automata that certify termination of string rewriting.* Int. J. Found. Comput. Sci. 16(3):471–486, 2005.
3 The Termination Problems Data Base, `termination-portal.org/wiki/TPDB/`.
4 J. Waldmann. *Polynomially bounded matrix interpretations.* Proc. 21st Int. Conf. on Rewriting Techniques and Applications (RTA), pp. 357–372, 2010.