Theoretical Computer Science 635 (2016) 1–15

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Weighted restarting automata and pushdown relations ☆

Qichao Wang, Friedrich Otto

Fachbereich Elektrotechnik/Informatik, Universität Kassel, 34109 Kassel, Germany

ARTICLE INFO

Article history:
Received 4 November 2015
Received in revised form 11 March 2016
Accepted 28 April 2016
Available online 7 May 2016
Communicated by D. Perrin

Keywords:
Weighted restarting automaton
Restarting transducer
Pushdown relation
Almost-realtime pushdown relation

ABSTRACT

Weighted restarting automata have been introduced to study quantitative aspects of computations of restarting automata. Here we study the special case that words over a given (output) alphabet are assigned as weights to the transitions of a restarting automaton. In this way the automaton is extended to define a mapping from the words over its input alphabet into the semiring of formal languages over a given (output) alphabet, generalizing the restarting transducers introduced by Hundeshagen (2013) [7]. We establish that the monotone restarting transducers that are allowed to use auxiliary symbols characterize the class of almost-realtime pushdown relations, and we characterize the deterministic pushdown functions by a particular type of deterministic monotone restarting transducer. Further, we show that already linearly bounded (word-)weighted monotone restarting automata that use auxiliary symbols are more expressive than the corresponding restarting transducers, both in the deterministic as well as in the nondeterministic case. Finally, we prove that for (word-)weighted monotone restarting automata with auxiliary symbols, the variant that may keep on reading after performing a rewrite step (the so-called RRWW-automaton) is strictly more expressive than the variant that must restart immediately after performing a rewrite step (the so-called RWW-automaton), which again holds in the deterministic as well as in the nondeterministic case. This is the first time that a version of the monotone RRWW-automaton is shown to differ in expressive power from the corresponding version of the monotone RWW-automaton.

1. Introduction

Analysis by reduction is a linguistic technique that is used to check the correctness of sentences of natural languages through sequences of local simplifications. The *restarting automaton* was invented as a formal model for the analysis by reduction [1]. In order to study quantitative aspects of computations of restarting automata, *weighted restarting automata* were introduced in [2]. These automata are obtained by assigning an element of a given semiring S as a weight to each transition of a restarting automaton. Then the product (in S) of the weights of all transitions that are used in a computation yields a weight for that computation, and by forming the sum over the weights of all accepting computations for a given input $w \in \Sigma^*$, a value from S is assigned to w. Thus, a partial function $f : \Sigma^* \dashrightarrow S$ is obtained. Here we consider the special case that S is the semiring of formal languages over some finite (output) alphabet Δ. Then f is a transformation from Σ^* into the languages over Δ. Thus, we obtain a generalization of the notion of a *restarting transducer* as introduced in [3].

☆ Some of the results of this paper have been announced at the 6-th International Conference on Algebraic Informatics, CAI 2015, in Stuttgart, Germany, September 2015. An extended abstract appeared in the proceedings of that conference.

E-mail addresses: wang@theory.informatik.uni-kassel.de (Q. Wang), otto@theory.informatik.uni-kassel.de (F. Otto).

http://dx.doi.org/10.1016/j.tcs.2016.04.038
0304-3975/

It is well known (see, e.g., [4]) that the class of languages that are accepted by monotone RWW- and RRWW-automata[1] (see Section 2 for the definitions) coincides with the class of context-free languages, and that the class of languages that are accepted by deterministic monotone RWW- and RRWW-automata coincides with the class of deterministic context-free languages. Accordingly, we are interested in the classes of transformations that are computed by various types of weighted restarting automata that are *monotone*. In this paper we compare some of these classes to each other and we relate them to the class of *pushdown relations* and some of its subclasses. In particular, we prove that monotone RWW- and RRWW-transducers characterize the class of almost-realtime pushdown relations and that deterministic monotone RWW-transducers compute exactly the deterministic pushdown functions, while the deterministic monotone RRWW-transducers can actually compute more functions. Then we establish that already linearly bounded (word-)weighted monotone RWW- and RRWW-automata are strictly more expressive than the corresponding types of restarting transducers, both in the deterministic as well as in the nondeterministic case. Further, we prove that (word-)weighted monotone RRWW-automata compute strictly more transformations than (word-)weighted monotone RWW-automata, which again holds in the deterministic as well as the nondeterministic case. This is the first instance that a version of the monotone RRWW-automaton is shown to differ in expressive power from the corresponding version of the monotone RWW-automaton.

This paper is structured as follows. In Section 2 we recall some basic notions concerning weighted restarting automata, and in Section 3 we look at the pushdown relations and some of their subclasses. Then, in Section 4 we study the classes of transformations that are computed by monotone restarting transducers, and in Section 5 we investigate the computational power of (word-)weighted RWW- and RRWW-automata that are monotone. The paper closes with a short summary and some problems for future work.

2. Weighted restarting automata

We assume that the reader is familiar with the standard notions and concepts of theoretical computer science, such as monoids, finite automata, and semirings. Throughout the paper we will use $|w|$ to denote the length of a word w and λ to denote the empty word. Further, $\mathbb{P}(X)$ denotes the power set of a set X, and $\mathbb{P}_{\text{fin}}(X)$ denotes the set of all finite subsets of X.

A *restarting automaton* (or RRWW-automaton) is a nondeterministic machine with a finite-state control, a flexible tape with endmarkers, and a read/write window of a fixed finite size. Formally, it is described by an 8-tuple $M = (Q, \Sigma, \Gamma, ¢, \$, q_0, k, \delta)$, where Q is a finite set of states, Σ is a finite input alphabet, Γ is a finite tape alphabet containing Σ, the symbols $¢, \$ \notin \Gamma$ are used as markers for the left and right border of the work space, respectively, $q_0 \in Q$ is the initial state, $k \geq 1$ is the size of the *read/write window*, and δ is the *transition relation*, which is specified as follows. Let $\mathcal{PC}^{(0)} = \{\lambda\}$ and, for $i \geq 1$, $\mathcal{PC}^{(i)} = (¢ \cdot \Gamma^{i-1}) \cup \Gamma^i \cup (\Gamma^{\leq i-1} \cdot \$) \cup (¢ \cdot \Gamma^{\leq i-2} \cdot \$)$, where $\Gamma^{\leq i} = \bigcup_{j=0}^{i} \Gamma^j$ and $\mathcal{PC}^{\leq(k-1)} = \bigcup_{i=0}^{k-1} \mathcal{PC}^{(i)}$. Then $\mathcal{PC}^{(k)}$ is the set of *possible contents* of the read/write window of M, and

$$\delta \subseteq Q \times \mathcal{PC}^{(k)} \times ((Q \times (\{\mathsf{MVR}\} \cup \mathcal{PC}^{\leq(k-1)})) \cup \{\mathsf{Restart}, \mathsf{Accept}\}),$$

where MVR, Restart, and Accept denote the move-right, restart, and accept operations. As seen from the above specification, δ may contain four different types of transition steps:

(1) A *move-right step* has the form (q, u, q', MVR), where $q, q' \in Q$ and $u \in \mathcal{PC}^{(k)}$, $u \neq \$$. If M is in state q and sees the string u in its read/write window, then this move-right step causes M to shift the read/write window one position to the right and to enter state q'. However, if the content u of the read/write window is only the symbol \$, then no move-right step is possible.
(2) A *rewrite step* has the form (q, u, q', v), where $q, q' \in Q$, $u \in \mathcal{PC}^{(k)}$, $u \neq \$$, and $v \in \mathcal{PC}^{\leq(k-1)}$ such that $|v| < |u|$. It causes M to replace the content u of the read/write window by the string v, and to enter state q'. Further, the read/write window is placed immediately to the right of the string v. However, some additional restrictions apply in that the border markers ¢ and \$ must not disappear from the tape nor that new occurrences of these markers are created. Further, the read/write window must not move across the right border marker \$, that is, if the string u ends in \$, then so does the string v, and after performing the rewrite operation, the read/write window is placed on the \$-symbol.
(3) A *restart step* has the form $(q, u, \mathsf{Restart})$, where $q \in Q$ and $u \in \mathcal{PC}^{(k)}$. It causes M to move its read/write window to the left end of the tape, so that the first symbol it contains is the left border marker ¢, and to reenter the initial state q_0.
(4) An *accept step* has the form (q, u, Accept), where $q \in Q$ and $u \in \mathcal{PC}^{(k)}$. It causes M to halt and accept.

For some $q \in Q$ and $u \in \mathcal{PC}^{(k)}$, if there is no operation op such that $(q, u, op) \in \delta$, then M necessarily halts in a corresponding situation, and we say that M *rejects* in this case. Further, the letters in $\Gamma \setminus \Sigma$ are called *auxiliary symbols*.

[1] Here we follow the notation of [1,4]. The prefix R- denotes a restarting automaton for which each rewrite step is combined with a restart operation, while the prefix RR- denotes a restarting automaton for which rewrite and restart steps are separate operations. The suffix -WW denotes those restarting automata that can perform rewrite steps (instead of just deletions) and that may use non-input symbols in their rewrite steps.