Theoretical Computer Science 610 (2016) 121–132

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Non-recursive trade-offs between two-dimensional automata and grammars

Daniel Průša

Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University, Zikova 1903/4, 166 36 Prague 6, Czech Republic

ARTICLE INFO

Article history:
Received 31 October 2014
Received in revised form 15 April 2015
Accepted 22 May 2015
Available online 28 May 2015

Keywords:
Picture languages
Four-way automata
Two-dimensional context-free grammars
Descriptional complexity
Non-recursive trade-offs

ABSTRACT

We study succinctness of descriptional systems for picture languages. Basic models of two-dimensional finite automata and generalizations of context-free grammars are considered. They include the four-way automaton of Blum and Hewitt, the two-dimensional online tessellation automaton of Inoue and Nakamura and the context-free Kolam grammar of Siromoney et al. We show that non-recursive trade-offs between the systems are very common. Basically, each separation result proving that one system describes a picture language which cannot be described by another system can usually be turned into a non-recursive trade-off result between the systems. These findings are strongly based on the ability of the systems to simulate Turing machines.

1. Introduction

Many concepts and techniques from the theory of formal languages have been generalized to two-dimensional (2D) languages, where the basic entity, the *string*, has been replaced by a rectangular array of symbols, called a *picture*. The *four-way finite automaton* of Blum and Hewitt [1] was introduced already in 1967. It has a finite-state control unit and a head that traverses the input picture by performing movements right, left, up, and down.

Another notable device is the *two-dimensional online tessellation automaton* (2OTA) proposed by Inoue and Nakamura [2]. This is a restricted nondeterministic cellular automaton where a "transition wave" passes once diagonally across the cells. Its recognition power coincides with the power of tiling systems [3] which are the basis for the well known family of recognizable picture languages (REC) of Giammaresi and Restivo [4].

The early models of picture grammars include matrix and Kolam grammars of Siromoney et al. [5,6]. Kolam grammars were independently proposed and studied by Matz [7] and also by Schlesinger [8,9] who designed them as a tool for structural pattern recognition. The grammars are characterized by the form of productions which resembles the Chomsky normal form. Two extensions of the grammars are known – the first one described by Průša [10] and the second one, even more general, studied by Pradella et al. [11].

Since the beginning, it was evident that the two-dimensional topology of pictures changes a lot those properties of automata and grammars known from the one-dimensional case. For example, Blum and Hewitt proved that the non-deterministic four-way finite automaton (4NFA) is more powerful than the deterministic four-way finite automaton (4DFA). Equipping 4DFA by a pebble again results in a more powerful device. Advantages of the four-way alternating automaton (4AFA) were described by Kari and Moore [12]. The mentioned 2D grammars generate different classes of picture languages. Many differences have also been revealed for closure properties and decidability problems.

E-mail address: prusapa1@cmp.felk.cvut.cz.

http://dx.doi.org/10.1016/j.tcs.2015.05.033
0304-3975/

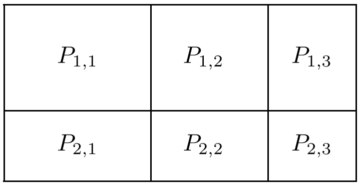

Fig. 1. The product of $\bigoplus[P_{ij}]_{2\times 3}$.

Despite these extensive studies, so far, no comparison has been done with respect to the descriptional complexity of the considered models. This paper aims to fill this gap. We show that there are many non-recursive trade-offs between the descriptional systems for picture languages.

Examples of such trade-offs are well known in the case of descriptional systems for string languages. The first non-recursive trade-off was presented by Meyer and Fisher [13]. They showed that the gain in economy of description can be arbitrary when the size of finite automata and general context-free grammars generating regular languages is compared. Since their work, other non-recursive trade-offs were reported, such as in [14–18]. Besides these particular results, the important properties of systems leading to non-recursive trade-offs were identified and generic proof schemes were established [19,20].

Our results are based on the ability of two-dimensional systems to simulate Turing machines. Known principles of a Turing machine simulation by a 4DFA are exploited to prove non-recursive trade-offs between 2D automata. In addition, a new technique of a Turing machine simulation by the 2D grammar from [10] is presented. It is utilized to prove non-recursive trade-offs between automata and grammars and also between the grammars themselves. Applicability of this result goes even beyond the scope of descriptional complexity, as it also answers some decidability questions.

The paper is structured as follows. In Section 2 we give the basic notions and notations on picture languages. In Section 3 we show non-recursive trade-off between 4DFA and 4NFA. It is also explained how this result extends to other automata. Descriptional complexity of 2D grammars is studied in Section 4. A detailed construction of a 2D grammar simulating a Turing machine is included here. The paper closes with a summary and some open problems in Section 5.

2. Preliminaries

Here we use the common notation and terms on pictures and picture languages (see, e.g., [21]). If Σ is a finite alphabet, then $\Sigma^{*,*}$ is used to denote the set of all rectangular pictures over Σ, that is, if $P \in \Sigma^{*,*}$, then P is a two-dimensional array (matrix) of symbols from Σ. If P is of size $m \times n$, this is denoted by $P \in \Sigma^{m,n}$. We also write $\mathrm{rows}(P) = m$ and $\mathrm{cols}(P) = n$. If P is a square picture $n \times n$, we shortly say P is of size n. $\Sigma^{+,+} = \{P \in \Sigma^{*,*} \mid \mathrm{rows}(P) > 0 \wedge \mathrm{cols}(P) > 0\}$ is the set of non-empty pictures. The empty picture Λ is defined as the only picture of size 0×0.

We use $[a_{ij}]_{m\times n}$ as a notation for a general matrix with m rows and n columns where the element in the i-th row and j-th column is denoted as a_{ij}.

Two (partial) binary operations are introduced to concatenate pictures. Let $A = [a_{ij}]_{k\times\ell} \in \Sigma^{k,\ell}$ and $B = [b_{ij}]_{m\times n} \in \Sigma^{m,n}$. The *column concatenation* $A \mathbin{\Phi} B$ is defined iff $k = m$, and the *row concatenation* $A \ominus B$ is defined iff $\ell = n$. The products are specified by the following schemes:

$$A \mathbin{\Phi} B = \begin{bmatrix} a_{11} & \dots & a_{1\ell} & b_{11} & \dots & b_{1n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{k1} & \dots & a_{k\ell} & b_{m1} & \dots & b_{mn} \end{bmatrix} \text{ and } A \ominus B = \begin{bmatrix} a_{11} & \dots & a_{1\ell} \\ \vdots & \ddots & \vdots \\ a_{k1} & \dots & a_{k\ell} \\ b_{11} & \dots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{m1} & \dots & b_{mn} \end{bmatrix}.$$

We generalize $\ominus$ and Φ to a *grid concatenation* which is applied to a matrix of pictures $[P_{ij}]_{m\times n}$ where each $P_{ij} \in \Sigma^{*,*}$. The operation $\bigoplus[P_{ij}]_{m\times n}$ is defined iff

$$\mathrm{rows}(P_{i1}) = \mathrm{rows}(P_{i2}) = \ldots = \mathrm{rows}(P_{in}) \quad \forall i = 1,\ldots,m\,,$$

$$\mathrm{cols}(P_{1j}) = \mathrm{cols}(P_{2j}) = \ldots = \mathrm{cols}(P_{mj}) \qquad \forall j = 1,\ldots,n\,.$$

Then, $\bigoplus[P_{ij}]_{m\times n} = P_1 \ominus P_2 \ominus \ldots \ominus P_m$, where $P_k = P_{k1} \mathbin{\Phi} P_{k2} \mathbin{\Phi} \ldots \mathbin{\Phi} P_{kn}$ for $k = 1,\ldots,m$. An example is given in Fig. 1.

In order to enable all considered finite automata to detect the border of an input picture P, they always work over the *boundary picture* $\widehat{P}$ over $\Sigma \cup \{\#\}$ of size $(\mathrm{rows}(P)+2)\times(\mathrm{cols}(P)+2)$, defined by the scheme in Fig. 2. We assume that the *background symbol* # is not contained in any considered input alphabet Σ.