NONDETERMINISM VERSUS DETERMINISM OF FINITE AUTOMATA OVER DIRECTED ACYCLIC GRAPHS *

Andreas Potthoff Sebastian Seibert Wolfgang Thomas

Abstract

Three types of finite-state graph automata are compared over directed acyclic graphs (where vertices and edges are labelled). The automata are distinguished by the way how states are attached to an input graph ("vertex-marking", "edge-marking", and "1-sphere-marking"). We note the equivalence of these models, relate them to logical definability notions, and show that deterministic versions are strictly weaker (thus correcting an error of [9]).

1 Introduction

The question of an adequate notion of recognizability of graph properties has recently attracted much attention, and many competing approaches have been developed. The starting point in this research is the notion of (nondeterministic or deterministic) finite automaton over words. In a first step towards more general inputs than words, finite tree automata were introduced by Doner [5] and Thatcher and Wright [13]. It was shown that many characterizations of recognizable word languages, namely in terms of regular expressions, recognizability in finite algebras, and definability in monadic second-order logic, are all naturally preserved when passing from words to trees. An important point in this theory is the reduction of nondeterministic tree automata (in the "bottom-up" or "frontier to root" version) to deterministic ones. Some time later, a model of finite automaton over planar directed acyclic graphs was introduced by Kamimura and Slutzki [9]. Whereas tree automata attach states to the vertices of a given input, the automata of [9] generate state labels on the edges of a graph. Continuing this research, Bossut, Dauchet, and Warin [2, 3] characterized nondeterministic automata on planar directed acyclic graphs by a calculus of regular expressions. More recently, Thomas [11, 12] introduced a third and more expressive notion of finite-state acceptor (over arbitrary finite graphs of bounded degree) whose recognition power matches existential monadic second-order logic. In this paper we

*The present work was supported by EBRA Working Group 6317 "Algebraic and Syntactic Methods in Computer Science (ASMICS 2)".

Received by the editors November 1993, revised February 1994.
Communicated by M. Boffa.
AMS Classification numbers : 68Q10, 68Q68, 68R10, 03D05.
Key words : labelled finite graphs, recognizability of graph languages, automata on acyclic graphs, non deterministic graph automata, deterministic graph automata.

first present a framework (given by a suitable class of graphs) in which the first two approaches and a natural weakening of the third one turn out to be equivalent. Secondly, the effect of restricting to deterministic acceptors will be analyzed.

Let us mention further approaches to graph recognition which are not pursued here. An important class of acyclic graphs arises in the theory of partially commutative monoids and Mazurkiewicz traces; in fact, a trace can be identified with a directed acyclic graph of a special kind. Since traces are introduced as a model of concurrent behaviour, also concurrent modes of recognition are of special interest (in the form of "asynchronous automata", see e.g. [14] and [1]). A more general type of asynchronous recognition (in a similar spirit as [11]) was introduced by Litovsky, Métivier, and Zielonka in [10]. Finally, over the domain of arbitrary finite graphs, a powerful algebraic theory of recognizability (in terms of locally finite graph algebras) was developed by Courcelle [4]; it encompasses more properties than those expressible even in full monadic second-order logic.

For the comparison of different kinds of graph automata, we use a common class of labelled graphs as input domain. It will consist of connected directed acyclic graphs with vertex labels from a given finite alphabet Σ and an additional labelling of the edges which in particular induces an ordering on the set of incoming, resp. outgoing edges for each vertex. More precisely, we restrict to graphs where on the incoming, resp. the outgoing edges of a vertex each edge label appears at most once. Thus we deal with "ordered" acyclic graphs; however, we do not insist on planarity (as done in some of the above mentioned papers).

In Section 1, we introduce the three types of graph automata and show their equivalence in expressive power (in the nondeterministic version). Let us briefly describe them informally.

The first automaton model, called *vertex-marking* automaton, is the direct generalization of tree automata to the domain of acyclic ordered graphs : The run of such an automaton attaches a state to each vertex of the underlying graph, and a transition allows to proceed from a tuple of states, associated with the sources of the incoming edges of a vertex v, and the alphabet label (from Σ) of v to the state to be associated with v.

In the second type of graph automaton, originating in [9] and [2], called *edge-marking* below, a run attaches a state to each edge, and a transition specifies how to proceed from a tuple of states associated with the incoming edges of a vertex v and the alphabet label of v to the tuple of states for the outgoing edges.

Finally, we consider graph automata that are derived as a special case of the acceptors in the sense of [11, 12] and called *1-sphere-marking* here. The idea is to use transitions which check "local neighbourhoods" of (or "spheres around") vertices v, in the general case consisting of vertices within a given distance n to v. Here we deal with "1-spheres" which just consist of a vertex v (the "center") together with the vertices incident to the incoming and outgoing edges of v. A run of the automaton associates a state to each vertex, such that each 1-sphere of the graph expanded by this state assignment matches a transition. Apart from the fact that all three types of graph automata are equivalent in recognition power we shall see that they are strictly weaker than existential monadic second-order logic (not even all first-order properties are covered). Thus the more general acceptors of [11, 12], which capture exactly existential monadic second-order logic, cannot be reduced to

the present version without loss of expressive power.

In Section 3 we discuss the issue of determinism. We start from the intuition that over acyclic graphs, a run of a deterministic automaton should be determined uniquely when traversing the graph in accordance with the partial order induced by the edges. We shall restrict to the vertex-marking and to the edge-marking automata, since (as we shall explain) the uniqueness of a run is not necessarily determined locally when 1-sphere-marking automata are considered. A drastic deficiency in the expressive power of deterministic automata is well-known from the theory of tree automata (when used in root-to-frontier tree traversal)[7]. Over acyclic graphs, this effect can to some extent be avoided when only graphs with a "co-root" are admitted (i.e. with a "final" vertex which is reachable from each vertex by a path and where the results of a deterministic computation can be collected). But also within this domain of acyclic graphs with co-root we shall present example sets that are recognized nondeterministically but not in deterministic mode (by vertex-marking and by edge-marking automata). These examples reveal an error in [9] where a subset construction for edge-marking automata is presented, claimed to establish a reduction from nondeterministic to deterministic edge-marking automata. The class of underlying graphs is somewhat specialized in [9] (referring to graphs that originate in derivations of phrase structure grammars), but our counterexamples apply to these special graphs as well.

The final section gives further comments concerning "globally deterministic", i.e. *nonambiguous* graph automata.

2 Three types of nondeterministic graph automata

2.1 Labelled graphs

A *vertex labelled graph* over the label alphabet Σ is of the form $G = (V, E, \mu)$, where V is the set of vertices, $E \subseteq (V \times V \setminus \mathrm{id}_V)$ is the set of edges, and $\mu : V \to \Sigma$ is the labelling function. For a vertex v, an edge (u, v) (resp. (v, u)) is called incoming (resp. outgoing) edge. The number of incoming, resp. outgoing edges of a vertex is its in-degree, resp. out-degree. A vertex with in-degree i and out-degree j is said to be of *rank* (i, j).

In order to control the flow of information in automata, we shall require a "coloring" $\gamma : E \to C$ of the edges by colors from a set $C = \{c_1, \ldots, c_m\}$, and restrict to those graphs where for each vertex there are no two occurrences of the same color on the incoming edges, and where the same holds for the outgoing edges. This allows to order the incoming, resp. outgoing edges by the sequence of indices of the occurring colors, and thus to speak of the "tuple of incoming edges", resp. "tuple of outgoing edges". We shall make use of the colors when associating extra labels (usually states) with edges, resp. their sources or targets. To describe such a labelling, say in the set Q, we specify for each color c_i a corresponding label from Q if an edge with this color is present, and else a dummy label $*$. Thus, over a color set $C = \{c_1, \ldots, c_m\}$, the labels on a set of (either incoming or outgoing) edges are represented by a tuple $(q_1, \ldots, q_m)$ where $q_i \in Q \cup \{*\}$, with $q_i \in Q$ iff an edge colored c_i is present, else

$q_i = *$. If $q_{i_1}, \ldots, q_{i_r}$ are the labels for the existing edges with colors $c_{i_1}, \ldots, c_{i_r}$, respectively, we call the extension to an m-tuple $(q_1, \ldots, q_m) \in (Q \cup \{*\})^m$ by insertion of $*$ for $j \in \{1, \ldots, m\} \setminus \{i_1, \ldots, i_r\}$ the m-tuple *associated with* $(q_{i_1}, \ldots, q_{i_r})$.

In the definition of "run" we shall refer to the colors of edges incident to a given vertex. Call a vertex v of *type* (σ, τ) if σ (resp. τ) is the set of colors of incoming (resp. outgoing) edges incident to v. We say that a set Q of states is *typed* if Q is a (not necessarily disjoint) union of sets $Q_{(\sigma,\tau)}$ where $Q_{(\sigma,\tau)}$ contains those states that are admissible on vertices of type (σ, τ).

The colors of edges serve here as an alternative to the approach taken in [9] and [2], where incoming and outgoing edges are ordered according to ranks of vertex symbols. Connections from vertex to vertex are then identifiable by these orderings together with the restiction that only planar graphs are considered. In the present paper we employ edge colors to establish a unique identification of an outgoing edge of one vertex with an incoming edge of another vertex by only local information. This will allow to include for example nonplanar graphs. Moreover, graphs with edge colors represent a natural generalization of trees with ordered successors (and bounded finite branching), as considered in the theory of tree automata.

Let us also note that for a color set $C = \{c_1, \ldots, c_m\}$ only graphs of both in-degree and out-degree bounded by $2m$ are admitted. By a Σ-C-*graph* we mean a connected acyclic vertex-labelled graph over the label alphabet Σ and color set C, respecting the mentioned condition on at most one occurrence of each color on a set of incoming, resp. outgoing edges. Recall that a graph $G = (V, E)$ is called *acyclic* if the reflexive-transitive closure of E is a partial order on V; and it is *connected* if for each pair (u, v) of vertices there is a sequence $v_0, v_1, \ldots, v_n$ such that $v_0 = u$, $v_n = v$, and $(v_i, v_{i+1}) \in E$ or $(v_{i+1}, v_i) \in E$ for $0 \leq i < n$.

2.2 Graph automata

For the following definitions we fix the color set $C = \{c_1, \ldots, c_m\}$.

Definition 2.1 *A* vertex-marking graph automaton *over Σ-C-graphs is a structure $\mathcal{A} = (Q, \Sigma, \Delta)$, where $Q = (Q_{(\sigma,\tau)})_{(\sigma,\tau)}$ is a typed finite set of states and Δ is a finite set of transitions. Here a transition is of the form*

$$[(q_1, \ldots, q_m, a) \to q]$$

where $q_1, \ldots, q_m \in Q \cup \{*\}$, $q \in Q$, and $a \in \Sigma$. Moreover, if $\sigma = \{c_i \mid q_i \neq *\}$ then $q \in Q_{(\sigma,\tau)}$ for some τ.

A *run* of $\mathcal{A}$ on a Σ-C-graph $G = (V, E, \mu)$ is a map $\rho : V \to Q$ which is compatible with the types (in the sense that for a vertex v of type (σ, τ) we have $\rho(v) \in Q_{(\sigma,\tau)}$) and is compatible with the transitions from Δ, as follows : If vertex $v \in V$ with $\mu(v) = a$ has the incoming edges $(v_{i_1}, v), \ldots, (v_{i_r}, v)$ with colors $c_{i_1}, \ldots, c_{i_r}$, respectively, and $(q_1, \ldots, q_m)$ is the m-tuple (over $(Q \cup \{*\})^m$) associated with $(\rho(v_{i_1}), \ldots, \rho(v_{i_r}))$, then

$$[(q_1, \ldots, q_m, a) \to \rho(v)] \in \Delta.$$

We say that $\mathcal{A}$ *accepts* G if there is a run of $\mathcal{A}$ on G. (Note that "initial states" are implicitly given as those in the sets $Q_{(\emptyset,\tau)}$, and "final states" as those in the sets $Q_{(\sigma,\emptyset)}$.) Let us call a set L of Σ-C- graphs *recognizable by vertex-markings* if for some vertex-marking graph automaton, the set of accepted Σ-C-graphs coincides with L.

Definition 2.2 *An* edge-marking *graph automaton over Σ-C-graphs is of the form $\mathcal{A} = (Q, \Sigma, \Delta)$, where Q is a finite set of states, Σ is as above, and Δ consists of transitions of the form*

$$[(p_1, \ldots, p_m, a) \to (q_1, \ldots, q_m)]$$

where $p_1, \ldots, p_m, q_1, \ldots, q_m \in Q \cup \{*\}$ and $a \in \Sigma$.

A *run* of $\mathcal{A}$ on a Σ-C-graph $G = (V, E, \mu)$ is now a map $\rho : E \to Q$ such that for each vertex $v \in V$ say with label $a \in \Sigma$, incoming edges $(v_{i_1}, v), \ldots, (v_{i_r}, v)$ colored by $c_{i_1}, \ldots, c_{i_r}$ and outgoing edges $(v, w_{j_1}), \ldots, (v, w_{j_s})$ colored by $d_{j_1}, \ldots, d_{j_s}$, respectively, we have

$$[(p_1, \ldots, p_m, a) \to (q_1, \ldots, q_m)] \in \Delta$$

for $(p_1, \ldots, p_m)$ and $(q_1, \ldots, q_m)$ associated with $\rho((v_{i_1}, v)), \ldots, \rho((v_{i_r}, v))$ and with $\rho((v, w_{j_1})), \ldots, \rho((v, w_{j_s}))$. "Initial" and "final" transitions are respectively of the form $[(*, \ldots, *, a) \to (q_1, \ldots, q_m)]$ and $[(p_1, \ldots, p_m, a) \to (*, \ldots, *)]$. The automaton $\mathcal{A}$ *accepts* G if there is a run of $\mathcal{A}$ on G; and a set L of Σ-C-graphs is *recognizable by edge-markings* if some edge-marking graph automaton accepts (within the Σ-C-graphs) exactly the graphs in L.

Our third type of graph automaton originates in [11] and rests on the notion of "1-sphere". The 1-sphere around a vertex v of a graph $G = (V, E)$ is the subgraph of G whose vertex set is $\{v\} \cup \{u \in V \mid (u, v) \in E \text{ or } (v, u) \in E\}$ and whose edges are those incident with v.

Definition 2.3 *A* 1-sphere-marking graph automaton *over Σ-C-graphs is of the form $\mathcal{A} = (Q, \Sigma, \Delta)$, where Q is a finite set of states and Σ an alphabet, and Δ consists of transitions of the form*

$$[(p_1, \ldots, p_m, (a, q), q_1, \ldots, q_m)]$$

where $p_1, \ldots, p_m, q_1, \ldots, q_m \in Q \cup \{*\}$, $a \in \Sigma$.

A *run* of $\mathcal{A}$ on G is a map $\rho : V \to Q$ such that the following compatibility with Δ is satisfied : Assume vertex v has incoming edges colored $c_{i_1}, \ldots, c_{i_r}$ with sources $v_{i_1}, \ldots, v_{i_r}$ and outgoing edges colored $d_{j_1}, \ldots, d_{j_s}$ with targets $w_{i_1}, \ldots, w_{i_r}$, and that $(p_1, \ldots, p_m)$ and $(q_1, \ldots, q_m)$ are the m-tuples associated with $(\rho(v_{i_1}), \ldots, \rho(v_{i_r}))$ and $(\rho(w_{j_1}), \ldots, \rho(w_{j_s}))$, respectively. Then $[(p_1, \ldots, p_m, (\mu(v), \rho(v)), q_1, \ldots, q_m)]$ should belong to Δ. Again, we say that $\mathcal{A}$ *accepts* G if there is a run of $\mathcal{A}$ on G. A set L of Σ-C-graphs is *recognizable by 1-sphere-markings* if some 1-sphere-marking graph automaton accepts (within the Σ-C-graphs) exactly the graphs in L.

Proposition 2.4 *The following conditions are equivalent for a set L of Σ-C-graphs :*

1. *L is recognizable by vertex-markings.*

2. *L is recognizable by edge-markings.*

3. *L is recognizable by 1-sphere-markings.*

Proof. We verify the implication chain (a)⇒(c)⇒(b)⇒(a).
(a)⇒(c) : Suppose $\mathcal{A} = (Q, \Sigma, \Delta)$ is a vertex-marking automaton. The transitions of a corresponding 1-sphere-marking automaton can be obtained as extensions of transitions of Δ. Define the 1-sphere-marking automaton $\mathcal{A}' = (Q, \Sigma, \Delta')$ by taking Δ' as the set of all transitions $[(q_1, \ldots, q_m, (a, q), p_1, \ldots, p_m)]$ where $[(q_1, \ldots, q_m, a) \to q] \in \Delta$ and q is of an admissible type (i.e. for $\sigma = \{c_i \mid q_i \neq *\}$ and $\tau = \{c_i \mid p_i \neq *\}$ we have $q \in Q_{(\sigma,\tau)}$).
(c)⇒(b) : This implication is also easy, by the similarity of edge-marking transitions and 1-spheres. Both include all edges connected to a certain vertex. The effect of the different positions of states in the two models can be handled by assigning to each edge the pair of states at its source and target. From a 1-sphere-marking automaton $\mathcal{A} = (Q, \Sigma, \Delta)$ we thus obtain an equivalent edge-marking automaton $\mathcal{A}' = (Q^2, \Sigma, \Delta')$ by taking Δ' as the set of all transitions $[((q_1, q), \ldots, (q_m, q), a) \to ((q, p_1), \ldots, (q, p_m))]$ where $[(q_1, \ldots, q_m, (a, q), p_1, \ldots, p_m)] \in \Delta$.
(b)⇒(a) : This is the main direction where we make use of the particular edge colorings in Σ-C-graphs. In simulating a given edge-marking automaton by a vertex-marking automaton, the idea is to store at each vertex a tuple of states, each component corresponding to an *outgoing* edge identified by its color. Thus we can propagate the appropriate state along each edge when executing the next transition at a successor vertex. (Without uniqueness of colors on the edges it would not be possible, when processing a successor vertex, to distinguish the position of the edge among the outgoing edges of the predecessor vertex. Thus we would loose the possibility of propagating different information along different outgoing edges as in edge-marking automata.)
Formally, let $\mathcal{A} = (Q, \Sigma, \Delta)$ be an edge-marking automaton over Σ-C-graphs, where $C = \{c_1, \ldots, c_m\}$.
We define an equivalent vertex-marking automaton $\mathcal{A}' = (Q', \Sigma, \Delta')$ by

- $Q' := (Q \cup \{*\})^m$,

- Δ' contains, for any type (σ, τ), exactly the transitions $[(\overline{p_1}, \ldots, \overline{p_m}, a) \to \overline{q}]$ satisfying the following conditions : $\{c_i \mid \bar{p}_i \neq *\} = \sigma$, $\bar{q} = (q_1, \ldots, q_m)$ with $\{c_i \mid q_i \neq *\} = \tau$, and each $\overline{p_i} \neq *$ is of the form $(p_{i1}, \ldots, p_{im})$ where $p_{ii} \neq *$ (clearly the source of an edge must have a state to be propagated along that edge, identified by its color). Moreover, there must be a transition $[(r_1, \ldots, r_m, a) \to (q_1, \ldots, q_m)] \in \Delta$, where $r_i = \begin{cases} p_{ii}, \text{ if } q_i \neq *, \\ * \text{ otherwise.} \end{cases}$

■

In the sequel let us call a set of graphs *recognizable (by graph automata)* if L is recognizable by a graph automaton of one of the three types above.

2.3 Examples

We now discuss examples which give an impression of the expressive power (resp. weakness) of graph automata.

Example 2.5 *Let* COL_k *($k \geq 2$) be the set of $\{a\}$-C-graphs that are k-colorable (i.e., admitting an assignment of vertices by colors from a set of k colors such that adjacent vertices are assigned different colors).* COL_k *is recognizable.*

An appropriate vertex-marking automaton has k states, each state corresponding to one (vertex-) color. Only transitions $[(p_1, \ldots, p_m, a) \to q]$ are allowed where q is different from all p_i. Then, in a run, the automaton "guesses" a color for each vertex, and checks for correctness of this coloring by its transitions.

■

Admissible vertex colorings are defined by a *universal* local condition (on *all* pairs of adjacent vertices). The next example is concerned with an existential condition (of single vertices) for all paths.

Example 2.6 *Let $\Sigma = \{a, b\}$ and C be an arbitrary alphabet of colors. The set L_0 consists of the Σ-C-graphs that contain, on each path from a vertex of in-degree 0 ("source vertex") to a vertex of out-degree 0 ("target vertex") at least one vertex labelled b. L_0 is recognizable.*

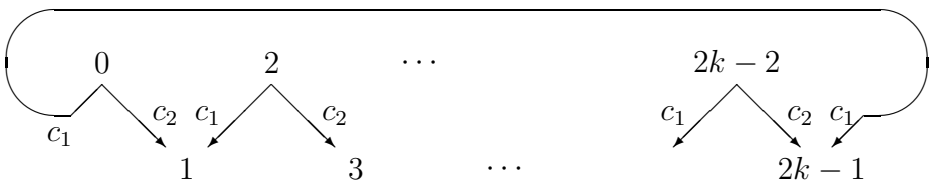
This claim is verified by means of a vertex-marking graph automaton with two states q_a, q_b, where on a given graph, state q_a indicates at vertex v that on some path from a source vertex to v only labels a are present, and q_b stands for the complementary condition. This is ensured if transitions $[(q_1, \ldots, q_m, a) \to q_b]$ are used exactly when no q_a occurs among the q_i, otherwise $[(q_1, \ldots, q_m, a) \to q_a]$ is included, and transitions $[(q_1, \ldots, q_m, b) \to q_b]$ are admitted for all choices of the q_i, and q_b is a " final state ", i.e. for any set σ of edge colors, we have $Q_{(\sigma,\emptyset)} = \{q_b\}$.

■

In the next example we see that graph automata are too weak to check for the mere existence of vertices (i.e., no more referring to existence within paths).

Example 2.7 *Let L_b be the set of Σ-C-graphs that contain at least one vertex labelled b. L_b is not recognizable.*

Suppose L_b is recognizable by the vertex-marking automaton $\mathcal{A}$ with say n states. Consider the graphs $G_k = (V_k, E_k, \mu)$ where $V_k = \{0, \ldots, 2k-1\}$, and E_k consists of the edges $\{(2i, 2i+1) \mid i < k\}$ colored c_2 and $\{(2i, 2i-1(\mathrm{mod}(2k))) \mid 0 < i < k\}$ colored c_1.

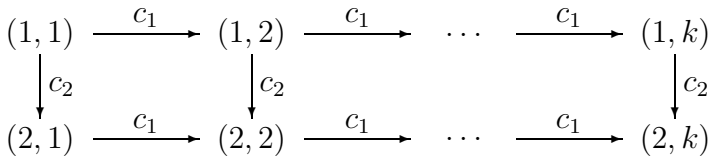


Furthermore, let $\mu(0) = b$, otherwise $\mu(v) = a$. For $k > n$, a run of $\mathcal{A}$ on G_k will have a state repetition on the vertex set $\{2, \ldots, 2k-2\}$, say at vertices i and $i+p$. Identifying these two vertices and restricting the vertex set to $\{i, \ldots, i+p\}$ we obtain a graph labelled only with a which is also accepted by $\mathcal{A}$, a contradiction.

■

To overcome the deficiency in expressive power of graph automata there are two possibilities. The first has been pursued by [2] in the domain of planar graphs, where a global ordering on the sets of source vertices (of in-degree 0), resp. target vertices (of out-degree 0) can be derived from the orderings of incoming and outcoming edges of single vertices. On the orderings of source or target vertices one can introduce extra finite-state tests (e.g. by usual finite automata) in order to control or check the global computation. For tree automata, where an ordering of the tree frontier is considered, this has been investigated further in [8]. Another approach, to be discussed in the present paper, is to restrict the class of graphs under consideration to those with *co-root*, i.e., with a vertex that is reachable from any vertex by a path. We thus study recognizability *relative to* the class of Σ-C-graphs with co-root. Under this relativization, the graph set L_b of the previous example clearly becomes recognizable. The advantage of a co-root is also well-known from the theory of tree automata in frontier-to-root mode. However, even relative to the class of graphs with co-root there are rather simple properties that cannot be recognized. We use an example from [11].

Example 2.8 *Let* LD *be the set of so-called "ladder-graphs" H_k over $\Sigma = \{a\}$ and $C = \{c_1, c_2\}$. The ladder graph H_k consists of the vertex set $\{1,2\} \times \{1, \ldots, k\}$ and has edges $((i,j),(i,j+1))$ colored c_1 for $i = 1, 2$, $1 \leq j < k$, and edges $((1,j),(2,j))$ colored c_2 for $1 \leq j < k$.*



LD *is not recognizable even relative to the class of Σ-C-graphs that have a co-root.* Before verifying this claim, we note that ladder graphs share the following commutation property : If we reach from vertex u the vertex v by passing first a c_1-colored edge and then a c_2-colored edge, the same vertex v will be reached by first passing a c_2 colored edge and then a c_1-colored one. Assuming now that LD is recognizable by a vertex marking graph automaton $\mathcal{A}$, we obtain a contradiction as follows. For sufficiently large k, a run of $\mathcal{A}$ will contain a repetition of a triple of states at certain vertices $(2, i-1), (1, i), (2, i)$ and $(2, j-1), (1, j), (2, j)$ where $i < j$. After replacement of the two vertical edges $((1,i),(2,i))$, $((1,j),(2,j))$ by $((1,i),(2,j))$, $((1,j),(2,i))$, respectively, a new acyclic (!) graph is obtained which violates the above commutation property and hence is not a ladder graph. By choice of i, j, however, the new graph is accepted by $\mathcal{A}$.

■

The commutation property is easily formalized by a (universal) first-order sentence (in the signature appropriate for $\{a\}$-$\{c_1, c_2\}$-graphs, cf. [11]). Since any graph automaton recognizing this property has in particular to accept all ladder graphs, we see from the above argument that certain first-order properties are not definable by graph automata. In view of this weakness, one way out is to restrict to a still more special class of graphs (than those with co-root) and to study recognizability relative to this class. A natural choice is the class of "grid graphs" or "pictures" as investigated in [6]. They are generalizations of ladder graphs to vertex sets of the form $\{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$. From the results of [6] it follows immediately that relative to this class, the graph automata of the present paper have the same expressive power as existential monadic second-order logic.

3 Determinism

Intuitively, an automaton over directed acyclic graphs should be called deterministic if placement of transitions is fixed in a unique way when traversing the graph along the edges, following the partial order given by the edge relation. As in [9], we shall make the assignment of a state to a vertex also dependent on its rank, which means that also the number of *outgoing* edges is taken into account. In our framework this corresponds to the dependence of the transitions on the type of the vertices.

For vertex-marking and edge-marking automata, an assignment of a state to a vertex, resp. edge, is given by a unique transition within a run, assuming an assignment of states to the preceding vertices and edges in the partial ordering of the graph. However, for 1-sphere-marking automata, the assignment of a state to a vertex v may be fixed by several transitions. For example, if there is an edge (u, v) and a path $(u, u_1), (u_1, u_2), \ldots, (u_n, v)$, then the state assignment to v involves the merge of two sequences of state assignments starting from u and thus is not defined in terms of local information. For this reason, we consider here deterministic graph automata only in the vertex-marking and in the edge-marking version.

Definition 3.1 *A vertex-marking graph automaton $\mathcal{A}$ as presented in Definition 2.1 is called* deterministic *if for all $q_1, \ldots, q_m$ with say $\sigma = \{c_i \mid q_i \neq *\}$, for all $a \in \Sigma$, and for all τ there is at most one transition $[(q_1, \ldots, q_m, a) \to q]$ with $q \in Q_{(\sigma,\tau)}$.*
An edge-marking graph automaton $\mathcal{A}$ as presented in Definition 2.2 is called deterministic *if for all $p_1, \ldots, p_m$ with say $\sigma = \{c_i \mid p_i \neq *\}$, for all $a \in \Sigma$, and for all τ there is at most one transition $[(p_1, \ldots, p_m, a) \to (q_1, \ldots, q_m)]$ such that $\tau = \{c_i \mid q_i \neq *\}$.*

Proposition 3.2 *A set of Σ-C-graphs is recognizable by a deterministic edge-marking automaton iff it is recognized by a deterministic vertex-marking graph automaton.*

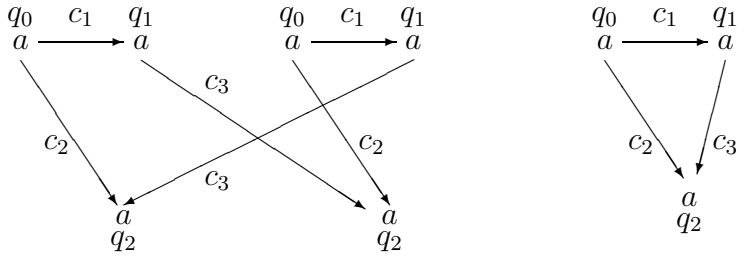
Proof. The proof is immediate by the construction of Proposition 2.4. Determinism is preserved in this construction, even when proceeding via 1-sphere-marking automata for the direction from vertex- to edge-marking automata.

■

If L is recognized by a deterministic graph automaton, either vertex-marking or edge marking, we say that L is *recognizable deterministically*.

Example 3.3 *Let* COL_k *($k \geq 2$) be the set of k-colorable graphs as in Example 2.5.* COL_k *is not recognizable deterministically.*

Suppose COL_2 is recognizable by a deterministic vertex-marking automaton. Consider a run on the first graph shown below, which is 2-colorable. By determinism the run will be a state assignment as indicated (where some of the q_i may coincide). Thus also the second displayed graph will be accepted by the indicated run. However, this graph is not 2-colorable.



Note that by the same argument even singleton sets may not be recognizable deterministically : Any deterministic automaton accepting the first graph also accepts the second.

■

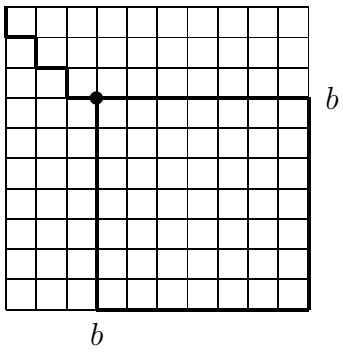
In the remainder of the paper, we restrict to grid graphs as studied in [6]. A *grid graph* is a Σ-$\{c_1, c_2\}$-graph with a vertex set of the form $\{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$ and with edges $((i,j),(i,j+1))$ colored c_1 for $1 \leq i \leq n_1, 1 \leq j < n_2$ and edges $((i,j),(i+1,j))$ colored c_2 for $1 \leq i < n_1, 1 \leq j \leq n_2$. A first example for deterministic recognition is the set of (say trivially labelled) square grids (where $n_1 = n_2$). A deterministic vertex-marking automaton recognizing this set may for instance assign a special state p along the diagonal vertices (i,i) and another special state q at the "subdiagonal" vertices $(i+1,i)$, starting from the top left corner vertex $(1,1)$. A test for the square grid property results if the only state with type $(\{c_1,c_2\},\emptyset)$ is p. Other deterministically recognizable sets of grid graphs arise by Turing machine computations, coded as two-dimensional arrays of symbols (each horizontal row coding one configuration). In fact, for any Turing machine $\mathcal{M}$ there is a vertex-marking graph automaton $\mathcal{A}_\mathcal{M}$ which recognizes (relative to the class of grid graphs) the set of halting computations of $\mathcal{M}$. We omit the details.

Let us present a set of grid graphs that is recognizable (relative to the class of grid graphs) but not deterministically recognizable. As a preparation, consider the set G_0 of square grid graphs over $\Sigma = \{a, b\}$ with exactly one occurrence of b on the lowest row of vertices and exactly one occurrence of b on the rightmost column of vertices. G_0 is easily seen to be recognizable by a deterministic vertex-marking automaton.

Example 3.4 *Let $H_=$ be the set of grid graphs in G_0 where the two symbols b occur on the same counterdiagonal, i.e., over $(n \times n)$-grids at positions (n,i) and (i,n) for*

some i. $H_=$ is recognizable relative to the class of grid graphs, but not recognizable deterministically.

Recognizablity of $H_=$ is easily verified using a nondeterministic automaton which associates special states to the positions that are marked in the following figure. The position marked by a big dot is guessed nondeterministically.



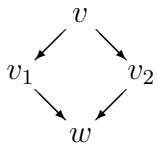
From the guessed position on the diagonal, special states are propagated to the right towards the rightmost column and down to the last row. There checks on the occurrence of letter b are carried out, the results being sent to the bottom right corner. Adding a second component to the states for the test whether the graph is in G_0 finishes the construction.

Let us verify that $H_=$ is not recognizable by a deterministic vertex-marking automaton. Over a $n \times n$-grid in G_0 the state at position $(n-1, n)$ (resp. at position $(n, n-1)$) depends only on the position of symbol b in the rightmost column (resp. in the lowest row). Thus a deterministic automaton with $m < n$ states reaches the same state at position $(n-1, n)$ for different locations of symbol b on the rightmost column. Therefore the automaton can not check correctly that the symbols b occur on the same counterdiagonal.

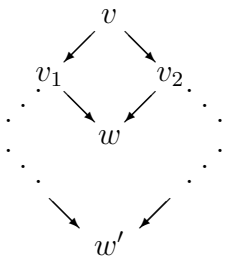
■

The example $H_=$ shows that a central claim of [9] is not correct, in particular Theorem 3.6 of that paper where a subset construction over acyclic planar graphs is presented in order to reduce nondeterministic graph automata to deterministic ones. The automata of [9] are edge-marking automata, where ranks are associated with vertex labels instead of states. This can be seen as a special case of the present edge-marking automata, since ranks of vertex labels can be controlled by an appropriate restriction in the transitions (using the edge-colors). The underlying graphs of [9] are planar directed acyclic graphs that arise as derivation graphs of phrase structure grammars ("d-dags"). Our example above does not fall precisely in this class of d-dags. However, one obtains a d-dag if each edge of a graph in our set G_0 of grid graphs is split into two consecutive edges (adding an extra intermediate vertex). By a copy of the argument given above, applied to these amplifications of grid graphs, we obtain a set that is recognizable but not deterministically recognizable (in our sense and hence also by the edge-marking automata in the sense of [9]).

The point where the argument of [9] fails is the following : In a "window" of the form



it must be clear at vertex w which pair of states q_1, q_2 was chosen when passing from v to v_1, v_2. In [9] this information is made available by a "generalized subset construction". However, it does not suffice to ensure this for each individual window, since the information about chosen pairs must be available at *all* points where paths from v_1, v_2 meet for the first time (say at w' in the figure below). This turns out to involve unbounded information within a deterministic state labelling.



Example 3.4 is concerned with "top-down" automata on grids (in the sense of traversals from the top left to the bottom right corner). For top-down automata over *trees* the well-known divergence between determinism and nondeterminism is overcome by inverting the traversal to bottom-up mode. In the present case of grids it is easy to adjust Example 3.4 to show that determinism is weaker than nondeterminism for both modes of traversal. For this, it suffices to supply the grids of $H_=$ with appropriate occurrences of b on the top and left border, again on the same counterdiagonal.

4 Concluding remarks

We have studied natural but rather weak models of finite-state recognizers over directed acyclic graphs. The expressive weakness is apparent especially from the fact that simple first-order properties are not recognizable. Useful applications become possible when the class of graphs is restricted, for example when considering only grid graphs. In all cases, however, we showed that determinism leads to a loss in recognition power.

In the present paper we leave open some questions on other restrictions of recognizability, for example concerning unambiguous recognizability. A recognizable graph set L is *unambiguous* if (say) a vertex-marking automaton recognizes it such that each graph in L admits at most one run of the automaton. The set $H_=$ of the last example is unambiguous, as is its complement relative to the class of grid graphs. So there are unambiguous (and at the same time co-unambiguous) graph

sets that are not recognizable deterministically. What is the relation of unambiguity to recognizability (say over grid graphs)? A more refined question considers sets of grid graphs which are recognizable and co-recognizable (i.e., such that their complement in the class of grid graphs is recognizable). Is there a recognizable and co-recognizable set which is not unambiguous?

References

[1] P. Bonizzoni, G. Mauri, G. Pigghizzini, N. Sabadini, Recognizing sets of labelled acyclic graphs, in : *Tree Automata and Languages* (M. Nivat, A. Podelski, eds.), Elsevier Science Publishers, Amsterdam (1992) 201-224.

[2] F. Bossut, M. Dauchet, B. Warin, Automata and rational expressions on planar graphs, in : Proc. MFCS 1988 (M.P.Chytil et al., eds.), Springer LNCS **324** (1988) 190-200.

[3] F. Bossut, B. Warin, Automata and pattern matching in planar directed acyclic graphs, in : LATIN 1992 (I.Simon, ed.), Springer LNCS **583** (1992) 76-86.

[4] B. Courcelle, Graph rewriting : An algebraic and logic approach, in : J.v. Leeuwen (ed.), *Handbook of Theoretical Computer Science*, Vol. B (1990) 193-242.

[5] J. Doner, Tree acceptors and some of their applications, J. Comput. System Sci. **4** (1970) 406-451.

[6] D. Giammarresi, A. Restivo, S. Seibert, W. Thomas, Monadic second-order logic over rectangular pictures and recognizability by tiling systems, Proc. STACS'94, Springer LNCS (to appear).

[7] F. Gecség, M. Steinby, *Tree Automata*, Akadémiai Kiadó, Budapest (1984).

[8] E. Jurvanen, A. Potthoff, W. Thomas, Tree languages recognizable by regular frontier-check, Rep. 9311, Inst. f. Informatik u. Prakt. Math., Univ. Kiel (1993).

[9] T. Kamimura, G. Slutzki, Parallel and two-way automata on directed ordered acyclic graphs, Inform. Contr. **49** (1981) 10-51.

[10] I. Litovsky, Y. Métivier, W. Zielonka, The power and the limitations of local computations on graphs and networks, Rep. 91-31, LaBRI, Université Bordeaux I (1991).

[11] W. Thomas, On logics, tilings, and automata, in : Proc. 18th Int. Coll. on Automata, Languages and Programming (ICALP), J. Leach Albert et al., eds., Springer LNCS **510** (1991) 441-453.

[12] W. Thomas, Finite-state recognizability of graph properties, in : Théorie des Automates et Applications, D. Krob, ed., Publ. de l'Univ. de Rouen No.**176** (1992) 147-159.

[13] J.W. Thatcher, J.B. Wright, Generalized finite automata with an application to a decision problem of second-order logic, Math. Systems Theory **2** (1968) 57-82.

[14] W. Zielonka, Notes on finite asynchronous automata, RAIRO Inform. Théor. **21** (1987) 99-135.

Andreas Potthoff, Sebastian Seibert, Wolfgang Thomas
Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität Kiel
40, Olshausenstr.
D-24098 Kiel
Deutschland
e-mail : {apo,ss,wt}@informatik.uni-kiel.d400.de