Volume 14 Number 4

ACTA CYBERNETICA

Szeged, 2000

ACTA CYBERNETICA

Information for authors. Acta Cybernetica publishes only original papers in the field of Computer Science. Contributions are accepted for review with the understanding that the same work has not been published elsewhere.

Manuscripts must be in English and should be sent in triplicate to any of the Editors. On the first page, the *title* of the paper, the *name(s)* and *affiliation(s)*, together with the *mailing* and *electronic address(es)* of the author(s) must appear. An *abstract* summarizing the results of the paper is also required. References should be listed in alphabetical order at the end of the paper in the form which can be seen in any article already published in the journal. Manuscripts are expected to be made with a great care. If typewritten, they should be typed double-spaced on one side of each sheet. Authors are encouraged to use any available dialect of TEX.

After acceptance, the authors will be asked to send the manuscript's source TEX file, if any, on a diskette to the Managing Editor. Having the TEX file of the paper can speed up the process of the publication considerably. Authors of accepted contributions may be asked to send the original drawings or computer outputs of figures appearing in the paper. In order to make a photographic reproduction possible, drawings of such figures should be on separate sheets, in India ink, and carefully lettered.

There are no page charges. Fifty reprints are supplied for each article published.

Publication information. Acta Cybernetica (ISSN 0324–721X) is published by the Department of Informatics of the University of Szeged, Szeged, Hungary. Each volume consists of four issues, two issues are published in a calendar year. For 2000 Numbers 3–4 of Volume 14 are scheduled. Subscription prices are available upon request from the publisher. Issues are sent normally by surface mail except to overseas countries where air delivery is ensured. Claims for missing issues are accepted within six months of our publication date. Please address all requests for subscription information to: Department of Informatics, University of Szeged, H-6701 Szeged, P.O.Box 652, Hungary. Tel.: (36)-(62)-420-184, Fax:(36)-(62)-420-292.

URL access. All these information and the contents of the last some issues are available in the Acta Cybernetica home page at http://www.inf.u-szeged.hu/local/acta.

H. Bunke
Universität Bern
Institut für Informatik und
angewandte Mathematik
Längass strasse 51., CH-3012 Bern
Switzerland

B. Courcelle
Universitè Bordeaux-1
LaBRI, 351 Cours de la Libèration
33405 TALENCE Cedex
France

J. Demetrovics
MTA SZTAKI
Budapest, P.O.Box 63
H-1502 Hungary

B. Dömölki
IQSOFT
Budapest, Teleki Blanka u. 15-17.
H-1142 Hungary

J. Engelfriet
Leiden University
Computer Science Department
P.O. Box 9512, 2300 RA Leiden
The Netherlands

Z. Ésik
University of Szeged
Department of Foundations of
Computer Science
Szeged, Aradi vértanúk tere 1.
H-6720 Hungary

L. Lovász
Eötvös Loránd University
Budapest Múzeum krt. 6-8.
H-1088 Hungary

G. Păun
Institute of Mathematics
Romanian Academy
P.O.Box 1-764, RO-70700
Bucuresti, Romania

A. Prékopa
Eötvös Loránd University
Budapest, Múzeum krt. 6-8.
H-1088 Hungary

A. Salomaa
University of Turku
Department of Mathematics
SF-20500 Turku 50, Finland

L. Varga
Eötvös Loránd University
Budapest, Múzeum krt. 6-8.
H-1088 Hungary

H. Vogler
Dresden University of Technology
Department of Computer Science
Foundations of Programming
D-01062 Dresden, Germany

G. Wöginger
Technische Universität Graz
Institut für Mathematik (501B)
Steyrergasse 30
A-8010 Graz, Österreich

Acta Cybernetica **14** (2000) 523-531.

Acts Over Completely 0-Simple Semigroups

Avdeyev A. Yu. * Kozhukhov I. B. †

The aim of this work is to describe, in the set-theoretical and group-theoretical terms, all the acts (automata) over completely 0-simple semigroups and also over completely simple and zero semigroups. As the consequence of this results we obtain a description of all the acts over rectangular groups, rectangular bands, right (or left) groups, and right (or left) zero semigroups. Moreover, we find all the subacts of some mentioned acts. Our results generalize the results of [3]. Theorem 1, Proposition 2 and Corollaries 9, 10, 11 of this work were published in [1]. We give them for the sake of completeness. Theorem 4 was announced by the second author in [7], Corollary 6 – by both authors in [2].

Recall that a *right act* (or *right operand*, or *S-set*) over a semigroup S is a set X with a mapping $X \times S \to X$ (the image of (x, s) we denote xs) such that the axiom $(xs)t = x(st)$ is held (for $x \in X$, $s, t \in S$) [6]. This notation coincides, in fact, with the notation "Moore's automaton" $V = (A, Q, \delta)$ where A is the input alphabet, Q is the set of the states, and δ is the transition function [8]. For the act X, we may assume that $Q = X$, A is the set of generators of S, and $\delta(x, s) = xs$. The S-set X is called *unitary* if S has a unity and $x \cdot 1 = x$ for all $x \in X$.

If the semigroup S has a simple structure, all the S-acts can be described. For example, an act X over the cyclic semigroup $S = \langle a \rangle$ is an unar (X, f) [9], i.e., the set X with the mapping $f : X \to X$; we have $x \cdot a^i = f^i(x)$. Ésik and Imreh [5] described the subdirectly irreducible commutative automata. Babcsányi and Nagy [3] obtained a description of the automata X over a right group S in case when the following conditions are satisfied:

$$XS = X, \tag{1}$$

$$\forall x, y \in X \forall s, t \in S \quad (xs = xt \Rightarrow ys = yt). \tag{2}$$

The condition (2) is called "state-independence". In this work we describe the right group acts (automata) in general case, i.e., without assuming (1), (2).

The notations and definitions of semigroup theory can be found in [4]. A completely 0-simple semigroup is signed by $\mathcal{M}^0(G, I, \Lambda, P)$, completely simple semigroup – by $\mathcal{M}(G, I, \Lambda, P)$. Here G is a group, I and Λ are sets, $P = \|p_{\lambda i}\|$ is a sandwich-matrix ($i \in I$, $\lambda \in \Lambda$, $p_{\lambda i} \in G \cup \{0\}$ or $p_{\lambda i} \in G$ resp.). The non-zero

*kv. 105, k. 200-5, 103305 Zelenograd, Russia.
†kv. 51, k. 1209, 103460 Zelenograd, Russia.

elements of $\mathcal{M}^0(G, I, \Lambda, P)$ have a form $(g)_{i\lambda}$ (where $g \in G$, $i \in I$, $\lambda \in \Lambda$) and their multiplication is defined by the rule

$$(g)_{i\lambda} \cdot (h)_{j\mu} = \begin{cases} (gp_{\lambda j}h)_{i\mu} & \text{if} \quad p_{\lambda j} \neq 0, \\ 0 & \text{if} \quad p_{\lambda j} = 0. \end{cases}$$

Let A be a set and θ an equivalence on A. Then A/θ is the set of θ-classes and $a\theta$ is the class of the element $a \in A$. An equivalence θ and a subset $B \subseteq A$ are called *compatible* if $|a\theta \cap B| = 1$ for every $a \in A$; in this case, the set B is called a *transversal* of θ. Let $\varphi : A \to A$ be a mapping. The *kernel* $\ker\varphi$ and the *image* $\mathrm{im}\varphi$ are defined as usual:

$$\ker\varphi = \{(a, b) | \varphi(a) = \varphi(b)\},$$

$$\mathrm{im}\varphi = \{\varphi(a) | a \in A\}.$$

If $\varphi^2 = \varphi$, $\ker\varphi$ and $\mathrm{im}\varphi$ are compatible, the opposite is false. If A is a right act over a semigroup S and $s \in S$, $\ker s$ and $\mathrm{im} s$ are the kernel and the image of the mapping $a \mapsto as$.

The element z of an S-act X is called a *zero* if $zs = z$ for all $s \in S$. Of course, an act X may have no zero. If the act X has an unique zero, we denote it by 0. Let (X_α) be a family of the S-acts X_α. Then $\bigsqcup\limits_\alpha X_\alpha$ is the *coproduct* (or *disjoint union*) of the acts X_α.

Let G be a group and H be a subgroup of G, not necessarily normal. Denote by G/H the set of the classes Hg where $g \in G$. The set G/H is an unitary right G-act with respect to the action $*$ where $Hg * g' = Hgg'$. Every unitary right act over the group G is obviously a disjoint union of *orbits* xG of the elements of X. It can be easily verified that every orbit is isomorphic (as a right G-act) to an act of form G/H for some subgroup H of G. Thus, we have the obvious assertion:

Lemma 1. *If G is a group and X is an unitary right G-act, then $X \cong \bigsqcup\limits_\alpha (G/H_\alpha)$ where (H_α) is a family of subgroups of G.*

Recall some definitions of the semigroup theory.

Zero semigroup is a semigroup S with 0 such that $ab = 0$ for all $a, b \in S$.

Left zero semigroup (L) is a semigroup satisfying the identity $xy = x$.

Right zero semigroup (R) is a semigroup with identity $xy = y$.

Rectangular band $(L \times R)$ is a semigroup determined by the identities $x^2 = x$, $xyz = xz$. It is known [4] that the rectangular band is isomorphic to a direct product of the left zero semigroup and the right zero one. Moreover, the rectangular band is isomorphic to the Rees matrix semigroup $\mathcal{M}(\{e\}, I, \Lambda, P)$ where $p_{\lambda i} = e$ for all $\lambda \in \Lambda$, $i \in I$.

Left group $(L \times G)$ is a direct product of a group and a left zero semigroup.

Right group $(R \times G)$ is a direct product of a group and a right zero semigroup.

We shall describe all the acts over zero semigroups. Let A be a set which is a disjoint union of some subsets A_α, i.e., $A = \cup\{A_\alpha | \alpha \in \Gamma\}$, B_α $(\alpha \in \Gamma)$ is some subset of A_α, and b_α $(\alpha \in \Gamma)$ is some element of B_α. Further, let S be a non-empty set and let φ_s, $s \in S$, be a family of mappings $\varphi_s : A \to A$ such that $\varphi_s(A_\alpha) \subseteq B_\alpha$ and $\varphi_s(B_\alpha) = \{b_\alpha\}$ for all $\alpha \in \Gamma$. Moreover, assume that there exists an element $\theta \in S$ such that $\varphi_\theta(A_\alpha) = \{b_\alpha\}$ for all $\alpha \in \Gamma$. If we put $st = \theta$ for all $s, t \in S$, then S turns a zero semigroup (with zero θ). Define the action of the semigroup S on the set A as follows: $as = \varphi_s(a)$ $(a \in A,\ s \in S)$.

Theorem 2. *The set A is a right act over the zero semigroup S. Conversely, every right act over a zero semigroup can be obtained by this way.*

Proof. At the first we check that A is a right S-act. Indeed, let $a \in A$ and $s, t \in S$. Then $a \in A_\alpha$ for some $\alpha \in \Gamma$. We have $(as)t = \varphi_t(\varphi_s(a)) \in \varphi_t(B_\alpha) = \{b_\alpha\}$, i.e., $(as)t = b_\alpha$. Moreover, $a(st) = a\theta = \varphi_\theta(a) = b_\alpha$. Thus, $(as)t = a(st)$.

Conversely, let A be an arbitrary right act over the zero semigroup S and θ is the zero of S. Introduce the equivalence σ on S putting $a\sigma b \Leftrightarrow a\theta = b\theta$. The equivalence determines the partition $A = \cup\{A_\alpha | \alpha \in \Gamma\}$. Check that $A_\alpha s \subseteq A_\alpha$ for all $\alpha \in \Gamma$, $s \in S$. Indeed, let $a \in A_\alpha$, $s \in S$. As $(as)\theta = a(s\theta) = a\theta$, then $(as, a) \in \sigma$. Therefore, $as \in A_\alpha$. Thus, $A_\alpha s \subseteq A_\alpha$. Put $B_\alpha = A_\alpha S$ for any $\alpha \in \Gamma$. If $a, b \in A_\alpha$ and $(a, b) \in \sigma$, we have $a\theta = b\theta$, therefore $|A_\alpha\theta| = 1$, and hence $A_\alpha\theta = \{b_\alpha\}$ for some b_α. Define, for any $s \in S$, the mapping $\varphi_s : A \to A$ putting $\varphi_s(a) = as$ for $a \in A$. Then $\varphi_s(A_\alpha) \subseteq B_\alpha$ and $\varphi_s(B_\alpha) = \{b_\alpha\}$. The theorem is proved.

The following proposition gives a description of all subacts of the act over a zero semigroup. The statements can be easily checked, and the proofs are omitted.

Proposition 3. *Let $A = \cup\{A_\alpha | \alpha \in \Gamma\}$ be a right act over the zero semigroup S, and $B_\alpha = A_\alpha S$ for $\alpha \in \Gamma$, and $\{b_\alpha\} = B_\alpha S$. If $\Delta \subseteq \Gamma$ is a non-empty subset and $A'_\delta \subseteq A_\delta$ (for $\delta \in \Delta$) such that $A'_\delta s \subseteq A'_\delta$ for all $s \in S$, then the act $\cup\{A'_\delta | \delta \in \Delta\}$ is a subact of A. Conversely, every subact of A can be obtained by this way.*

Now we shall consider the case of the completely 0-simple semigroup $S = \mathcal{M}^0(G, I, \Lambda, P)$. We may assume without loss of generality that $1 \in I \cap \Lambda$ and $p_{11} = e$ where e is the unity of the group G. The following theorem describes all the acts over such semigroups. We require here that $0 \cdot s = x \cdot 0 = 0$ for all $s \in S$, $x \in X$ where X is a right S-act, and 0 denotes the zero of S and the zero of X. The assumption of the existence of zero does not restrict the generality because of the fact that every act can be complemented by zero.

Theorem 4. *Let $S = \mathcal{M}^0(G, I, \Lambda, P)$ be a completely simple semigroup and X be a set with some element 0 (it is called conditionally as zero). Further, let (H_α) be a family of subgroups of the group G, $Q = \bigsqcup_\alpha (G/H_\alpha)$ is the coproduct of the right G-acts, and $Q^0 = Q \sqcup 0$. Finally, let us suppose that, for $i \in I$ and $\lambda \in \Lambda$, the*

mappings $\kappa_\lambda : Q^0 \to X$ *and* $\pi_i : X \to Q^0$ *are defined such that*

$$\kappa_\lambda(0) = 0; \quad \pi_i(0) = 0; \tag{3}$$

$$\pi_i(\kappa_\lambda(q)) = q * p_{\lambda i} \quad \textit{for all} \quad q \in Q^0. \tag{4}$$

Put, for $x \in X$ *and* $s = (g)_{i\lambda} \in S$,

$$x \cdot s = x \cdot (g)_{i\lambda} = \kappa_\lambda(\pi_i(x) * g) \quad \textit{and} \quad x \cdot 0 = 0. \tag{5}$$

Then X *is a right* S*-act with zero. Conversely, every right act with zero over a completely 0-simple semigroup can be obtained by this way.*

Proof. At the first, we shall check that the set X satisfying the written conditions is really a right S-act. Clearly, it is sufficient to prove that

$$(x \cdot (g)_{i\lambda}) \cdot (h)_{j\mu} = x \cdot ((g)_{i\lambda} \cdot (h)_{j\mu}). \tag{6}$$

We have

$$(x \cdot (g)_{i\lambda}) \cdot (h)_{j\mu} = \kappa_\lambda(\pi_i(x) * g) \cdot (h)_{j\mu} =$$

$$\kappa_\mu(\pi_j(\kappa_\lambda(\pi_i(x) * g)) * h) = \kappa_\mu(\pi_i(x) * g * p_{\lambda j} * h) =$$

$$= \begin{cases} 0, & \text{if } p_{\lambda j} = 0, \\ (gp_{\lambda j}h)_{i\mu} & \text{if } p_{\lambda j} \neq 0. \end{cases}$$

This implies (6).

Now, let X be a right S-act with zero. Put $Y = X \cdot (e)_{11}$. Define an action of the group G on the set Y as follows: $y * g = y \cdot (g)_{11}$ for $y \in Y$, $g \in G$. Because of condition $p_{11} = e$ we have $(y * g) * h = (y \cdot (g)_{11}) \cdot (h)_{11} = y \cdot ((g)_{11} \cdot (h)_{11}) = y \cdot (gh)_{11} = y * gh$. Moreover, $y * e = (x \cdot (e)_{11}) \cdot (e)_{11} = x \cdot (e)_{11} = y$. Therefore, Y is a unitary right G-act with zero. It follows from Lemma 1 that there exists a family of subgroups $H_\alpha \subseteq G$ and an isomorphism $\theta : Y \to Q^0 = \bigsqcup_\alpha (G/H_\alpha) \bigsqcup 0$ of right G-acts.

Construct, for every $\lambda \in \Lambda$, the mapping $\kappa_\lambda : Q^0 \to X$ putting $\tau_\lambda(x) = x \cdot (e)_{1\lambda}$ and $\kappa_\lambda(q) = \tau_\lambda(\theta^{-1}(q))$ where $x \in X$, $q \in Q^0$. Then construct, for $i \in I$, the mapping $\pi_i : X \to Q^0$ putting $\pi_i(x) = \theta(x \cdot (e)_{i1})$ where $x \in X$. If $i \in I$, $\lambda \in \Lambda$, $q \in Q^0$, and $p_{\lambda i} \neq 0$, we obtain $\pi_i(\kappa_\lambda(q)) = \theta(\kappa_\lambda(q) \cdot (e)_{i1}) = \theta(\tau_\lambda(\theta^{-1}(q)) \cdot (e)_{i1}) = \theta(\theta^{-1}(q) \cdot (e)_{1\lambda} \cdot (e)_{i1}) = \theta(\theta^{-1}(q) \cdot (p_{\lambda i})_{11}) = \theta(\theta^{-1}(q) * p_{\lambda i}) = \theta(\theta^{-1}(q * p_{\lambda i})) = q * p_{\lambda i}$. If $p_{\lambda i} = 0$, we obtain $\pi_i(\kappa_\lambda(q)) = \theta(\theta^{-1}(q) \cdot (e)_{1\lambda} \cdot (e)_{i1}) = \theta(\theta^{-1}(q) \cdot 0) = 0 = q * 0 = q * p_{\lambda i}$. Therefore, the equality (4) is satisfied in any case.

Finally, we verify the equality (5). We have $\kappa_\lambda(\pi_i(x) * g) = \kappa_\lambda(\theta(x \cdot (e)_{i1}) * g) = \kappa_\lambda(\theta((x \cdot (e)_{i1}) * g)) = \kappa_\lambda(\theta(x \cdot (e)_{i1} \cdot (g)_{11})) = \kappa_\lambda(\theta(x \cdot (g)_{i1})) = \tau_\lambda(\theta^{-1}(\theta(x \cdot (g)_{i1}))) = \tau_\lambda(x \cdot (g)_{i1}) = x \cdot (g)_{i1} \cdot (e)_{1\lambda} = x \cdot (g)_{i\lambda}$. The theorem is proved.

Now we consider the case of the completely simple semigroup $S = \mathcal{M}(G, I, \Lambda, P)$. As before, we assume that $1 \in I \cap \Lambda$ and $p_{11} = e$ where e is the unity of the group G. Moreover, as the matrix P has only non-zero elements,

then we may assume (without loss of generality) that some column and some row consists only of unities. Let $p_{\lambda 1} = p_{1i} = e$ for all $i \in I, \lambda \in \Lambda$. The description of the acts over completely simple semigroup is given by the following theorem.

Theorem 5. *Let X be a set, $S = \mathcal{M}(G, I, \Lambda, P)$ be a completely simple semigroup, (H_α) be a family of subgroups of G, and $Q = \bigsqcup_\alpha (G/H_\alpha)$ be the coproduct of G-acts. Suppose that, for every $i \in I$, an equivalence σ_i on X is given, for every $\lambda \in \Lambda$, a subset $X_\lambda \subseteq X$ is given, for $i \in I$, the mappings $\pi_i : X \to Q$, $\kappa_\lambda : Q \to X$ are given. Suppose that the following conditions hold (for $i \in I$, $\lambda \in \Lambda$, $x \in X$, $q \in Q$):*

$$\ker \pi_i = \sigma_i, \tag{7}$$

$$\mathrm{im} \kappa_\lambda = X_\lambda, \tag{8}$$

$$|X_\lambda \cap x\sigma_i| = 1, \tag{9}$$

$$(\pi_i \kappa_\lambda)(q) = q * p_{\lambda i}. \tag{10}$$

Put

$$x \cdot (g)_{i\lambda} = \kappa_\lambda(\pi_i(x) * g) \tag{11}$$

for $x \in X$, $(g)_{i\lambda} \in S$. Then X turns a right S-act. Conversely, every right act over the completely simple semigroup $S = \mathcal{M}(G, I, \Lambda, P)$ can be obtained by this way.

Proof. As is seen in the proof of THeorem 4, from the conditions (10) and (11), it can be shown that X is a right S-act.

Now we assume that X is an arbitrary right S-act. Put $e_i = (e)_{i1}$, $e_{i\lambda} = (p_{\lambda i}^{-1})_{i\lambda}$ for $i \in I$, $\lambda \in \Lambda$. Clearly, e_i and $e_{i\lambda}$ are idempotents. It is easy to check that $e_{i\lambda} e_i = e_i$, $e_i e_{i\lambda} = e_{i\lambda}$, $e_{1\lambda} e_{i\lambda} = e_{1\lambda}$ and $e_{i\lambda} e_{1\lambda}$. Put $X_\lambda = X e_{i\lambda}$. Then $X_\lambda = X e_{1\lambda} = X e_{1\lambda} e_{i\lambda} \subseteq X e_{i\lambda} = X_\lambda$. Then $X_\lambda = X e_{i\lambda}$ for any i.

For every $i \in I$, we put $\sigma_i = \{(x, y) \in X \times X | x e_i = y e_i\}$. Prove that

$$\forall x, y \in X \quad \forall \lambda \in \Lambda \quad (x e_i = y e_i \Leftrightarrow x e_{i\lambda} = y e_{i\lambda}). \tag{12}$$

Indeed, $x e_i = y e_i$ implies $x e_{i\lambda} = x e_i e_{i\lambda} = y e_i e_{i\lambda}$ and similarly $x e_{i\lambda} = y e_{i\lambda}$ implies $x e_i = y e_i$. Therefore the (12) holds.

We shall prove the property (9), i.e., every σ_i-class intersects with every X_λ in one element (in other words, X_λ is a set of representatives of σ_i). Let $x \in X$. Then from the above facts, we have that $x e_{i\lambda} \in X_\lambda$ and $(x e_{i\lambda}) e_i = x e_i$, so that $x e_{i\lambda} \in X_\lambda \cap x\sigma_i$. Then $X_\lambda \cap x\sigma_i \neq \emptyset$ (notice: if $s^2 = s$ and $x \in Xs$, then $xs = x$, since $x = us = us^2 = (us)s = xs$). Let $x, y \in X_\lambda$ with $(x, y) \in \sigma_i$. Then again from the above facts, we have $X = x e_{i\lambda} = y e_{i\lambda} = y$. Thus $X_\lambda \cap x\sigma_i = \{x e_{i\lambda}\}$ for every $x \in X$.

For $i \in I$ and $\lambda \in \Lambda$, let π_i and κ_λ be as in the proof of Theorem 4. Then we can similarly show that the conditions (10) and (11) hold. This completes the proof.

Corollary 6. *Let G be a group, X, L, R be sets, (H_α) be a family of subgroups of G, and $Q = \bigsqcup_\alpha (G/H_\alpha)$ be the coproduct of the right G-acts. Assume that the following objects are given:*

the equivalences σ_l on X for all $l \in L$,

the subsets $X_r \subseteq X$ for all $r \in R$,

the mappings $\pi_l : X \to Q$, $\kappa_r : Q \to X$ for all $l \in L$, $r \in R$

such that the following conditions hold (for $x \in X$, $l \in L$, $r \in R$):

$$\ker \pi_l = \sigma_l, \quad \operatorname{im} \kappa_r = X_r, \quad |X_r \cap x\sigma_l| = 1, \quad \pi_l \kappa_r = \mathrm{id}_Q.$$

Let $S = L \times G \times R$. Define the multiplication on S by the rule

$$(l, g, r) \cdot (l', g', r') = (l, gg', r')$$

and the action of S on X by the rule

$$x \cdot (l, g, r) = \kappa_r(\pi_l(x) * g).$$

Then S is a rectangular group and X is a right S-act. Conversely, every right act over a rectangular group can be obtained by this way.

Corollary 7. *Let G be a group, X and R be sets, (H_α) be a family of subgroups of G, and $Q = \bigsqcup_\alpha (G/H_\alpha)$ be the coproduct of the right G-acts. Assume that the following objects are given:*

the equivalence σ on X,

the subsets $X_r \subseteq X$ for all $r \in R$,

the mappings $\pi : X \to Q$, $\kappa_r : Q \to X$ for all $r \in R$

such that the following conditions hold (for $x \in X$, $r \in R$):

$$\ker \pi = \sigma, \quad \operatorname{im} \kappa_r = X_r, \quad |X_r \cap x\sigma| = 1, \quad \pi \kappa_r = \mathrm{id}_Q.$$

Let $S = G \times R$. Define the multiplication on S by the rule

$$(g, r) \cdot (g', r') = (gg', r')$$

and the action of S on X by the rule

$$x \cdot (g, r) = \kappa_r(\pi(x) * g).$$

Then S is a right group and X is an S-act. Conversely, every right act over a right group can be obtained by this way.

Remark. This corollary gives a description of all acts over the right groups, the "state-independence" and the condition (1) are not necessarily satisfied. Let us see what will be obtained in case when this conditions (1), (2) are fulfilled.

Let $S = G \times R$ and X be an S-act with the properties (1), (2). At first we notice that X_r is a unitary right G-act with respect to the operation $a * g = a \cdot (g, r)$ for $a \in X_r$, $g \in G$. Indeed, $u * g = a \cdot (g, r) = \kappa_r(\pi(a) * g) \in \mathrm{im}\kappa_r = X_r$, therefore $X_r * G \subseteq X_r$. Further, $a * e = \kappa_r(\pi(a) * e) = \kappa_r(\pi(a)) = a$. Finally, $a * (g_1 g_2) = a \cdot (g_1 g_2, r) = a \cdot ((g_1, r) \cdot (g_2, r)) = (a \cdot (g_1, r)) \cdot (g_2, r) = (a * g_1) * g_2$.

Now we notice that $X_r \cong Q$ as the right G-acts. Indeed, as $\pi\kappa_r = \mathrm{id}_Q$, then κ_r is an injection. It implies that κ_r is a bijection from Q onto $\mathrm{im}\kappa_r = X_r$. Moreover, $\kappa_r(q) * g = \kappa_r(q) \cdot (g, r) = \kappa_r(\pi(\kappa_r(q)) * g) = \kappa_r(q * g)$. Thus, κ_r is an isomorphism of X_r and Q.

The condition (1) implies that $X = \cup\{X_r | r \in R\}$. Check that X_r are disjoint. Let $X_r \cap X_{r'} \neq \emptyset$, and $a \in X_r \cap X_{r'}$. Then $a = \kappa_r(q) = \kappa_{r'}(q')$ for some $q, q' \in Q$. As $\pi(a) = \pi\kappa_r(q) = q$ and similarly $\pi(a) = q'$, then $q = q'$. Further, $a \cdot (e, r) = \kappa_r(\pi(a) * e) = \kappa_r \pi(a) = \kappa_r(q) = a$ and similarly $a \cdot (e, r') = a$. Because of the property (2) we have $x \cdot (e, r) = x \cdot (e, r')$ for all $x \in X$, i.e., $\kappa_r(\pi(x)) = \kappa_{r'}(\pi(x))$. Since π is surjective, we have $\kappa_r = \kappa_{r'}$, and hence $X_r = X_{r'}$. Thus, *X is a disjoint union of the pairwise isomorphic G-acts X_r*. This is the main result of [3].

Corollary 8. *Let G be a group, X and L be sets, (H_α) be a family of subgroups of G, and $Q = \bigsqcup_\alpha (G/H_\alpha)$ be the coproduct of the right G-acts. Assume that the following objects are given:*

the equivalences σ_l on X for all $l \in L$,

the subset $Y \subseteq X$,

the mappings $\pi_l : X \to Q$, $\kappa : Q \to X$ for all $l \in L$,

such that the following conditions hold (for $x \in X$, $l \in L$):

$$\ker \pi_l = \sigma_l, \quad \mathrm{im}\, \kappa = Y, \quad |Y \cap x\sigma_l| = 1, \quad \pi_l \kappa = \mathrm{id}_Q.$$

Let $S = L \times G$. Define the multiplication on S by the rule

$$(l, g) \cdot (l', g') = (l, gg')$$

and the action of S on X by the rule

$$x \cdot (l, g) = \kappa(\pi_l(x) * g).$$

Then S is a left group and X is a right S-act. Conversely, every right act over a left group can be obtained by this way.

Corollary 9 [1]. *Let X, L, R be sets. Assume that the following objects are given:*

the equivalences σ_l on X for all $l \in L$,

the subsets $X_r \subseteq X$ for all $r \in R$.

Also assume that the following conditions hold, for any $r, r' \in R$, $l, l' \in L$, $x \in X$:

$$|X_r \cap x\sigma_l| = 1, \tag{13}$$

$$\forall a \in X_r \quad \forall b \in X_{r'} \quad (a, b) \in \sigma_l \Leftrightarrow (a, b) \in \sigma_{l'}. \tag{14}$$

Define the multiplication on the set $S = L \times R$ by the rule

$$(l,r) \cdot (l',r') = (l,r')$$

and the action of S on X by the rule

$$a \cdot (l,r) = b \quad \text{where} \quad a\sigma_l \cap X_r = \{b\}.$$

Then S is a rectangular band and X is a right S-act. Conversely, every right act over a rectangular band can be obtained by this way.

Proof. We give the proof another than the proof of [1]. Clearly, the formulated rule determines a rectangular band. We shall prove that X is a right S-act. Indeed, let $a \in X$, $a \cdot (l,r) = b$, and $b \cdot (l',r') = c$. Then $a\sigma_l \cap X_r = \{b\}$ and $b\sigma_{l'} \cap X_{r'} = \{c\}$. We see that $(b,c) \in \sigma_{l'}$, therefore, because of the (14), $(b,c) \in \sigma_l$. As $(a,b) \in \sigma_l$, then $(a,c) \in \sigma_l$. Since $c \in X_{r'}$, $a \cdot (l,r') = c$. Thus, $a \cdot ((l,r) \cdot (l',r')) = a \cdot (l,r') = c = b \cdot (l',r') = (a \cdot (l,r)) \cdot (l',r')$. We see that X is a right S-act.

Further, we need to prove that the sets X_r and the equivalences σ_l of Corollary 6 satisfy to (14). Indeed, let $(a,b) \in \sigma_l$ where $a \in X_r$, $b \in X_{r'}$. Then we have $a = \kappa_r(q)$, $b = \kappa_{r'}(q')$ for some $q, q' \in Q$. As $(a,b) \in \sigma_l$, then $\pi_l(a) = \pi_l(b)$. We have $\pi_l(a) = \pi_l(\kappa_r(q)) = q$ and similarly $\pi_{l'}(a) = q$, $\pi_l(b) = \pi_{l'}(b) = q'$. As $\pi_l(a) = \pi_l(b)$, then $q = q'$. It implies $(a,b) \in \sigma_{l'}$.

We want to show that the Corollary 6 coincides with the Corollary 9 in case when $G = \{1\}$. Indeed, we may take $Q = X_{r_0}$ where $r_0 \in R$ is a fixed element and put $\pi_l(x) = y$ when $x\sigma_l \cap X_{r_0} = \{y\}$. Also we put $\kappa_r(q) = x$ when $q\sigma_l \cap X_r = \{x\}$ (the correctness, i.e., independence on l follows from (13) and (14): as $q \in X_{r_0}$, then $(q,x) \in \sigma_l \Leftrightarrow (q,x) \in \sigma_{l'} \Leftrightarrow q\sigma_{l'} \cap X_r = \{x\}$). It remains to show that $\pi_l\kappa_r = \mathrm{id}_Q$. Let $q \in Q$, $\kappa_r(q) = x$, and $\pi_l(x) = q'$. Then $q\sigma_l \cap X_r = \{x\}$ and $x\sigma_l \cap X_r = \{q'\}$. We have $(q',x) \in \sigma_l$. It follows that $q, q' \in X_{r_0} \cap x\sigma_l$. The condition (13) implies $q = q'$.

Corollary 10. *Let X and S be sets, σ be an equivalence on X, and (X_s), $s \in S$ be a family of subsets of the set X such that $|X_s \cap a\sigma| = 1$ for all $s \in S$, $a \in X$. Define the multiplication on the set S by the rule $st = t$ for all $s,t \in S$, and define the action of S on X by the rule*

$$as = b \Leftrightarrow X_s \cap a\sigma = \{b\}.$$

Then S is a right zero semigroup, and X is a right S-act. Conversely, every right act over a right zero semigroup can be obtained by this way.

Corollary 11. *Let X and S be sets, Y be a non-empty subset of X, and (σ_s), $s \in S$ be a family of the equivalences on X such that $|Y \cap a\sigma_s| = 1$ for all $s \in S$, $a \in X$. Define the multiplication on the set S by the rule $st = s$ for all $s,t \in S$, and define the action of S on X by the rule*

$$as = b \Leftrightarrow Y \cap a\sigma_s = \{b\}.$$

Then S is a left zero semigroup, and X is a right S-act. Conversely, every right act over a right zero semigroup can be obtained by this way.

The authors are thankful to the referees for their valuable suggestions.

References

[1] Avdeyev A.Yu., Kozukhhov I.B. Acts over semigroups of simple structure. The 6th Conf. of Moscow State Social Univ. "Mathematical Methods and Applications". Abstr. of reports, p. 103-107. Moscow, 1999 (in Russian).

[2] Avdeyev A.Yu., Kozhukhov I.B. Acts over the rectangular groups. 12th Intern. Conf. "Problems of Theoretical Cybernetics" (Nizhny Novgorod, Russia, 1999). Abstr. of reports, p. 5. Moscow, 1999 (in Russian).

[3] Babcsányi I., Nagy A. Right group-type automata. Acta Cybernetica, 1995, 12, 131-136.

[4] Clifford A.H., Preston G.B. The Algebraic Theory of Semigroups. Amer. Math. Soc., Providence, I(1961), II(1967).

[5] Ésik Z., Imreh B. Subdirectly irreducible commutative automata. Acta cybernetica, 1981, 5, 251-260.

[6] Kilp M., Knauer U., Mikhalev A.V. Monoids, acts and categories. W. de Gruyter, Berlin – New York, 2000.

[7] Kozhukhov I.B. Acts over completely simple semigroups. Intern. Seminar "Universal Algebra and Its Applications" due to memory of Prof. Skornjakov L.A. (Volgograd, Russia, 1999). Abstr. of reports, p. 35-36. Volgograd, 1999 (in Russian).

[8] Kudryavtsev V.B., Podkolzin A.S. Introduction into Theory of Abstract Automata. Moscow, Moscow State Univ., 1985 (in Russian).

[9] Skornjakov L.A. Unars. Colloq. Math. Soc. János Bolyai. 29. Universal Algebra, Esztergom (Hungary), 1977, p. 735-742.

Received November, 1999

Acta Cybernetica 14 (2000) 533-540.

The Logic of Knights, Knaves, Normals and Mutes

L. Aszalós *

Abstract

R. M. Smullyan wrote in his book about islands, knights and knaves. The knights always tell the truth and the knaves are always lying. Instead of *say* we shall examine the *can say* modal operator. We show the soundness and the completeness of this logic.

1 Introduction

At first we introduce the characters of the puzzles. Then we describe a logical language suitable to formulate puzzles. Later we prove soundness and completeness of this logic and eventually we show some interesting properties of this logic.

In Smullyan's famous book [2] the knights always tell the truth. Consequently they cannot say false statements. Smullyan does not mention any taboo in his puzzles, so we can assume that the knights can say any true statement. For the knaves the opposite holds, so they can say any false statement and can not say any true statement. We can arrange our information in columns:

	can say false statements	can say no false statements
can say true statements		knights
can say no true statements	knaves	

Later Smullyan introduced a third type of islanders: the normals, who sometimes tell the truth and sometimes lie. If we put this type into the table, one entry will remain empty. To fill this gap we need a new type of islanders, who can not say anything; hence we call them *mutes*. So the complete table is the following:

	can say false statements	can say no false statements
can say true statements	normals	knights
can say no true statements	knaves	mutes

*University of Debrecen, 4010 Debrecen, PO Box 12, Hungary, e-mail: aszalos@math.klte.hu

2 Syntax

In the following we shall use the well-known definition of the syntax of propositional logic:

Definition 1 *Let be $\mathcal{S}$ a finite set of propositional letters. The set of propositional formulae is the smallest set $\mathcal{F}$ such that*

1. $\mathcal{S} \subset \mathcal{F}$.
2. *If* $A \in \mathcal{F}$ *then* $\neg A \in \mathcal{F}$.
3. *If* A, $B \in \mathcal{F}$ *then* $(A \wedge B)$, $(A \vee B)$ *and* $(A \supset B) \in \mathcal{F}$.

In the definition above the connectives are the usual: $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction) and $\supset$ (implication). To formulate the puzzles we need to express that the person x can say true statements, the person x can say false statements and the person x *can say* the statement A. For this we introduce $\mathtt{T}_x$, $\mathtt{F}_x$ and $\mathtt{S}_x A$, respectively. Definition 1. is extended to

Definition 2 *Let be $\mathcal{P}$ a finite set. The set of formulae is the smallest set $\mathcal{F}$ that satisfies 1-3. and*

4. *If* $x \in \mathcal{P}$ *then* $\mathtt{T}_x \in \mathcal{F}$ *and* $\mathtt{F}_x \in \mathcal{F}$.
5. *If* $x \in \mathcal{P}$ *and* $A \in \mathcal{F}$ *then* $\mathtt{S}_x A \in \mathcal{F}$.

In this definition $\mathcal{P}$ is the set of persons, and elements of $\mathcal{P}$ will be denoted by a, b, ... This definition allows the embedding of $\mathtt{S}_x$ in the formulae, so for example $\mathtt{S}_a \neg \mathtt{S}_b \mathtt{T}_a$ is a legal formula, which means that a *can say* that b *cannot say* that a *can say* true statements.

3 Semantics

In the propositional logic the prime components are the propositional letters. In our logic the truth value of a formula can depend on the type of persons, so the formulae describing the type of persons are prime components too. Hence the definition of the valuation will be more complicated than usual:

Definition 3 *Let $\vartheta_S \subset \mathcal{S}$, $\vartheta_T \subset \mathcal{P}$ and $\vartheta_F \subset \mathcal{P}$. The valuation $\vartheta = \langle \vartheta_S, \vartheta_T, \vartheta_F \rangle$ assigns a truth value to every formula. If the formula A is true in a valuation ϑ this is denoted by ${}_\vartheta\models A$.*

- *If* $P \in \mathcal{S}$, ${}_\vartheta\models P$ *iff* $P \in \vartheta_S$
- ${}_\vartheta\models \mathtt{T}_x$ *iff* $x \in \vartheta_T$
- ${}_\vartheta\models \mathtt{F}_x$ *iff* $x \in \vartheta_F$

- $_\vartheta\models \neg A$ *iff* $_\vartheta\not\models A$
- $_\vartheta\models A \wedge B$ *iff* $_\vartheta\models A$ *and* $_\vartheta\models B$
- $_\vartheta\models A \vee B$ *iff* $_\vartheta\models A$ *or* $_\vartheta\models B$
- $_\vartheta\models A \supset B$ *iff* $_\vartheta\not\models A$ *or* $_\vartheta\models B$
- $_\vartheta\models \mathtt{S}_x A$ *iff* ($_\vartheta\models \mathtt{T}_x$ *and* $_\vartheta\models A$) *or* ($_\vartheta\models \mathtt{F}_x$ *and* $_\vartheta\not\models A$)

A formula A is satisfiable *if there exists a valuation ϑ such that $_\vartheta\models A$ and a formula A is valid if at every valuation ϑ, $_\vartheta\models A$.*

In ϑ model the sets of knights, knaves, normals and mutes are $\vartheta_T \cap \overline{\vartheta_F}$, $\overline{\vartheta_T} \cap \vartheta_F$, $\vartheta_T \cap \vartheta_F$ and $\overline{\vartheta_T} \cap \overline{\vartheta_F}$, respectively.

4 Sequent Calculus

In our proofs we shall use the sequent calculus described for example in [1, §48.]. We shall use the notations and definitions of this book, but we shall give informally the basic definitions for whose are unfamiliar with this topic. We do not need the last four rules about quantifiers [1, p. 289], but we need two other rules about *can say*

$$\frac{\Gamma, \mathtt{T}_x, A \longrightarrow \Theta \ ; \ \Gamma, \mathtt{F}_x \longrightarrow A, \Theta}{\Gamma, \mathtt{S}_x A \longrightarrow \Theta} \quad \text{and} \quad \frac{\Gamma, A \longrightarrow \mathtt{T}_x, \Theta \ ; \ \Gamma, \longrightarrow A, \mathtt{F}_x, \Theta}{\Gamma \longrightarrow \mathtt{S}_x A, \Theta}.$$

We say a sequent $\Gamma \longrightarrow \Theta$ is *falsifiable*, if there exists a valuation such that all formulae of Γ are true and all formulae of Θ are false.

5 Soundness

Theorem 4 *For each of 12 rules: The sequent written below the line is falsifiable iff the sequent or at least one of the two sequents written is above the line is falsifiable.*

Proof. For the first 10 rules this was proven in [1], so we prove the claim only for rules of *can say*.

If $\Gamma, \mathtt{S}_x A \longrightarrow \Theta$ is falsifiable then there exists a ϑ such that all formulae of Γ and $\mathtt{S}_x A$ are true and all formulae of Θ are false in ϑ. If $\mathtt{S}_x A$ is true then by definition either A and $\mathtt{T}_x$ are true or A is false and $\mathtt{F}_x$ is true. In the first case $\Gamma, \mathtt{T}_x, A \longrightarrow \Theta$, in the other case $\Gamma, \mathtt{F}_x \longrightarrow A, \Theta$ is falsifiable. To prove it in other direction

1) If $\Gamma, \mathtt{T}_x, A \longrightarrow \Theta$ is falsifiable, then there exists a ϑ such that all formulae of Γ, $\mathtt{T}_x$ and A are true and all formulae of Θ are false in ϑ and by definition $\mathtt{S}_x A$ is true in ϑ so $\Gamma, \mathtt{S}_x A \longrightarrow \Theta$ is falsifiable.

2) If $\Gamma, \mathtt{F}_x \longrightarrow A, \Theta$ is falsifiable, then there exists a ϑ such that all formulae of Γ and $\mathtt{F}_x$ are true and all formulae of Θ and A are false in ϑ and by definition $\mathtt{S}_x A$ is true in ϑ so $\Gamma, \mathtt{S}_x A \longrightarrow \Theta$ is falsifiable.

It easy to check that ${}_\vartheta\not\models \mathtt{S}_x A$ iff (${}_\vartheta\models A$ and ${}_\vartheta\not\models \mathtt{T}_x$) or (${}_\vartheta\not\models A$ and ${}_\vartheta\not\models \mathtt{F}_x$), and the proof about other rule is similar. □

The axioms of the sequent calculus are $\Gamma, A \longrightarrow A, \Theta$. This kind of sequent is not falsifiable, so it is valid. We can prove a formula A in the sequent calculus if we can construct a tree according to the rules such that each path ends in an axiom. Since all axiom are valid, we can go upside-down on the tree line by line and by the lemma above (which states that if the sequents above the line are valid then the sequent below the line is valid, too) all the sequents in the tree are valid; hence A is too. This proves the following theorem:

Theorem 5 *Each provable formula is valid.*

6 Completeness.

We want to prove that any valid formula is provable. At first we shall show that any proof-tree is finite. To do this we define a function:

Definition 6 *On the* rank of a formula *we understand a natural number such that*

- *Rank of propositional letters are* 0.
- *If $x \in \mathcal{P}$ then the ranks of $\mathtt{T}_x$ and $\mathtt{F}_x$ are* 0, *too.*
- *If the rank of A is n, then rank of $\neg A$ and $\mathtt{S}_x A$ are $n+1$.*
- *If the rank of A is n and the rank of B is m then the rank of $A \wedge B$, $A \vee B$ and $A \supset B$ are $m+n+1$, so the ranks of subformulae are smaller than the rank of the formulae.*

The rank of a sequent and the rank of the set of formulae are the sum of the ranks of its formulae.

Lemma 7 *The rank of a sequent above the line is smaller than the rank of the sequent below the line.*

Proof. Let us show this only for one of the new rules. For the others the proof is similar. If the rank of Γ, Θ and $\mathtt{S}_x A$ are n, m and l, respectively, then the rank of $\Gamma, \mathtt{S}_x A \longrightarrow \Theta$, $\Gamma, \mathtt{T}_x, A \longrightarrow \Theta$ and $\Gamma, \mathtt{F}_x \longrightarrow A, \Theta$ are $n+m+l$, $n+m+l-1$ and $n+m+l-1$, respectively. □

When we construct a proof-tree then in each step we reduce the rank of the sequents. This can be done finitely many times, because the rank of the original formula was finite. This proves the following lemma.

Lemma 8 *Every proof-tree is finite.*

Theorem 9 *Every valid formula is provable.*

Proof. Let us assume that there is a valid formula A which is not provable. Take a maximal proof ending with the given formula A. By the lemma above its proof-tree is finite and since the formula is not provable, one path of the tree does not end with an axiom and no rule can be applied here, so this node contains only prime components, namely predicate letters, formulae of types $\mathtt{T}_x$ and $\mathtt{F}_x$. The sequent is not axiom, so the two sets of formulae of this sequent are disjunct, hence falsifiable, and we only need to assign the value true to each formula to the left of the arrow (antecedent) and the value false to each formula to the right of the arrow (succedent). By theorem 4. the sequent below this is falsifiable, too, and repeating the process we get the original formula falsifiable, but we assumed that it was valid. We get a contradiction because we assumed that this formula was unprovable. □

7 A puzzle and some properties.

It is hard to typeset the proof-trees in the original form so we shall use a different notation. We typeset

$$\frac{\Gamma, A \longrightarrow \Theta; \Gamma, B \longrightarrow \Theta}{\Gamma, A \vee B \longrightarrow \Theta}$$

as

$$\begin{array}{c} \boxed{\Gamma, B \longrightarrow \Theta} \\ \boxed{\Gamma, A \longrightarrow \Theta} \\ \Gamma, A \vee B \longrightarrow \Theta \end{array}$$

so the two paths are boxed and positioned vertically.

This logic is not as nice as the logic of belief or logic of knowledge. For example, we do not have here the two common properties **T** and **4**. Fig. 1. contains the proofs. The problematic paths are denoted by a star.

Smullyan did not examined this logic, so no puzzles are for it. After Smullyan it is hard to invent new puzzles but we shall try it: *We met three islander: A, B and C. A said that B cannot say that C is a knight. B said that C cannot say that A is a knight. C said that A cannot say that B is a knight. Let us prove that at least one of them isn't a knight.* We formulate this puzzle by the following formula: $\mathtt{S}_a\neg\mathtt{S}_b(\mathtt{T}_c \wedge \neg\mathtt{F}_c) \wedge \mathtt{S}_b\neg\mathtt{S}_c(\mathtt{T}_a \wedge \neg\mathtt{F}_a) \wedge \mathtt{S}_c\neg\mathtt{S}_a(\mathtt{T}_b \wedge \neg\mathtt{F}_b) \supset \neg(\mathtt{T}_a \wedge \neg\mathtt{F}_a) \vee \neg(\mathtt{T}_b \wedge \neg\mathtt{F}_b) \vee \neg(\mathtt{T}_c \wedge \neg\mathtt{F}_c)$. Without the first step our proof is in Fig. 2.

We have seen in Fig. 1. that two properties are lacking. This is also true for many of usual properties, but the formula $\mathtt{S}_x(A \supset B) \supset (\mathtt{S}_x A \supset \mathtt{S}_x B)$, also known as **K**, is still valid. We prove this in Fig. 3.

$$\boxed{\begin{array}{c} * \\ \mathrm{F}_a \longrightarrow A, A \end{array}}$$

$$\boxed{\mathrm{T}_a, A \longrightarrow A}$$

$$\begin{array}{l} \mathrm{S}_a A \longrightarrow A \\ \longrightarrow \mathrm{S}_a A \supset A \end{array}$$

$$\boxed{\mathrm{S}_a A \longrightarrow \mathrm{S}_a A, \mathrm{F}_a}$$

$$\boxed{\begin{array}{c} \boxed{\begin{array}{c} * \\ \mathrm{F}_a, \mathrm{F}_a \longrightarrow A, A, \mathrm{T}_a \end{array}} \\ \boxed{\mathrm{F}_a, \mathrm{T}_a, A \longrightarrow A, \mathrm{T}_a} \\ \mathrm{S}_a A, \mathrm{F}_a \longrightarrow A, \mathrm{T}_a \\ \boxed{\mathrm{S}_a A, \mathrm{T}_a, A \longrightarrow \mathrm{T}_a} \\ \mathrm{S}_a A, \mathrm{S}_a A \longrightarrow \mathrm{T}_a \end{array}}$$

$$\begin{array}{l} \mathrm{S}_a A \longrightarrow \mathrm{S}_a \mathrm{S}_a A \\ \longrightarrow \mathrm{S}_a A \supset \mathrm{S}_a \mathrm{S}_a A \end{array}$$

Fig. 1.

$$\boxed{\mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{F}_a \longrightarrow \neg \mathrm{S}_b (\mathrm{T}_c \wedge \neg \mathrm{F}_c), \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c}$$

$$\boxed{\begin{array}{c} \boxed{\begin{array}{c} \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a, \mathrm{F}_c \longrightarrow \mathrm{F}_b, \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c \\ \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a \longrightarrow \neg \mathrm{F}_c, \mathrm{F}_b, \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c \end{array}} \\ \boxed{\mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a, \longrightarrow \mathrm{T}_c, \mathrm{F}_b, \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c} \\ \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a, \longrightarrow \mathrm{T}_c \wedge \neg \mathrm{F}_c, \mathrm{F}_b, \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c \\ \boxed{\mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a, \mathrm{T}_c \wedge \neg \mathrm{F}_c \longrightarrow \mathrm{T}_b, \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c} \\ \begin{array}{c} \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a \longrightarrow \mathrm{S}_b (\mathrm{T}_c \wedge \neg \mathrm{F}_c), \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c \\ \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c, \mathrm{T}_a, \neg \mathrm{S}_b (\mathrm{T}_c \wedge \neg \mathrm{F}_c) \longrightarrow \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c \end{array} \end{array}}$$

$$\begin{array}{l} \mathrm{S}_a \neg \mathrm{S}_b (\mathrm{T}_c \wedge \neg \mathrm{F}_c), \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \mathrm{T}_b, \mathrm{T}_c \longrightarrow \mathrm{F}_a, \mathrm{F}_b, \mathrm{F}_c \\ \mathrm{S}_a \neg \mathrm{S}_b (\mathrm{T}_c \wedge \neg \mathrm{F}_c), \mathrm{S}_b \neg \mathrm{S}_c (\mathrm{T}_a \wedge \neg \mathrm{F}_a), \mathrm{S}_c \neg \mathrm{S}_a (\mathrm{T}_b \wedge \neg \mathrm{F}_b), \mathrm{T}_a, \neg \mathrm{F}_a, \mathrm{T}_b, \neg \mathrm{F}_b, \mathrm{T}_c, \neg \mathrm{F}_c \longrightarrow \end{array}$$

Fig. 2.

$$\mathtt{F}_a, A, \mathtt{F}_a \longrightarrow A, B, \mathtt{S}_a B$$

$$\mathtt{F}_a, A, \mathtt{T}_a, A \longrightarrow \mathtt{F}_a, B, B$$

$$\mathtt{F}_a, A, \mathtt{T}_a, A, B \longrightarrow \mathtt{T}_a, B$$

$$\mathtt{F}_a, A, \mathtt{T}_a, A \longrightarrow B, \mathtt{S}_a B$$

$$\mathtt{S}_a A, \mathtt{F}_a, A \longrightarrow B, \mathtt{S}_a B$$

$$\mathtt{S}_a A, \mathtt{F}_a \longrightarrow A \supset B, \mathtt{S}_a B$$

$$\mathtt{T}_a, \mathtt{F}_a \longrightarrow A, B, \mathtt{F}_a, A$$

$$\mathtt{T}_a, A, \mathtt{T}_a \longrightarrow B, \mathtt{F}_a, A$$

$$\mathtt{S}_a A, \mathtt{T}_a \longrightarrow B, \mathtt{F}_a, A$$

$$\mathtt{S}_a A, \mathtt{T}_a, B \longrightarrow \mathtt{T}_a, A$$

$$\mathtt{S}_a A, \mathtt{T}_a \longrightarrow A, \mathtt{S}_a B$$

$$\mathtt{S}_a A, \mathtt{T}_a, B \longrightarrow B, \mathtt{F}_a$$

$$\mathtt{S}_a A, \mathtt{T}_a, B, B \longrightarrow \mathtt{T}_a$$

$$\mathtt{S}_a A, \mathtt{T}_a, B \longrightarrow \mathtt{S}_a B$$

$$\mathtt{S}_a A, \mathtt{T}_a, A \supset B \longrightarrow \mathtt{S}_a B$$

$$\mathtt{S}_a(A \supset B), \mathtt{S}_a A \longrightarrow \mathtt{S}_a B$$

$$\mathtt{S}_a(A \supset B) \longrightarrow \mathtt{S}_a A \supset \mathtt{S}_a B$$

$$\longrightarrow \mathtt{S}_a(A \supset B) \supset (\mathtt{S}_a A \supset \mathtt{S}_a B)$$

Fig. 3.

8 Acknowledgements

The author is grateful to K. Pásztor-Varga and A. Kron for helpful comments and useful discussions. We are also want to pay tribute to A. G. Dragalin who guided us before his death in 1998.

References

[1] S. C. Kleene. *Mathematical Logic.* John Wiley & Sons, Inc., 1967.

[2] R. M. Smullyan. *What is the name of this book? (The riddle of Dracula and other logical puzzles).* Prentice Hall, Inc., 1978.

Received October, 1999

Acta Cybernetica 14 (2000) 541-552.

Equivalence of Mealy and Moore Automata

István Babcsányi *

Abstract

It is proved here that every Mealy automaton is a homomorphic image of a Moore automaton, and among these Moore automata (up to isomorphism) there exists a unique one which is a homomorphic image of the others. A unique simple Moore automaton $\mathbf{M}$ is constructed (up to isomorphism) in the set $MO(\mathbf{A})$ of all Moore automata equivalent to a Mealy automaton $\mathbf{A}$ such that $\mathbf{M}$ is a homomorphic image of every Moore automaton belonging to $MO(\mathbf{A})$. By the help of this construction, it can be decided in steps $|X|^k$ that automaton mappings inducing by states of a k-uniform finite Mealy [Moore] automaton are equal or not. The structures of simple k-uniform Mealy [Moore] automata are described by the results of [1]. It gives a possibility for us to get the k-uniform Mealy [Moore] automata from the simple k-uniform Mealy [Moore] automata. Based on these results, we give a construction for finite Mealy [Moore] automata.

1 Preliminaries

Let X be a nonempty set. A *Mealy automaton* (*over* X) is a system $\mathbf{A} = (A, X, Y, \delta, \lambda)$ consisting of a (nonempty) state set A, the input set X, a (nonempty) output set Y, a transition function $\delta : A \times X \to A$ and a surjective output function $\lambda : A \times X \to Y$.

A *Moore automaton* (*over* X) is a system $\mathbf{A} = (A, X, Y, \delta, \mu)$ consisting of a (nonempty) state set A, the input set X, a (nonempty) output set Y, a transition function $\delta : A \times X \to A$ and a surjective sign function $\mu : A \to Y$.

If A, X and Y are finite, the Mealy [Moore] automaton $\mathbf{A}$ is called *finite*.

For arbitrary Moore automaton $\mathbf{A} = (A, X, Y, \delta, \mu)$, the system $\mathbf{A}_\lambda = (A, X, Y, \delta, \lambda)$ with $\lambda = \mu\delta$ is a Mealy automaton over X. The Mealy automaton $\mathbf{A}_\lambda$ is called *the Mealy automaton associated with the Moore automaton* $\mathbf{A}$. It is said that λ is *the output function of the Moore automaton* $\mathbf{A}$. The Mealy automaton $\mathbf{A} = (A, X, Y, \delta, \lambda)$ fulfils the *Moore criterion* if

$$\delta(a_1, x_1) = \delta(a_2, x_2) \quad \Longrightarrow \quad \lambda(a_1, x_1) = \lambda(a_2, x_2)$$

for every $a_1, a_2 \in A$ and $x_1, x_2 \in X$. If $\mu : A \to Y$ is a surjective mapping such that $\lambda = \mu\delta$, the Moore automaton $\mathbf{A}_\mu = (A, X, Y, \delta, \mu)$ is called *a Moore*

*Department of Algebra, Mathematical Institute, Technical University of Budapest, 1521 Budapest, Műegyetem rkp. 9., Hungary, E-mail: babcs@math.bme.hu

automaton associated with the Mealy automaton $\mathbf{A}$. Furthermore, we say that μ is *a sign function of the Mealy automaton* $\mathbf{A}$. We note that the output function λ is determined by restriction of μ to the subset $\delta(A,X) = \{\delta(a,x); a \in A, x \in X\}$ of A. Thus, the restrictions of all sign functions of the Mealy automaton $\mathbf{A}$ to $\delta(A,X)$ are equal. The Mealy automaton $\mathbf{A} = (A,X,Y,\delta,\lambda)$ is called *real* if there exist $a_1, a_2 \in A$ and $x_1, x_2 \in X$ such that

$$\delta(a_1,x_1) = \delta(a_2,x_2) \quad \text{and} \quad \lambda(a_1,x_1) \neq \lambda(a_2,x_2).$$

Let Z^* and Z^+ denote the free monoid and the free semigroup over a nonempty set Z, respectively. If $\mathbf{A} = (A,X,Y,\delta,\lambda)$ is a Mealy automaton, the functions δ and λ can be extended to $A \times X^*$ in the usual forms as follows:

$$\delta(a,e) = a, \quad \delta(a,px) = \delta(a,p)\delta(ap,x),$$

$$\lambda(a,e) = e, \quad \lambda(a,px) = \lambda(a,p)\lambda(ap,x),$$

where $a \in A$, $p \in X^+$, ap denotes the last letter of $\delta(a,p)$ and e denotes the empty word. ([5], [2]). If $\mathbf{A} = (A,X,Y,\delta,\mu)$ is a Moore automaton, the extension of δ is similar to the case when $\mathbf{A}$ is a Mealy automaton. The extension of μ to A^+ is given by

$$\mu(a_1a_2\ldots a_k) = \mu(a_1)\mu(a_2)\ldots\mu(a_k) \quad (a_1,a_2,\ldots,a_k \in A).$$

It means that if $\lambda = \mu\delta$, then

$$\lambda(a,p) = \mu(\delta(a,p)),$$

for all $a \in A$, $p \in X^+$. But $\lambda(a,e) = e$ and $\mu(\delta(a,e)) = \mu(a)$ for all $a \in A$.

The Mealy [Moore] automaton $\mathbf{A}' = (A',X,Y',\delta',\lambda'[\mu'])$ is a *subautomaton* of the Mealy [Moore] automaton $\mathbf{A}$ if $A' \subseteq A$, $Y' \subseteq Y$, δ' and λ' $[\mu']$ are restrictions of δ and λ $[\mu]$ to $A' \times X$ $[A']$.

Let $\mathbf{A}_i = (A_i,X,Y,\delta_i,\lambda_i[\mu_i])$ $(i = 1,2)$ be arbitrary Mealy [Moore] automata over X. We say that a mapping $\varphi : A_1 \to A_2$ is a *homomorphism* of $\mathbf{A}_1$ into $\mathbf{A}_2$ if

$$\varphi(\delta_1(a,x)) = \delta_2(\varphi(a),x), \quad \lambda_1(a,x) = \lambda_2(\varphi(a),x) \quad [\mu_1(a) = \mu_2(\varphi(a))]$$

for all $a \in A$ and $x \in X$. It is easy to see that

$$\lambda_1(a,p) = \lambda_2(\varphi(a),p)$$

for all $p \in X^*$. The mapping $\varphi : A_1 \to A_2$ is called a *homomorphism* of a Moore automaton $\mathbf{A}_1$ into a Mealy automaton $\mathbf{A}_2$ if φ is a homomorphism of $(\mathbf{A}_1)_\lambda$ into $\mathbf{A}_2$. We note that every homomorphic image of a real Mealy automata is real, too.

Every state $a \in A$ of a Mealy automaton $\mathbf{A}$ induces a mapping $\alpha_a : X^* \to Y^*$ given by $\alpha_a(p) = \lambda(a,p)$ $(p \in X^*)$. The mapping $\alpha : X^* \to Y^*$ is called *automaton mapping* if there exist a Mealy automaton $\mathbf{A}$ and a state $a \in A$ such that $\alpha = \alpha_a$. The mapping $\alpha : X^* \to Y^*$ is an automaton mapping if and only if it preserves the

length of words and the map of every prefix of a word is a prefix of the image word. The Mealy automata $\mathbf{A}$ and $\mathbf{B}$ are called *equivalent* if $\{\alpha_a; a \in A\} = \{\alpha_b; b \in B\}$. The Mealy automaton $\mathbf{A}$ and the Moore automaton $\mathbf{B}$ are *equivalent* if $\mathbf{A}$ and $\mathbf{B}_\lambda$ are equivalent. Similarly, the Moore automata $\mathbf{A}$ and $\mathbf{B}$ are *equivalent* if $\mathbf{A}_\lambda$ and $\mathbf{B}_\lambda$ are equivalent.

An equivalence relation ρ of state set A of a Mealy [Moore] automaton $\mathbf{A}$ is called a *congruence* on $\mathbf{A}$ if

$$(a,b) \in \rho \Longrightarrow (\delta(a,x), \delta(b,x)) \in \rho, \quad \lambda(a,x) = \lambda(b,x) \quad [\mu(a) = \mu(b)]$$

for all $a, b \in A$ and $x \in X$. The ρ-class of $\mathbf{A}$ containing the state a is denoted by $\rho[a]$. The greatest congruence on $\mathbf{A}$ is the relation $\rho_{\mathbf{A}}$ $[\pi_{\mathbf{A}}]$ defined by

$$(a,b) \in \rho_{\mathbf{A}}[\pi_{\mathbf{A}}] \quad \Longleftrightarrow \quad \lambda(a,p) = \lambda(b,p) \quad [\mu(\delta(a,p)) = \mu(\delta(b,p))]$$

for all $p \in X^*$. Denoting the identity relation on the state set A by ι_A, we say that $\mathbf{A}$ is *simple* if $\rho_{\mathbf{A}} = \iota_A$ $[\pi_{\mathbf{A}} = \iota_A]$, that is, $\mathbf{A}$ and $\mathbf{A}/\rho_{\mathbf{A}}$ $[\mathbf{A}/\pi_{\mathbf{A}}]$ are isomorphic.

Since every homomorphic image of a Mealy automaton $\mathbf{A}$ is equivalent to $\mathbf{A}$ ([5], [7]), therefore we can give the automaton mappings with simple Mealy automata. The Mealy automata $\mathbf{A}$ and $\mathbf{B}$ are equivalent if and only if $\mathbf{A}/\rho_{\mathbf{A}}$ and $\mathbf{B}/\rho_{\mathbf{B}}$ are isomorphic ([5]). Thus, simple Mealy automata are equivalent if and only if they are isomorphic. For every Mealy automaton $\mathbf{A}$, there exists a Moore automaton $\mathbf{B}$ such that $\mathbf{A}$ and $\mathbf{B}$ are equivalent ([4], [5], [6]). From this it follows that we can give the automaton mappings by simple Moore automata.

2 Moore automata equivalent to a Mealy automaton

For a Mealy automaton $\mathbf{A} = (A, X, Y, \delta, \lambda)$ over X, let us denote by $\mathbf{A}_Y = (A \times Y, X, Y, \delta_Y, \mu_Y)$ the Moore automaton over X for which

$$\delta_Y((a,y),x) = (\delta(a,x), \lambda(a,x)) \quad \text{and} \quad \mu_Y(a,y) = y \quad (a \in A, y \in Y, x \in X).$$

If $\lambda_Y = \mu_Y \delta_Y$, then

$$\lambda_Y((a,y),x) = \mu_Y(\delta_Y((a,y),x)) = \mu_Y(\delta(a,x), \lambda(a,x)) = \lambda(a,x)$$

for every $a \in A, y \in Y, x \in X$, and hence, $\mathbf{A}_Y$ is equivalent to $\mathbf{A}$.

Lemma 1 *If the Mealy automaton $\mathbf{A}'$ is a homomorphic [isomorphic] image of the Mealy automaton $\mathbf{A}$, then $\mathbf{A}'_Y$ is a homomorphic [isomorphic] image of $\mathbf{A}_Y$.*

Proof. If φ is a homomorphism [isomorphism] of $\mathbf{A}$ onto $\mathbf{A}'$, the mapping $\psi : A \times Y \to A' \times Y$, such that

$$\psi(a,y) = (\varphi(a), y) \quad (a \in A, \ y \in Y),$$

is a homomorphism [isomorphism] of $\mathbf{A}_Y$ onto $\mathbf{A}'_Y$.

Consider the subautomata $\mathbf{M} = (M, X, Y, \delta'_Y, \mu'_Y)$ of $\mathbf{A}_Y$ where for every $a \in A$ there exists $y \in Y$ such that $(a, y) \in M$. Let $M(\mathbf{A})$ be the set of all such subautomata $\mathbf{M}$.

Lemma 2 *The Mealy automaton $\boldsymbol{A}$ is a homomorphic image of every automaton $\boldsymbol{M}$ in $M(\boldsymbol{A})$.*

Proof. It is easy to see that the mapping $\varphi : M \to A$, defined by $\varphi(a, y) = a$ $(a \in A)$, is a homomorphism of $\mathbf{M}_\lambda$ onto $\mathbf{A}$.

Theorem 1 *The Mealy automaton $\boldsymbol{A}_1 = (A_1, X, Y, \delta_1, \lambda_1)$ is a homomorphic image of a Moore automaton $\boldsymbol{A}_2 = (A_2, X, Y, \delta_2, \mu_2)$ if and only if there exists a homomorphic image of $\boldsymbol{A}_2$ in $M(\boldsymbol{A}_1)$.*

Proof. First, we note that every automaton $\mathbf{M} \in M(\mathbf{A}_1)$ is a Moore automaton. By Lemma 2, if there exists a homomorphic image of $\mathbf{A}_2$ in $M(\mathbf{A}_1)$, then $\mathbf{A}_1$ is a homomorphic image of $\mathbf{A}_2$.

Conversely, assume that φ is a homomorphism of the Moore automaton $\mathbf{A}_2$ onto the Mealy automaton $\mathbf{A}_1$. It is evident that by the state set $M = \{(\varphi(b), \mu_2(b)); b \in A_2\}$,

$$\mathbf{M} = (M, X, Y, \delta'_Y, \mu'_Y) \in M(\mathbf{A}_1).$$

We show that the mapping $\psi : A_2 \to M$, defined by

$$\psi(b) = (\varphi(b), \mu_2(b)) \quad (b \in A_2),$$

is a homomorphism of $\mathbf{A}_2$ onto $\mathbf{M}$. It is obvious that the mapping ψ is surjective. For every $b \in A_2$ and $x \in X$

$$\psi(\delta_2(b, x)) = (\varphi(\delta_2(b, x)), \mu_2(\delta_2(b, x))) = (\delta_1(\varphi(b), x), \lambda_2(b, x)) =$$

$$= (\delta_1(\varphi(b), x), \lambda_1(\varphi(b), x)) = \delta'_Y((\varphi(b), \mu_2(b)), x) = \delta'_Y(\psi(b), x),$$

$$\mu_2(b) = \mu'_Y(\varphi(b), \mu_2(b)) = \mu'_Y(\psi(b)).$$

Therefore, ψ is a homomorphism.

Theorem 2 *For every Mealy automaton $\boldsymbol{A}$ (up to isomorphism) there exists a unique automaton $\boldsymbol{M} \in M(\boldsymbol{A})$ which is a homomorphic image of any automaton in $M(\boldsymbol{A})$.*

Proof. First, we give the automaton $\mathbf{M}$. If $A \neq \delta(A, X)$, let κ be a mapping of $A \setminus \delta(A, X)$ into Y. For all $a \in A$, consider the sets $Y_a \subseteq Y$ such that

$$\lambda(b, x) \in Y_a \iff \delta(b, x) = a \quad (b \in A, x \in X).$$

We define the sets M_a $(a \in A)$ as follows. If $a \in \delta(A, X)$, let $M_a = \{(a, y); y \in Y_a\}$, and if $a \notin \delta(A, X)$, let $M_a = \{(a, \kappa(a))\}$. Let $M = \cup_{a \in A} M_a$. Then $\mathbf{M} =$

$(M, X, Y, \delta_Y, \mu_Y) \in M(\mathbf{A})$. Let $M'(\mathbf{A})$ be the set of all such automata $\mathbf{M}$. If $A = \delta(A, X)$, then $|M'(\mathbf{A})| = 1$. We show that if $A \neq \delta(A, X)$, then all automata in $M'(\mathbf{A})$ are isomorphic. Assume that κ_i $(i = 1, 2)$ are arbitrary mappings of $A \setminus \delta(A, X)$ into Y and the automaton $\mathbf{M}_i \in M'(\mathbf{A})$ is defined by the mapping κ_i. It can be easily verified that the mapping $\varphi : M_1 \to M_2$, defined by

$$\varphi(a, y) = \begin{cases} (a, y) & \text{if } y \neq \kappa_1(a); \\ (a, \kappa_2(a)) & \text{if } y = \kappa_1(a), \end{cases}$$

is an isomorphism of $\mathbf{M}_1$ onto $\mathbf{M}_2$.

Now we show that for every $\mathbf{B} \in M(\mathbf{A})$, there is an $\mathbf{M} \in M'(\mathbf{A})$ such that $\mathbf{M}$ is a homomorphic image of $\mathbf{B}$. We define the following partition of the state set B:

$$B_a = \{(a, y); (a, y) \in B\} \quad (a \in A).$$

Take an automaton $\mathbf{M} \in M'(\mathbf{A})$ such that $M_a \subseteq B_a$ $(a \in A)$. By the definition of $M'(\mathbf{A})$, one can see that there exists such an automaton $\mathbf{M}$. Let ψ be an arbitrary mapping of B onto M for which

$$\{\psi(b); b \in B_a\} = M_a \quad \text{and} \quad \forall b \in M_a : \psi(b) = b.$$

It is clear that ψ is a homomorphism of $\mathbf{B}$ onto $\mathbf{M}$.

Lemma 3 ([7]) Let $\mathbf{A}$ be a Mealy automaton and $ME(\mathbf{A})$ be the set of all Mealy automata equivalent to $\mathbf{A}$. Then (up to isomorphism) there exists a unique simple Mealy automaton in $ME(\mathbf{A})$ which is a homomorphic image of every automaton in $ME(\mathbf{A})$.

We have a similar statement for Moore automata which are equivalent to a Mealy automaton.

Theorem 3 *Let $\mathbf{A}$ be a Mealy automaton and $MO(\mathbf{A})$ be the set of all Moore automata which are equivalent to $\mathbf{A}$. Then (up to isomorphism) there exists a unique simple Moore automaton in $MO(\mathbf{A})$ which is a homomorphic image of each automaton in $MO(\mathbf{A})$.*

Proof. Let $\mathbf{A}_0$ denote a simple Mealy automaton in $ME(\mathbf{A})$ which is homomorphic image of any automaton in $ME(\mathbf{A})$. By Lemma 3, such an automaton exists. Moreover, by Theorem 2, (up to isomorphism) there is a unique Moore automaton $\mathbf{M}_0 \in M(\mathbf{A}_0)$ which is homomorphic image of any automaton in $M(\mathbf{A}_0)$. Using the last fact, it can be seen that $\mathbf{M}_0$ is a simple Moore automaton.

Now, let $\mathbf{B}$ be an arbitrary Moore automaton equivalent to $\mathbf{A}$. We prove that $\mathbf{M}_0$ is a homomorphic image of $\mathbf{B}$. Since $\mathbf{B}$ is equivalent $\mathbf{A}$, $\mathbf{B}_\lambda \in ME(\mathbf{A})$, and hence, $\mathbf{A}_0$ is a homomorphic image of $\mathbf{B}$. This implies, by Theorem 1, that there is an $\mathbf{M} \in M(\mathbf{A}_0)$ such that $\mathbf{M}$ is a homomorphic image of $\mathbf{B}$, and therefore, $\mathbf{M}_0$ is a homomorphic image of $\mathbf{B}$ as well.

3 Uniform automata

Let $\mathbf{A} = (A, X, Y, \delta, \lambda[\mu])$ be a Mealy [Moore] automaton over X. Denote by $|p|$ the length of the word $p \in X^*$. Let $X^k = \{p \in X^*; |p| = k\}$ and $X(k) = \{p \in X^*; |p| \leq k\}$. For every nonnegative integer k, we define the equivalence relation η_k on A as follows:

$$(a, b) \in \eta_k \iff \lambda(a, p) = \lambda(b, p) \; [\mu(\delta(a, p)) = \mu(\delta(b, p))]$$

for all $p \in X(k)$. We note that if $\mathbf{A}$ is a Mealy automaton, the relation η_0 is the universal relation on A and η_1 is the output-equivalence of $\mathbf{A}$ ([2]). If $\mathbf{A}$ is a Moore automaton, η_0 is the sign-equivalence of $\mathbf{A}$ ([3]).

Lemma 4 *If a and b are arbitrary states of a Mealy [Moore] automaton $\mathbf{A} = (A, X, Y, \delta, \lambda[\mu])$, then*

$$(a, b) \in \eta_k \iff \lambda(a, p) = \lambda(b, p) \quad [\mu(a) = \mu(b), \; \lambda(a, p) = \lambda(b, p)]$$

for all $p \in X^k$.

Proof. If $(a, b) \in \eta_k$, the statement follows from the definition of η_k.

Conversely, assume that if $\mathbf{A}$ is a Mealy automaton, $\lambda(a, p) = \lambda(b, p)$, and if $\mathbf{A}$ is a Moore automaton, then $\mu(a) = \mu(b)$, $\lambda(a, p) = \lambda(b, p)$ holds for every $p \in X^k$. Take arbitrary words $q, r \in X^*$ such that $|q| \leq k$ and $|r| = k - |q|$. Then

$$\lambda(a, q)\lambda(aq, r) = \lambda(a, qr) = \lambda(b, qr) = \lambda(b, q)\lambda(bq, r).$$

Thus, $\lambda(a, q) = \lambda(b, q)$, which implies our statement.

The Mealy [Moore] automaton $\mathbf{A}$ is called *k-uniform* if $\eta_k = \rho_{\mathbf{A}}[\pi_{\mathbf{A}}]$. The k-uniform Mealy [Moore] automata are $(k+1)$-uniform. Every subautomaton of a k-uniform Mealy [Moore] automaton is k-uniform, too. An arbitrary homomorphic image of a Mealy [Moore] automaton is k-uniform if and only if it is k-uniform. The Mealy [Moore] automaton is said to be *uniform* if there exists a positive integer k such that it is k-uniform. Every finite Mealy [Moore] automaton is k-uniform for some positive integer k. Let α_a and α_b be automaton mappings induced by states a and b of a k-uniform finite Mealy [Moore] automaton $\mathbf{A} = (A, X, Y, \delta, \lambda[\mu])$, respectively. If $\alpha_a(p) = \alpha_b(p)$ for every $p \in X^k$, then $\alpha_a = \alpha_b$. Thus, it can be decided in $|X|^k$ steps whether two automaton mappings of this kind are equal or not.

Theorem 4 *If the Moore automaton $\mathbf{A} = (A, X, Y, \delta, \mu)$ is k-uniform, the Mealy automaton $\mathbf{A}_\lambda$ is (k+1)-uniform.*

Proof. We note that $\mathbf{A}_\lambda$ is (k+1)-uniform if and only if $\rho_{\mathbf{A}_\lambda} = \zeta_{k+1}$, where $(a, b) \in \zeta_{k+1}$ $(a, b \in A)$ if and only if $\lambda(a, p)) = \lambda(b, p))$ for all $p \in X(k+1)$.

Let the Moore automaton $\mathbf{A} = (A, X, Y, \delta, \mu)$ be k-uniform, that is, $\eta_k = \pi_{\mathbf{A}}$. Assume that $(a, b) \in \zeta_{k+1}$. Then,

$$\mu(\delta(a, x)) = \lambda(a, x) = \lambda(b, x) = \mu(\delta(b, x)),$$

$$\mu(\delta(\delta(a, x), q)) = \lambda(\delta(a, x), q) = \lambda(\delta(b, x), q) = \mu(\delta(\delta(b, x), q))$$

for every $x \in X$, $q \in X^k$. Thus, by Lemma 4, $(\delta(a, x), \delta(b, x)) \in \eta_k = \pi_{\mathbf{A}}$. This yields that

$$\lambda(\delta(a, x), r) = \mu(\delta(\delta(a, x), r)) = \mu(\delta(\delta(b, x), r)) = \lambda(\delta(b, x), r)$$

for all $r \in X^+$. Therefore, $(\delta(a, x), \delta(b, x)) \in \zeta_{k+1}$, that is, ζ_{k+1} is a congruence on $\mathbf{A}_\lambda$. Thus, $\zeta_{k+1} = \rho_{\mathbf{A}_\lambda}$. From this we get that $\mathbf{A}_\lambda$ is (k+1)-uniform.

Theorem 5 *The Mealy [Moore] automaton $\boldsymbol{A} = (A, X, Y, \delta, \lambda[\mu])$ is k-uniform if and only if $\eta_k = \eta_{k+1}$.*

Proof. Assume that the Mealy [Moore] automaton $\mathbf{A}$ is k-uniform, that is, $\eta_k = \rho_{\mathbf{A}}$. Since $\eta_{k+1} \subseteq \eta_k$ and $\cap_{k=0}^{\infty} \eta_k = \rho_{\mathbf{A}}[\pi_{\mathbf{A}}]$, therefore $\eta_k = \eta_{k+1}$.

Conversely, assume that $\eta_k = \eta_{k+1}$. If $\mathbf{A}$ is a Mealy automaton, η_0 is the universal relation on A. If $\eta_0 = \eta_1$, the relation η_1 is a congruence on $\mathbf{A}$. It yields that $\eta_0 = \eta_1 = \rho_{\mathbf{A}}$. Furthermore let us assume that $\mathbf{A}$ is a Mealy automaton and $1 \leq k$. Let $(a, b) \in \eta_k$. Since $\eta_k = \eta_{k+1}$, then $(a, b) \in \eta_{k+1}$. By Lemma 4, $\lambda(a, xp) = \lambda(b, xp)$ for every $x \in X$ and $p \in X^k$. From this it follows that

$$\lambda(\delta(a, x), p) = \lambda(\delta(b, x), p).$$

Moreover, if $\mathbf{A}$ is a Moore automaton,

$$\mu(\delta(a, x)) = \lambda(a, x) = \lambda(b, x) = \mu(\delta(b, x)),$$

that is, $(\delta(a, x), \delta(b, x)) \in \eta_k$. This results in that η_k is a congruence on $\mathbf{A}$, and so $\eta_k = \rho_{\mathbf{A}}[\pi_{\mathbf{A}}]$. Hence, $\mathbf{A}$ is k-uniform.

Lemma 5 *If a and b are arbitrary states of a Mealy [Moore] automaton $\boldsymbol{A} = (A, X, Y, \delta, \lambda[\mu])$, then*

$$(a, b) \in \eta_{k+1} \iff (a, b) \in \eta_k \quad \textit{and} \quad (\delta(a, x), \delta(b, x)) \in \eta_k, \quad \textit{for all} \quad x \in X.$$

Proof. Assume that $(a, b) \in \eta_{k+1}$. Since $\eta_{k+1} \subseteq \eta_k$, then $(a, b) \in \eta_k$. By Lemma 4, $\lambda(a, xp) = \lambda(b, xp)$ for every $x \in X$ and $p \in X^k$. But

$$\lambda(a, x)\lambda(\delta(a, x), p) = \lambda(a, xp) = \lambda(b, xp) = \lambda(b, x)\lambda(\delta(b, x), p),$$

and so

$$\lambda(\delta(a, x), p) = \lambda(\delta(b, x), p).$$

Moreover, if $\mathbf{A}$ is a Moore automaton,

$$\mu(\delta(a,x)) = \lambda(a,x) = \lambda(b,x) = \mu(\delta(b,x)).$$

By Lemma 4, this yields that $(\delta(a,x), \delta(b,x)) \in \eta_k$.

Conversely, assume that $(a,b) \in \eta_k$ and $(\delta(a,x), \delta(b,x)) \in \eta_k$ for every $x \in X$. If $x \in X$ and $q \in X^k$, then $\lambda(a,x) = \lambda(b,x)$ and $\lambda(\delta(a,x),q) = \lambda(\delta(b,x),q)$. From this it follows that

$$\lambda(a,xq) = \lambda(a,x)\lambda(\delta(a,x)q) = \lambda(b,x)\lambda(\delta(b,x),q) = \lambda(b,xq).$$

Moreover, if $\mathbf{A}$ is a Moore automaton, $\mu(a) = \mu(b)$. By Lemma 4, $(a,b) \in \eta_{k+1}$.

Theorem 6 *For every Mealy automaton $\mathbf{A} = (A, X, Y, \delta, \lambda)$, $\mathbf{A}_Y$ is k-uniform [simple] if and only if $\mathbf{A}$ is k-uniform [simple].*

Proof. If $a \in A$, $y \in Y$ and $p \in X^+$, then $\mu_Y(\delta_Y((a,y),p)) = \lambda(a,p)$.

We note that $\mathbf{A}_Y$ is k-uniform if $\pi_{\mathbf{A}_Y} = \zeta_k$, where ζ_k is an equivalence relation on $A \times Y$ for which

$$((a,y_1),(b,y_2)) \in \zeta_k \iff \mu_Y(\delta_Y((a,y_1),p)) = \mu_Y(\delta_Y((b,y_2),p))$$

for all $p \in X(k)$.

Assume that the Mealy automaton $\mathbf{A}$ is k-uniform. Consider two arbitrary elements (a,y_1) and (b,y_2) of $A \times Y$ with $((a,y_1),(b,y_2)) \in \zeta_k$. Then

$$y_1 = \mu_Y(a,y_1) = \mu_Y(b,y_2) = y_2,$$

$$\lambda(a,p) = \mu_Y(\delta_Y((a,y_1),p)) = \mu_Y(\delta_Y((b,y_2),p)) = \lambda(b,p)$$

for all $p \in X^k$. By Lemma 4, this implies $(a,b) \in \eta_k = \rho_{\mathbf{A}}$. By Theorem 5, $(a,b) \in \eta_{k+1}$, that is,

$$\mu_Y(\delta_Y((a,y_1),p)) = \lambda(a,p) = \lambda(b,p) = \mu_Y(\delta_Y((b,y_2),p))$$

for all $p \in X^{k+1}$ which results in $(a,b) \in \zeta_{k+1}$. Thus, $\zeta_k = \zeta_{k+1}$. By Theorem 5, $\mathbf{A}_Y$ is k-uniform.

Conversely, assume that $\mathbf{A}_Y$ is k-uniform. Let $(a,b) \in \eta_k$. If $y \in Y$, then $((a,y),(b,y)) \in \zeta_k = \pi_{\mathbf{A}_Y}$. By Theorem 5, $((a,y),(b,y) \in \zeta_{k+1}$, and thus $(a,b) \in \eta_{k+1}$. Therefore, $\eta_k = \eta_{k+1}$, that is, $\mathbf{A}$ is k-uniform.

We can prove, in a similar way, that $\mathbf{A}_Y$ is simple if and only if $\mathbf{A}$ is simple (see Lemma 2 in [1]).

By Theorem 6 and Lemma 2, every k-uniform Mealy automaton is equivalent to a k-uniform Moore automaton. By Theorem 3, among these Moore automata (up to isomorphism) there exists a unique simple k-uniform Moore automaton which is a homomorphic image of these Moore automata, that is, the cardinality of its state set is the least among these Moore automata.

4 Simple uniform automata

In this part of the paper, we describe the structure of the simple uniform Mealy [Moore] automata using the results of paper [1].

Lemma 6 (Lemma 3 in [1]) Every subautomaton of a simple Mealy [Moore] automaton $\mathbf{A}$ over X is simple and the subautomata of $\mathbf{A}$ are isomorphic if and only if they are equal.

Denote the set of mappings $\alpha^{(i)} : X^i \to Y$ by $\mathcal{A}^{(i)}$ for every integer $i > 0$. Consider the set $\mathcal{A} = \prod_{i=1}^{\infty} \mathcal{A}^{(i)}$. Let

$$\alpha = (\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(i)}, \ldots) \quad (\alpha^{(i)} \in \mathcal{A}^{(i)}),$$

$$\alpha_x^{(i)}(x_1, x_2, \ldots, x_i) = \alpha^{(i+1)}(x, x_1, x_2, \ldots, x_i) \quad (x, x_1, x_2, \ldots, x_i \in X),$$

$$\alpha_x = (\alpha_x^{(1)}, \alpha_x^{(2)}, \ldots, \alpha_x^{(i)}, \ldots).$$

Assume that $\alpha_e = \alpha$ and let $\alpha_{px} = (\alpha_p)_x$ for every $p \in X^*$ and $x \in X$. Define the Mealy automaton $\underline{\mathcal{A}} = (\mathcal{A}, X, Y, \delta, \lambda)$ with transition and output functions:

$$\delta(\alpha, x) = \alpha_x, \quad \lambda(\alpha, x) = \alpha^{(1)}(x) \quad (\alpha \in \mathcal{A},\ x \in X).$$

Theorem 7 (Theorem 4 in [1]) The Mealy automaton $\underline{\mathcal{A}}$ is simple. A Mealy automaton $\mathbf{A} = (A, X, Y', \delta, \lambda)$ over X is simple if and only if it is isomorphic to a subautomaton of $\underline{\mathcal{A}}$, where $Y' \subseteq Y$.

Theorem 8 (Theorem 5 in [1]) The Moore automaton $\underline{\mathcal{A}}_Y$ is simple and $\underline{\mathcal{A}}$ is a homomorpic image of $\underline{\mathcal{A}}_Y$. A Moore automaton $\mathbf{A} = (A, X, Y', \delta, \mu)$ $(Y' \subseteq Y)$ over X is simple if and only if it is isomorphic to a subautomaton of $\underline{\mathcal{A}}_Y$.

Consider the set $\mathcal{A}_k = \prod_{i=1}^{k} \mathcal{A}^{(i)}$ and a mapping $g : \mathcal{A}_k \to \mathcal{A}^{(k+1)}$. Let

$$\alpha_k = (\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}) \quad (\alpha^{(i)} \in \mathcal{A}^{(i)}),$$

$$\alpha_{k,g,x} = (\alpha_x^{(1)}, \alpha_x^{(2)}, \ldots, \alpha_x^{(k)}),$$

where $\alpha^{(k+1)} = g(\alpha_k)$.

We define the Mealy automaton $\underline{\mathcal{A}}_{k,g} = (\mathcal{A}_k, X, Y, \delta, \lambda)$ with the following transition and output functions:

$$\delta(\alpha_k, x) = \alpha_{k,g,x}, \quad \lambda(\alpha_k, x) = \alpha^{(1)}(x) \quad (\alpha_k \in \mathcal{A}_k,\ x \in X).$$

Consider a nonempty set $H_0 \subseteq \mathcal{A}_k$. It is evident that

$$H_j = \{\alpha_{k,g,x}; \alpha_k \in H_{j-1}, x \in X\} \subseteq \mathcal{A}_k \ (j = 1, 2, \ldots).$$

If $H^{(j)} = H_0 \cup H_1 \cup \ldots \cup H_j$ for every nonnegative integer j, then $\mathbf{H}^{(j)}$ is a subautomaton of $\underline{\mathcal{A}}_{k,g}$ if and only if $H^{(j+1)} \subseteq H^{(j)}$. We note that if X and Y are finite sets, then there exists a nonnegative integer j such that $H^{(j+1)} \subseteq H^{(j)}$.

Theorem 9 *A Mealy automaton* $\mathbf{A}$ *over* X *is simple k-uniform if and only if there exists a mapping* $g : \mathcal{A}_k \to \mathcal{A}^{(k+1)}$ *such that* $\mathbf{A}$ *is isomorphic to some subautomaton of* $\underline{\mathcal{A}}_{k,g}$.

Proof. As in the proof of Theorem 7, we can show that the Mealy automaton $\underline{\mathcal{A}}_{k,g}$ is simple. By Lemma 6, every subautomaton of $\underline{\mathcal{A}}_{k,g}$ is simple. On the other hand, it is easy to verify that the subautomata of $\underline{\mathcal{A}}_{k,g}$ are k-uniform.

Therefore, by Theorem 7, it is sufficient to show that every k-uniform subautomaton of $\underline{\mathcal{A}}$ is isomorphic to an automaton $\mathbf{H}^{(j)}$. Let $\underline{\mathcal{A}}' = (\mathcal{A}', X, Y', \delta_{\mathcal{A}'}, \lambda_{\mathcal{A}'})$ be a k-uniform subautomaton of $\underline{\mathcal{A}}$. Let

$$\alpha_k = (\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)})$$

for every $\alpha = (\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}, \ldots) \in \mathcal{A}'$. Define a mapping $g : \mathcal{A}_k \to \mathcal{A}^{(k+1)}$ such that $g(\alpha_k) = \alpha^{(k+1)}$ for every $\alpha \in \mathcal{A}'$. Let $H_0 = \{\alpha_k; \alpha \in \mathcal{A}'\}$. Since $\underline{\mathcal{A}}'$ is a subautomaton of $\underline{\mathcal{A}}$, then $H_1 \subseteq H_0$. Thus, $\mathbf{H}^{(0)}$ is a subautomaton of $\underline{\mathcal{A}}_{k,g}$. The mapping $\varphi : \mathcal{A}' \to H_0$, for which $\varphi(\alpha) = \alpha_k$ $(\alpha \in \mathcal{A}')$, is an isomorphism of $\underline{\mathcal{A}}'$ onto $\mathbf{H}^{(0)}$.

Every finite Mealy [Moore] automaton is k-uniform for some nonnegative integer k. Thus, we get easily the following theorem from Theorem 9.

Theorem 10 *A finite Mealy automaton* $\mathbf{A}$ *over* X *is simple if and only if there exist a nonnegative integer* k *and a mapping* $g : \mathcal{A}_k \to \mathcal{A}^{(k+1)}$ *for which* $\mathbf{A}$ *is isomorphic to some subautomaton of* $\underline{\mathcal{A}}_{k,g}$.

By Theorems 6, 9 and 10, the following two theorems are true.

Theorem 11 *A Moore automaton* $\mathbf{A}$ *over* X *is simple k-uniform if and only if it is isomorphic to some subautomaton of* $(\underline{\mathcal{A}}_{k,g})_Y$.

Theorem 12 *A finite Moore automaton* $\mathbf{A}$ *over* X *is simple if and only if there exists a nonnegative integer* k *for which it is isomorphic to some subautomaton of* $(\underline{\mathcal{A}}_{k,g})_Y$.

Let $\underline{\mathcal{C}} = (\mathcal{C}, X, Y', \delta_{\mathcal{C}}, \lambda_{\mathcal{C}})$ be a subautomaton of the automaton $\underline{\mathcal{A}}$. Consider a family of nonempty sets U_α $(\alpha \in \mathcal{C})$ such that $U_\alpha \cap U_\beta = \emptyset$ if $\alpha \neq \beta$. Let $U_{\mathcal{C}} = \cup_{\alpha \in \mathcal{C}} U_\alpha$. For all $x \in X$ and $\alpha \in \mathcal{C}$, let $\varphi_{\alpha,x}$ be a mapping of U_α into U_{α_x}. Define the functions $\delta_{U_{\mathcal{C}}}(a, x) = \varphi_{\alpha,x}(a)$ and $\lambda_{U_{\mathcal{C}}}(a, x) = \alpha^{(1)}(x)$ for all $a \in U_\alpha$, $\alpha \in \mathcal{C}$ and $x \in X$. It can be easily verified that $\mathbf{U}_{\mathcal{C}} = (U_{\mathcal{C}}, X, Y', \delta_{U_{\mathcal{C}}}, \lambda_{U_{\mathcal{C}}})$ is a Mealy automaton ([2]).

Lemma 7 *Every Mealy automaton* $\mathbf{A} = (A, X, Y', \delta, \lambda)$ $(Y' \subseteq Y)$ *equals an automaton* $\mathbf{U}_{\mathcal{C}}$.

Proof. By Theorem 7, there exists an isomorphism φ of $\mathbf{A}/\rho_{\mathbf{A}}$ onto a subautomaton $\underline{C}$ of $\underline{A}$. Assume that $\varphi(\rho_{\mathbf{A}}[a]) = \alpha_a$, $U_{\alpha_a} = \rho_{\mathbf{A}}[a]$ $(a \in A)$, $\varphi_{\alpha_a,x} = \delta(a,x)$ and $U_C = \cup_{a\in A} U_{\alpha_a}$. Since

$$\lambda(a,x) = \lambda_{A/\rho_{\mathbf{A}}}(\rho_{\mathbf{A}}[a],x) = \lambda_{\underline{C}}(\alpha_a,x) = \alpha_a^{(1)}(x) = \lambda_{U_C}(a,x),$$

therefore $\mathbf{A} = \mathbf{U}_C$.

Theorem 13 *The automaton U_C is k-uniform if and only if $\underline{C}$ is simple k-uniform.*

Proof. It is evident that if the automaton $\mathbf{U}_C$ is k-uniform, then $\underline{C}$ is simple k-uniform.

Conversely, assume that the automaton $\underline{C}$ is simple k-uniform. Assume that $(a,b) \in \eta_k$ for some $a \in U_\alpha$ and $b \in U_\beta$. Then, by Lemma 4, for every $p \in X^k$

$$\lambda_C(\alpha,p) = \lambda_{U_C}(a,p) = \lambda_{U_C}(b,p) = \lambda_C(\beta,p).$$

But $\underline{C}$ is simple k-uniform, thus $\alpha = \beta$, that is, $a,b \in U_\alpha$. It means that $ap, bp \in U_{\alpha_p}$. Then, for all $x \in X$,

$$\lambda_{U_C}(a,px) = \lambda_{U_C}(a,p)\lambda_{U_C}(ap,x) = \lambda_{U_C}(b,p)\lambda_{U_C}(bp,x) = \lambda_{U_C}(b,px),$$

that is $(a,b) \in \eta_{k+1}$. By Theorem 5, $\mathbf{U}$ is a k-uniform automaton.

By Theorems 6 and 13, we get the following theorem:

Theorem 14 *The automaton $(U_C)_Y$ is k-uniform if and only if $\underline{C}_Y$ is simple k-uniform.*

By Theorems 10 and 12, we give a construction for finite simple Mealy and Moore automata. Thus, by using Theorems 13 and 14, we can give all finite Mealy and Moore automata.

Acknowledgement. The author express his thank to B. Imreh for his valuable comments on the original version of the manuscript.

References

[1] Babcsányi, I., *Simple Mealy and Moore automata*, Proceedings of the International Conference on Automata and Formal Languages IX, Vasszécseny, Hungary, August 9 - 13, 1999, Publicationes Mathematicae (to appear).

[2] Babcsányi, I. and A. Nagy, *Mealy-automata in which the output-equivalence is a congruence*, Acta Cybernetica, 11 (1994), 121-126.

[3] Babcsányi, I. and A. Nagy, *Moore-automata in which the sign-equivalence is a Moore-congruence*, Publicationes Mathematicae, 47 (1995), 393-401.

[4] Bloh, A. S., *O zadacsah, resajemüh poszledovatyelnosztnümi masinami*, Probl. kibernetyiki, 3 (1960), 81-88.

[5] Gécseg, F. and I. Peák, *Algebraic Theory of Automata*, Akadémiai Kiadó, Budapest, 1972.

[6] Gill, A., *Comparison of finite-state models*, IRE Trans. Circuit Theory, 7 (1960), 178-179.

[7] Gluskov, V. M., *Absztraktnaja tyeorija avtomatov*, Uszpehi matyem. nauk, 16 (1961), 3-62.

Received January, 2000

Acta Cybernetica **14** (2000) 553-567.

Pseudo-Hamiltonian Graphs

Luitpold Babel * Gerhard J. Woeginger †

Abstract

A pseudo-h-hamiltonian cycle in a graph is a closed walk that visits every vertex exactly h times. We present a variety of combinatorial and algorithmic results on pseudo-h-hamiltonian cycles.

First, we show that deciding whether a graph is pseudo-h-hamiltonian is NP-complete for any given $h \geq 1$. Surprisingly, deciding whether there exists an $h \geq 1$ such that the graph is pseudo-h-hamiltonian, can be done in polynomial time. We also present sufficient conditions for pseudo-h-hamiltonicity that are based on stable sets and on toughness. Moreover, we investigate the computational complexity of finding pseudo-h-hamiltonian cycles on special graph classes like bipartite graphs, split graphs, planar graphs, cocomparability graphs; in doing this, we establish a precise separating line between easy and difficult cases of this problem.

1 Introduction

For an integer $h \geq 1$, we shall say that an undirected graph $G = (V, E)$ is *pseudo-h-hamiltonian* if there exists a circular sequence of $h \cdot |V|$ vertices such that

- every vertex of G appears precisely h times in the sequence, and
- any two consecutive vertices in the sequence are adjacent in G.

A sequence with these properties will be termed a *pseudo-h-hamiltonian cycle*. In this sense, *pseudo-1-hamiltonian* corresponds to the standard notion *hamiltonian*, and a *pseudo-1-hamiltonian cycle* is just a *hamiltonian cycle*. The *pseudo-hamiltonicity number* $\mathrm{ph}(G)$ of the graph G, is the smallest integer $h \geq 1$ for which G is pseudo-h-hamiltonian; in case no such h exists, $\mathrm{ph}(G) = \infty$. A graph G with finite $\mathrm{ph}(G)$ is called *pseudo-hamiltonian*. Pseudo-h-hamiltonicity is a non-trivial graph property. E.g. for every $h \geq 2$, the graph G_h that results from glueing together h triangles at one of their vertices, is pseudo-h-hamiltonian but it is not pseudo-$(h-1)$-hamiltonian.

*Institut für Mathematik, TU München, D-80290 München, Germany. This author was supported by the Deutsche Forschungsgemeinschaft (DFG)., e-mail:babel@statistik.tu-muenchen.de.

†Institut für Mathematik B, TU Graz, Steyrergasse 30, A-8010 Graz, Austria. This author acknowledges support by the Start-program Y43-MAT of the Austrian Ministry of Science., e-mail:gwoegi@opt.math.tu-graz.ac.at.

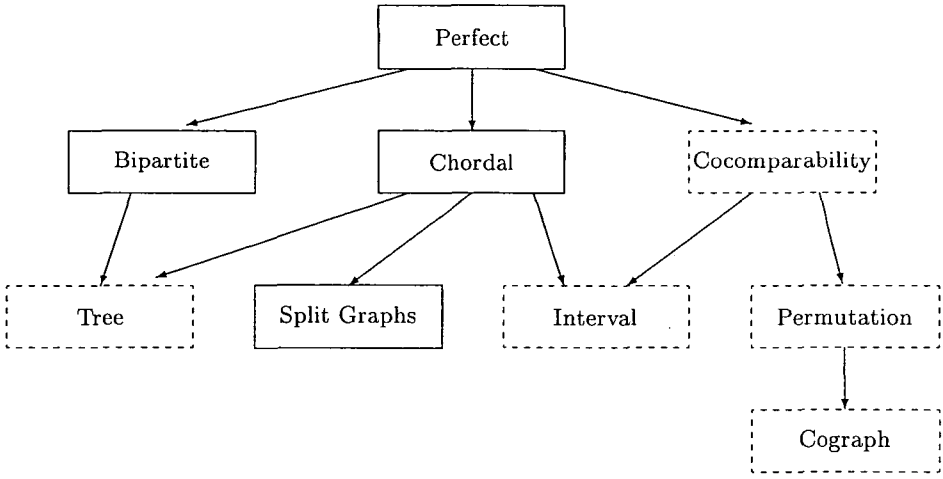

Figure 1: Complexity results for some of the treated graph classes. NP-complete problems have a solid frame, polynomially solvable problems have a dashed frame.

Results of this paper. The problem of deciding whether a given graph is hamiltonian is NP-complete. Hence, it is not surprising at all that for each fixed value of $h \geq 1$, the problem of deciding whether $\mathrm{ph}(G) \leq h$ holds for a given graph G is also NP-complete. However, if we just ask whether $\mathrm{ph}(G) < \infty$, i.e. whether there exists some value of h for which G is pseudo-h-hamiltonian, then we can answer this question in polynomial time (and this is perhaps surprising). This polynomial time result is based on the close relationship of pseudo-hamiltonian graphs with *regularizable* graphs (cf. Section 2).

We also provide a nice and simple characterization of pseudo-hamiltonian graphs that is based on the stable sets of vertices of the graph. We show that every pseudo-hamiltonian graph G must be $1/\mathrm{ph}(G)$-tough, and that every 1-tough graph is pseudo-hamiltonian. The square of a connected graph is always pseudo-hamiltonian. For d-regular graphs with $d \geq 3$, we derive a tight result of the following form: There exists a threshold $\tau(d)$ such that for $h < \tau(d)$, it is NP-complete to decide whether a d-regular graph is pseudo-h-hamiltonian, whereas for every $h \geq \tau(d)$, a d-regular graph automatically is pseudo-h-hamiltonian. Hence, the computational complexity of deciding pseudo-h-hamiltonicity of regular graphs jumps at $\tau(d)$ from trivial immediately to NP-complete.

Finally, we will investigate the computational complexity of computing $\mathrm{ph}(G)$ on many well-known special graph classes, like bipartite graphs, split graphs, partial k-trees, interval graphs, planar graphs etc. Figure 1 summarizes some of our results together with some of their implications for special graph classes. Directed arcs represent containment of the lower graph class in the upper graph class. For

classes with a solid frame, the computation of $\mathrm{ph}(G)$ is NP-complete, and for classes with a dashed frame, this problem is polynomial time solvable (for exact definitions of all these graph classes cf. Johnson [12]). Note that the results for trees, bipartite graphs, split graphs and cocomparability graphs imply all the other results in Figure 1.

Organization of the paper. Section 2 investigates the connections between pseudo-hamiltonicity and regularizable graphs, and it states several general complexity results. Section 3 relates pseudo-hamiltonicity to stable sets, to connectivity and to toughness. Section 4 derives the complexity threshold for d-regular graphs, and Section 5 deals with squares of graphs. Finally, Section 6 collects the complexity results for the special graph classes.

Notation and conventions. Throughout this paper, we only consider undirected graphs. All graphs have at least three vertices. For convenience we often write $G - W$ instead of $G(V - W)$, the graph that results from removing the vertices in W together with all incident edges from G. For a set $W \subseteq V$, we denote by $N(W)$ the set of all vertices outside W which are adjacent to vertices from W. A stable set is a set of pairwise non-adjacent vertices. A stable set S is maximal if there is no stable set S' which properly contains S. The stability number $\alpha(G)$ is the size of a largest stable set in G.

2 Complexity aspects of pseudo-hamiltonicity

In this section, we give several characterizations of pseudo-hamiltonian graphs that are based on regularizable graphs. These characterizations imply that one can decide in polynomial time whether $\mathrm{ph}(G) < \infty$. On the other hand, we will show that for every fixed integer $h \geq 1$ it is NP-complete to decide whether $\mathrm{ph}(G) \leq h$.

A graph $G = (V, E)$ is called *regularizable* (see Berge [2, 3]), if for each edge $e \in E$ there is a positive integer $m(e)$ such that the multigraph which arises from G by replacing every edge e by $m(e)$ parallel edges is a regular graph. A useful characterization of regularizable graphs can be found in Berge [2].

Proposition 2.1 *(Berge [2])*
A connected graph $G = (V, E)$ is regularizable if and only if one of the following two statements holds

(a) G is elementary bipartite
(i.e. G is bipartite, connected and every edge of G appears in a perfect matching);

(b) G is 2-bicritical
(i.e. $|N(S)| > |S|$ holds for every stable set $S \subseteq V$). □

Regularizable graphs are related to pseudo-hamiltonian graphs as follows.

Lemma 2.2 *A graph G is pseudo-hamiltonian if and only if G has a connected spanning regularizable subgraph.*

Proof. (Only if). Clearly, in a pseudo-h-hamiltonian cycle (considered as a multigraph) each vertex has degree $2h$. Hence, the *skeleton* of a pseudo-h-hamiltonian cycle (that is, the simple graph arising from replacing parallel edges by simple edges) of a graph G constitutes a regularizable subgraph of G which, additionally, is connected and contains all the vertices of G.

(If). Conversely, assume that a graph G has a connected spanning regularizable subgraph H. Let H^* denote the associated regular multigraph, say of degree $2h$ (if the degree of the regular multigraph is odd, multiply every number $m(e)$ by two). Clearly, H^* has an Eulerian cycle. This Eulerian cycle corresponds to a pseudo-h-hamiltonian cycle in G. □

A graph has a *perfect 2-matching* if one can assign weights 0, 1 or 2 to its edges in such a way that for each vertex, the sum of the weights of the incident edges is equal to 2. The following characterization of regularizable graphs can be found in the book by Lovász and Plummer [13].

Proposition 2.3 *(Lovász and Plummer [13])*
A graph $G = (V, E)$ is regularizable if and only if for each edge $e \in E$ there exists a perfect 2-matching of G in which e has weight 1 or 2. □

Proposition 2.3 has several important consequences.

Corollary 2.4 *(i) For any integer h with $1 \leq h < \mathrm{ph}(G)$, graph G does not possess a pseudo-h-hamiltonian cycle. (ii) For any integer $h \geq \mathrm{ph}(G)$, graph G does possess a pseudo-h-hamiltonian cycle.*

Proof. Statement (i) trivially follows from the definition of $\mathrm{ph}(G)$. In order to prove (ii), we show that if a graph has a pseudo-h-hamiltonian cycle then it also has a pseudo-$(h+1)$-hamiltonian cycle: Let C be a pseudo-h-hamiltonian cycle in G. Then the skeleton of C is regularizable, and consequently possesses a perfect 2-matching. If one adds this perfect 2-matching to the $2h$-regular multigraph that corresponds to C, one gets a $(2h+2)$-regular multigraph that corresponds to a pseudo-$(h+1)$-hamiltonian cycle. □

Proposition 2.3 together with Lemma 2.2 also allows us to construct an algorithm to decide efficiently whether a graph is pseudo-hamiltonian (or, equivalently, to decide whether a graph has a connected spanning regularizable subgraph). The algorithm repeatedly runs through all the edges of the graph and deletes all those edges which do not allow a perfect 2-matching with the desired property. If the remaining graph is disconnected then G is not pseudo-hamiltonian. Otherwise, one obtains a connected spanning regularizable subgraph of G, i.e. G is pseudo-hamiltonian.

Algorithm PSEUDO-HAMILTON(G)

1. UNCHECKED:= E; E^*:= E;
2. **While** UNCHECKED $\neq \emptyset$ **do**
   Pick an arbitrary edge $e \in$UNCHECKED;
   Check whether the graph (V, E^*) possesses a perfect 2-matching in which edge e has weight 1 or 2;
   If there is no such perfect 2-matching
   then E^*:= $E^* - \{e\}$;
   UNCHECKED:= UNCHECKED$-\{e\}$;
3. **If** the graph (V, E^*) is connected
   then return 'yes' **else return** 'no'.

Since perfect 2-matchings can be found in polynomial time (cf. Lovász and Plummer [13]), the whole algorithm can be implemented to run in polynomial time.

Theorem 2.5 *It can be decided in polynomial time, whether* $\mathrm{ph}(G) < \infty$ *holds for a given graph* G. □

In strong contrast to Theorem 2.5, it is NP-complete to compute $\mathrm{ph}(G)$ exactly.

Theorem 2.6 *For every fixed value* $h \geq 1$, *the problem of deciding whether* $\mathrm{ph}(G) \leq h$ *holds for a given graph* G *is NP-complete.*

Proof. It is well known that deciding pseudo-1-hamiltonicity (i.e. standard hamiltonicity) of a graph is NP-complete. Let $h \geq 2$ be some fixed integer. Consider some undirected graph $G' = (V', E')$, and construct another graph $G = (V, E)$ from it as follows: V contains the vertices in V' together with $3(h-1)|V'|$ new vertices. For every vertex $v \in V'$, there are $3h-3$ new vertices that are called a_v^i, b_v^i, and c_v^i, where $i = 1, \ldots, h-1$. The edge set E contains all edges in E' together with $4(h-1)|V'|$ new edges. For every vertex $v \in V'$, there are $4h-4$ new edges (v, a_v^i), (a_v^i, b_v^i), (b_v^i, c_v^i), and (c_v^i, a_v^i), where $i = 1, \ldots, h-1$. We claim that the constructed graph G possesses a pseudo-h-hamiltonian cycle if and only if the original graph G' possesses a hamiltonian cycle.

(Only if). Assume that G possesses a pseudo-h-hamiltonian cycle C. Consider for arbitrary $v \in V'$ and $1 \leq i \leq h-1$ the connected component consisting of a_v^i, b_v^i, and c_v^i. The cycle C can visit and leave this component only via the edge (v, a_v^i), and this edge must be used an even number of times. Hence, C uses at least $2h-2$ edges incident to v just for visiting the $(h-1)$ attached components. There remain only two edges that can connect v to other vertices in V', and it is easy to see that these pairs of edges taken over all vertices in V' correspond to a hamiltonian cycle in G'.

(If). Now assume that G' possesses a hamiltonian cycle. Construct a multigraph with vertex set V as follows: The multigraph contains all edges that are used by the hamiltonian cycle. Moreover, it contains for every $v \in V'$ and for every i, $1 \leq i \leq h-1$, two copies of the edge (v, a_v^i), $h-1$ copies of the edge (a_v^i, b_v^i), $h+1$ copies of the edge (b_v^i, c_v^i), and $h-1$ copies of the edge (c_v^i, a_v^i). The resulting

multigraph is connected and $2h$-regular. Hence, it contains an Eulerian cycle that corresponds to a pseudo-h-hamiltonian cycle in a natural way. □

Question 2.7 *What can be said about approximating* $\mathrm{ph}(G)$*? Can one always find in polynomial time a, say, pseudo-*$2\mathrm{ph}(G)$*-hamiltonian cycle?*

3 Stable sets, connectivity and toughness

This section discusses the relationship of pseudo-hamiltonicity with the structure of stable subsets, with the connectivity of a graph, and with the toughness of a graph. First, consider the following two conditions (C1) and (C2) on a graph $G = (V, E)$.

(C1) $|N(S)| \geq |S|$ holds for every maximal stable set $S \subseteq V$.

(C2) $|N(S)| > |S|$ holds for every non-maximal stable set $S \subseteq V$.

Lemma 3.1 *If a graph $G = (V, E)$ is pseudo-hamiltonian, then it fulfills the conditions (C1) and (C2).*

Proof. Consider a pseudo-h-hamiltonian cycle C and let S be a stable set in G. Every vertex from S appears h times in C. Since S is stable, each vertex from S must be followed by a vertex from $N(S)$. Hence the set $N(S)$ is visited at least $h \cdot |S|$ times. Since each vertex from $N(S)$ also appears h times in C we obtain

$$|N(S)| \geq |S|. \tag{1}$$

Now assume that $|N(S)| = |S|$. Then vertices from S and from $N(S)$ must alternate in C, and it is not possible to visit any vertex from $V - S - N(S)$. This implies that $V = S \cup N(S)$, or equivalently, that S is a maximal stable set. □

Corollary 3.2 *If the graph $G = (V, E)$ with $|V| \geq 3$ vertices is pseudo-hamiltonian then the following holds:*

(a) G has no vertices of degree one.

(b) $\alpha(G) \leq \frac{1}{2}|V|$. □

We can use the results on regularizable graphs (cf. Section 2) in order to show that, for a connected graph, the conditions (C1) and (C2) are also sufficient for the existence of a pseudo-hamiltonian cycle.

Lemma 3.3 *If a connected graph $G = (V, E)$ fulfills conditions (C1) and (C2), then it is pseudo-hamiltonian.*

Proof. If $|N(S)| > |S|$ holds for every stable set $S \subseteq V$ then G is 2-bicritical and, by Proposition 2.1, also regularizable. Since G is connected, Lemma 2.2 implies that in this case G is pseudo-hamiltonian.

Otherwise, there exists a stable set S with $|N(S)| = |S|$. Then by condition (C1), S is maximal and $V = S \cup N(S)$ holds. Let H denote the spanning subgraph

of G which arises from deleting all edges between vertices from $N(S)$. We show that H is elementary bipartite. Then, again by Proposition 2.1, the subgraph H is regularizable and, since H is also connected, Lemma 2.2 implies that G is pseudo-hamiltonian.

By construction, the graph H is bipartite. H is connected, since otherwise we can easily find a proper subset $S' \subset S$ with $|N(S')| \leq |S'|$ in contradiction to the assumption. Let (s,t) be an arbitrary edge in H with $s \in S$. In $H - \{s,t\}$ we have $|N(S')| \geq |S'|$ for each set $S' \subseteq S - \{s\}$ (note that S' is not maximal stable in G). It is well known that this condition implies the existence of a perfect matching in $H - \{s,t\}$ (cf. e.g. Lovász and Plummer [13]). Hence there is a perfect matching in H containing the edge (s,t). □

Every hamiltonian graph must be 2-connected. However, it is easy to see that this is not a necessary condition for a graph to be pseudo-h-hamiltonian for some $h \geq 2$. On the other side one may ask whether there exists a number k such that every k-connected graph is also pseudo-hamiltonian. The following example shows that this is not true in general.

Example 3.4 *Consider the complete bipartite graph $K_{k+1,k}$, i.e. the graph consisting of two stable sets S and S' of cardinality $k+1$ and k, respectively, where any two vertices from S and S' are adjacent. By deleting fewer than k vertices, we leave at least one node in the stable set S and at least one node in the stable set S'. Hence, this graph is k-connected. However, since $|N(S)| = k < k+1 = |S|$, we conclude from Lemma 3.1 that the graph is not pseudo-hamiltonian.*

Chvátal [7] defines the *toughness* $t(G)$ of a graph G (where G is not a complete graph) by

$$t(G) = \min \frac{|W|}{c(G-W)}, \tag{2}$$

where W is a cutset of G and $c(G-W)$ denotes the number of connected components of the graph $G - W$. It is well known that a hamiltonian graph has toughness at least 1. As an extension of this result we obtain:

Lemma 3.5 *If G is pseudo-h-hamiltonian, then $t(G) \geq \frac{1}{h}$.*

Proof. Let W^* be a cutset of G with $t(G) = |W^*|/c(G-W^*)$. Each path between two vertices of different connected components of $G - W^*$ contains vertices from W^*. Hence, in a pseudo-h-hamiltonian cycle of G there appears at least $c(G-W^*)$ times a vertex from W^*, i.e. each vertex from W^* appears at least $c(G-W^*)/|W^*|$ times. This implies $h \geq 1/t(G)$ and the correctness of the claim. □

It is known (cf. Chvátal [7]) that there are graphs with toughness 1 which are not hamiltonian. Similarly, the converse of Lemma 3.5 is not always true for $h \geq 2$. The complete bipartite graph $K_{3,2}$ has toughness $t(K_{3,2}) = 2/3 \geq 1/h$. However, as argued in Example 3.4 above, this graph is not pseudo-h-hamiltonian.

Another sufficient condition for pseudo-hamiltonicity relies on the toughness of the graph.

Lemma 3.6 *(i) Any graph G with $t(G) \geq 1$ is pseudo-hamiltonian. (ii) For every $\varepsilon > 0$, there exists a graph G with $t(G) \geq 1 - \varepsilon$ that is not pseudo-hamiltonian.*

Proof. Consider a graph G with toughness at least 1. Clearly, G is connected. We will show that G fulfills the conditions (C1) and (C2), and then Lemma 3.3 implies statement (i).

Let S be a maximal stable set in G and assume that $|N(S)| < |S|$ holds. With $W := N(S)$, we obtain $c(G - W) > |W|$ as the vertices of S form the connected components of $G - W$. Hence $t(G) < 1$, in contradiction to the assumption.

Let S be a non-maximal stable set in G and assume that $|N(S)| \leq |S|$. Define again $W := N(S)$. Then the vertices of S are again connected components of $G - W$, and since S is not maximal there is at least one further component. Hence $c(G - W) > |W|$ holds, which implies that $t(G) < 1$.

In order to prove (ii), consider the complete bipartite graphs $K_{k+1,k}$ from Example 3.4: $K_{k+1,k}$ has toughness $k/(k+1)$. As k tends to infinity, this expression tends to one. □

4 Regular graphs

In this section, we discuss the problem of deciding whether a given d-regular graph possesses a pseudo-h-hamiltonian cycle. We will show that for every d, there is a precise threshold for h where the computational complexity of recognizing pseudo-h-hamiltonian d-regular graphs jumps from NP-complete to trivial.

Lemma 4.1 *(i) For odd $d \geq 3$, every connected d-regular graph G fulfills* $\mathrm{ph}(G) \leq d$. *(ii) For even $d \geq 4$, every connected d-regular graph G fulfills* $\mathrm{ph}(G) \leq d/2$.

Proof. For even d, graph G itself is Eulerian and the Eulerian cycle yields a pseudo-$d/2$-hamiltonian cycle. For odd d, the multigraph that contains two copies of every edge in G is Eulerian and thus yields a pseudo-d-hamiltonian cycle. □

Lemma 4.2 *(i) For odd $d \geq 3$, it is NP-complete to decide whether* $\mathrm{ph}(G) \leq d - 1$ *holds for a d-regular graph G. (ii) For even $d \geq 4$, it is NP-complete to decide whether* $\mathrm{ph}(G) \leq d/2 - 1$ *holds for a d-regular graph G.*

Proof. We only prove (i). The proof of (ii) can be done by analogous (somewhat tedious) arguments.

For every odd $d \geq 3$, the proof of (i) is based on the following auxiliary graph H_d: H_d has $2d - 1$ vertices that are divided into three parts X, Y and Z. Part X consists of a single vertex x, parts $Y = \{y_1, \ldots, y_{d-1}\}$ and $Z = \{z_1, \ldots, z_{d-1}\}$ both contain $d - 1$ vertices. There is an edge between x and every vertex in Y, and there is an edge between every vertex in Y and every vertex in Z. Moreover, the vertices in Z are connected to each other by a perfect matching in such a way that z_1 and z_2 are matched with each other. This completes the description of H_d. Note that in H_d, vertex x has degree $d - 1$ and all vertices in $Y \cup Z$ have degree

d. Moreover, we will use the following connected multigraph $M(H_d)$: $M(H_d)$ has the same vertex set as H_d. Vertex x is connected by a single edge to y_1 and y_2, respectively, and by two edges to each vertex in $Y - \{y_1, y_2\}$. For $1 \leq j \leq 2$, y_j is connected by $2d - 3$ edges to z_j, and for $3 \leq j \leq d - 1$, y_j is connected by $2d - 4$ edges to z_j. Finally, there is one edge that connects z_1 to z_2, and there are two copies of every other edge in the matching over Z. Note that in the resulting graph $M(H_d)$, vertex x has degree $2d - 4$ and all vertices in $Y \cup Z$ have degree $2d - 2$.

The NP-completeness proof for result (i) is done by a reduction from the NP-complete hamiltonian cycle problem in cubic graphs (cf. Garey and Johnson [11]). Consider an instance $G' = (V', E')$ of this problem, and construct a d-regular graph $G = (V, E)$ from G' as follows:

- For every $v \in V'$, introduce a corresponding vertex v^* in V. Moreover, introduce $d - 3$ pairwise disjoint copies of H_d. The x-vertex of every such copy is connected to v^*.
- For every edge $(u, v) \in E'$, introduce two new vertices $a_{u,v}$ and $a_{v,u}$ together with the three edges $(u^*, a_{u,v})$, $(a_{u,v}, a_{v,u})$ and $(a_{v,u}, v^*)$, i.e. the vertices $a_{u,v}$ and $a_{v,u}$ essentially subdivide the original edge (u, v) into three sub-edges.
- For every new vertex $a_{u,v}$, create $d - 2$ pairwise disjoint copies of H_d and connect the x-vertex of every copy to $a_{u,v}$.

It is easy to verify that the resulting graph G is d-regular (since in H_d, vertex x has degree $d - 1$ and all other vertices have degree d). We claim that G possesses a pseudo-$(d - 1)$-hamiltonian cycle if and only if G' possesses a hamiltonian cycle.

(If). Assume that G' possesses a hamiltonian cycle. Construct from this hamiltonian cycle a $(2d - 2)$-regular multigraph M^* as follows: For every copy of H_d in G, introduce the corresponding edges of $M(H_d)$ in M^*, together with two edges that connect the x-vertex to that vertex to which the copy has been attached. For every edge (u, v) that is used by the hamiltonian cycle, introduce the three edges $(u^*, a_{u,v})$, $(a_{u,v}, a_{v,u})$ and $(a_{v,u}, v^*)$ in M^*. For every edge (u, v) that is not used by the hamiltonian cycle, introduce two copies of $(u^*, a_{u,v})$ and two copies of $(a_{v,u}, v^*)$ in M^*. The resulting multigraph is $(2d - 2)$-regular, is connected (as it simulates the hamiltonian cycle in G'), and it is spanning. Hence, the corresponding Eulerian cycle in G yields a pseudo-$(d - 1)$-hamiltonian cycle for G.

(Only if). Now assume that G possesses a pseudo-$(d - 1)$-hamiltonian cycle C. Then the edges that are traversed by C form a $(2d-2)$-regular connected multigraph M^C. For every copy of H_d in G, the cycle C traverses the edge that connects the x-vertex to the vertex to which the copy has been attached, at least twice and an even number of times. Hence, for every edge $(u, v) \in E'$ the vertex $a_{v,u}$ in M^C is connected by at least $2d - 4$ edges to the x-vertices of the attached copies of H_d, and there remain only two edges that can connect $a_{v,u}$ to the rest of the graph. With this it is easy to verify that there remain only two possibilities how the cycle C may traverse the three edges $(u^*, a_{u,v})$, $(a_{u,v}, a_{v,u})$ and $(a_{v,u}, v^*)$ that correspond to some edge $(u, v) \in E'$ in the original graph: Either all three edges are traversed

thus resulting multigraph is 4-regular and contains only edges from G^2. Hence, G^2 is pseudo-2-hamiltonian.

(Only if). Now assume that G^2 possesses a pseudo-2-hamiltonian cycle C. The following statements on the structure of C are easy to verify.

1. C traverses every edge (b_v^i, v) with $v \in V_1'$ and $1 \leq i \leq 4$ exactly once.
2. C traverses every edge (b_v^i, a_v^i) with $v \in V_1'$ and $1 \leq i \leq 4$ exactly three times.
3. For every $v \in V_1'$, C either traverses exactly one or exactly zero of the edges (a_v^i, a_v^j) with $1 \leq i < j \leq 4$.
4. C traverses every edge (d_v^i, v) with $v \in V_2'$ and $1 \leq i \leq 2$ exactly once.
5. C traverses every edge (d_v^i, c_v^i) with $v \in V_2'$ and $1 \leq i \leq 2$ exactly three times.
6. C traverses every edge (c_v^1, c_v^2) with $v \in V_2'$ exactly once.

Hence, every $v \in V_1'$ is only connected to vertices b_v^i. Every $v \in V_2'$ must be connected by two edges to some vertices a_u^i. Hence, there are exactly $2|V_2'|$ edges between V_2' and the a_u^i with $u \in V_1'$, and a simple counting argument shows that in statement (3) above, the "traverses exactly zero of the edges"-part can never hold. Hence, for every $v \in V_1'$ there exist exactly two edges in C that connect some a_v^i to some vertex $u \in V_2'$. It is straightforward to see that the union of all these edges corresponds to a hamiltonian cycle in G'. □

6 Special graph classes

In this section, we show that deciding whether a graph is pseudo-h-hamiltonian is NP-complete even for some very restricted classes of graphs that possess a strong combinatorial structure. Moreover, we present polynomial time algorithms for other classes of structured graphs.

6.1 Trees and planar graphs

By Corollary 3.2.(a), a pseudo-hamiltonian graph cannot have any vertices of degree one. Hence, $\mathrm{ph}(T) = \infty$ for any tree T.

If we start the construction in the proof of Theorem 2.6 with a planar graph G', then the constructed graph G is also planar. Since deciding hamiltonicity of planar graphs is NP-complete [11], we conclude that for every $h \geq 1$ it is NP-complete to decide whether a planar graph is pseudo-h-hamiltonian.

6.2 Partial k-trees

The class of *partial k-trees* is a well-known generalization of ordinary trees (see e.g. the survey articles by Bodlaender [4, 5, 6] and by van Leeuwen [14]). It is known that series-parallel graphs and outerplanar graphs are partial 2-trees and that Halin graphs are partial 3-trees. Large classes of algorithmic problems can be solved in polynomial time on partial k-trees if k is constant. Essentially, each graph problem

that is expressible in the Monadic Second Order Logic (MSOL) is solvable in linear time on partial k-trees with constant k (cf. e.g. Arnborg, Lagergren, Seese [1]).

Lemma 6.1 *For every $h \geq 1$ and for every $k \geq 1$, it can be decided in linear time whether a given partial k-tree is pseudo-h-hamiltonian.*

Proof. We only show the statement for $h = 2$; the other cases can be settled analogously. For a given graph $G = (V, E)$, the property of having a connected 4-regular submultigraph can be expressed in MSOL as follows:

1. There exist three pairwise disjoint subsets E_1, E_2 and E_3 of E
2. Every vertex is either incident to (i) four edges from E_1, or to (ii) two edges from E_1 and one edge from E_2, or to (iii) one edge from E_1 and one edge from E_3, or to (iv) two edges from E_2
3. There does not exist a partition of the vertex set V into two non-empty sets V_1 and V_2, such that none of the edges in $E_1 \cup E_2 \cup E_3$ connects V_1 to V_2.

Intuitively speaking, the edges in E_1 (E_2, E_3) occur once (twice, thrice) in the submultigraph. The second condition then takes care of the 4-regularity, and the third condition ensures that the submultigraph is connected. □

6.3 Bipartite graphs and split graphs

Lemma 6.2 *For every integer $h \geq 1$, it is NP-complete to decide whether a bipartite graph is pseudo-h-hamiltonian.*

Proof. It is NP-complete to decide whether a bipartite graph G' is hamiltonian (cf. Garey and Johnson [11]). Consider a bipartite graph $G' = (V', E')$ with bipartition $V' = V_1' \cup V_2'$, and construct from G' another bipartite graph G as follows. For every vertex $v \in V'$, introduce two vertices ℓ_v and r_v in V together with auxiliary vertices a_v^i and b_v^i, $i = 1, \ldots, 2h-2$. In E, there are the edges (ℓ_v, r_v) together with the edges (ℓ_v, a_v^i), (a_v^i, b_v^i), and (b_v^i, r_v) for $i = 1, \ldots, 2h-2$. Moreover, for every edge $(u, v) \in E'$ with $u \in V_1'$ and $v \in V_2'$, we introduce the two edges (ℓ_u, r_v) and (ℓ_v, r_u).

It can be verified that the resulting graph G is also bipartite. Moreover, one can show that G possesses a pseudo-h-hamiltonian cycle if and only if G' possesses a hamiltonian cycle. □

A *split graph* is a graph whose vertex set can be partitioned into two parts such that the subgraph induced by the first part is a clique and the subgraph induced by the second part is a stable set.

Corollary 6.3 *For every integer $h \geq 1$, it is NP-complete to decide whether a split graph is pseudo-h-hamiltonian.*

Proof. In the NP-completeness proof for bipartite graphs in Lemma 6.2, both classes in the bipartition of the constructed graph G are of equal cardinality. Transform G into a split graph G^* by adding all edges between vertices in one part of

the bipartition. It is easy to see that a pseudo-h-hamiltonian cycle in G^* can never use these added edges, and hence G^* is pseudo-h-hamiltonian if and only if G' is hamiltonian. □

6.4 Cocomparability graphs

A *comparability graph* is a graph $G = (V, E)$ whose edges are exactly the comparable pairs in a partial order on V. The complementary graph is called a *cocomparability graph*. The class of cocomparability graphs properly contains all cographs, permutation graphs and interval graphs.

Lemma 6.4 *For every integer $h \geq 1$, it can be decided in polynomial time whether a cocomparability graph is pseudo-h-hamiltonian.*

Proof. It is known that a hamiltonian cycle in a cocomparability graph can be found in polynomial time (cf. Deogun and Steiner [9]). Given a cocomparability graph $G = (V, E)$, we construct another cocomparability graph $G' = (V', E')$ as follows. V' contains the vertices in V together with $(h-1)|V|$ new vertices. For every vertex $v \in V$ there are $h-1$ new vertices that are called v^i, where $i = 2, \ldots, h$. For simplicity of notation, let $v^1 := v$. If (u, v) is an edge in E then all edges (u^i, v^j) with $i, j = 1, \ldots, h$ belong to E' (roughly spoken, G' arises from G by replacing each vertex by a stable set of h vertices). It is easy to see that G' is again a cocomparability graph. We show that G has a pseudo-h-hamiltonian cycle if and only if G' has a hamiltonian cycle.

(If). Assume that G' possesses a hamiltonian cycle. We obtain a pseudo-h-hamiltonian cycle in G if each vertex v^i, $i = 2, \ldots, h$, is replaced by the corresponding vertex v.

(Only if). Now assume that G possesses a pseudo-h-hamiltonian cycle C. Each vertex of G appears h times in C. For each $v \in V$ replace $h-1$ copies of v in C by $v^2, \ldots, v^h$. This yields a *2-factor* of G', i.e. a subgraph of G' such that each vertex has degree 2. If the 2-factor is a cycle then we have a hamiltonian cycle in G' and we are done. Otherwise the 2-factor is a disjoint union of cycles. In this case the following principle allows to reduce the number of cycles: Let C_1 and C_2 denote two disjoint cycles such that v^i belongs to C_1 and v^j belongs to C_2 (it is straightforward to see that such cycles must exist). Let further x be the predecessor of v^i in C_1 and y the predecessor of v^j in C_2. Replace the edges (x, v^i) and (y, v^j) by (x, v^j) and (y, v^i). One obtains a new cycle that contains all vertices from C_1 and C_2. Repeatedly merging cycles in this way finally provides the desired hamiltonian cycle in G'. □

We leave it as an open problem to determine the complexity of computing the pseudo-hamiltonicity number of *asteroidal triple-free* graphs, AT-free graphs for short (cf. Corneil, Olariu, and Stewart [8]). Note that for an AT-free graph G, the graph G' that is constructed in the proof of Lemma 6.4 above is also AT-free. However, the complexity of finding a hamiltonian cycle in AT-free graphs is currently unknown.

References

[1] S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms 12*, 1991, 308–340.

[2] C. Berge. Regularizable graphs I. *Discrete Mathematics 23*, 1978, 85–89.

[3] C. Berge. Regularizable graphs II. *Discrete Mathematics 23*, 1978, 91–95.

[4] H.L. Bodlaender. Some classes of graphs with bounded treewidth. *Bulletin of the EATCS 36*, 1988, 116–126.

[5] H.L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica 11*, 1993, 1–21.

[6] H.L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science 209*, 1998, 1–45.

[7] V. Chvátal. Tough graphs and hamiltonian circuits. *Discrete Mathematics 5*, 1973, 215–228.

[8] D.G. Corneil, S. Olariu, and L. Stewart. Asteroidal triple-free graphs. *Proceedings of the 19th International Workshop on Graph-Theoretic Concepts in Computer Science WG'93*, Springer Verlag, LNCS 790, 1994, 211–224.

[9] J.S. Deogun and G. Steiner. Polynomial algorithms for hamiltonian cycles in cocomparability graphs. *SIAM Journal on Computing 23*, 1994, 520–552.

[10] H. Fleischner. The square of every two-connected graph is hamiltonian. *Journal of Combinatorial Theory B 16*, 1974, 29–34.

[11] M.R. Garey and D.S. Johnson. *Computers and Intractability, A guide to the theory of NP-completeness.* Freeman, San Francisco, 1979.

[12] D.S. Johnson. The NP-Completeness Column: an Ongoing Guide. *Journal of Algorithms 6*, 1985, 434–451.

[13] L. Lovász and M.D. Plummer. Matching Theory. *Annals of Discrete Mathematics 29*, North-Holland, 1986.

[14] J. van Leeuwen. Graph algorithms. in *Handbook of Theoretical Computer Science, A: Algorithms and Complexity Theory*, 527–631, North Holland, Amsterdam, 1990.

Received April, 2000

Acta Cybernetica 14, (2000) 569-582.

Two remarks on variants of simple eco-grammar systems*

Judit Csima †

Abstract

Two powerful variants of simple eco-grammar systems, namely extended tabled simple eco-grammar systems (ETEG systems) and weak extended simple eco-grammar systems (wEEG systems) are studied. It is proved that both modifications of the original definition result in universal power: all recursively enumerable languages can be obtained both by ETEG and by wEEG systems.

1 Introduction

Eco-grammar systems form a grammatical framework proposed in [2] for modelling living systems consisting of several agents and a common environment. In the original definition of an eco-grammar system, the environment is described by a Lindenmayer system which determines its evolution; the agents are represented by context-free grammars and by Lindenmayer systems : the Lindenmayer systems determine their development, while the context-free grammars describe their actions. The interaction between the agents and the environment is ensured by the computable functions ψ and ϕ, which allow the agents to adapt to the environment both in their development and in their actions.

In the original model there are no terminal and nonterminal symbols. In [2] it was shown that this model is very strong as far as the generative capacity is concerned: all recursively enumerable languages can be obtained as languages of extended eco-grammar systems (that is with systems with a distinguished terminal alphabet) with a very simple choice of the functions ψ and ϕ.

Because of this result another, simpler variant of eco-grammar systems was introduced in [2]: the simple eco-grammar system. In a simple eco-grammar system the interaction between the environment and the agents is restricted and the agents do not have an inner representation. Other variants like non-extended simple eco-grammar systems in [3] and conditional tabled eco-grammar systems in [4], [5] and [10] were introduced and studied.

*Research supported by the Hungarian Scientific Foundation "OTKA" Grant No. T029615

†Computer and Automation Research Institute, Hungarian Academy of Sciences, Kende u. 13-17, H-1111, Budapest, Hungary. E-mail: csima@cs.bme.hu

Besides these directions, the study of extended simple eco-grammar systems continued in [6], where it was proved that the hierarchy according to the number of the agents is a collapsing one and that the class of languages generated by extended simple eco-grammar systems without λ-rules is between the class of languages generated by extended $0L$ systems and the class of languages generated by matrix grammars with appearance checking. (For more information about these language classes the reader is referred to [9] and [8].) The general case, where λ-rules are allowed in the system, remained open. In [1] it was shown that the hierarchy collapses in the general case as well, even if we consider different derivation modes, but the place of this collapsing language class remained unsolved.

In [11] simple eco-grammar systems with prescribed teams were examined. It was proved there that extended tabled simple eco-grammar systems with teams of agents with prescribed members and operating according to a weak rewriting steps (that is the derivation is not blocked if some agents from a team cannot perform any action) can generate all recursively enumerable languages. In this article we present a stronger result: it is not necessary to use prescribed teams to reach the power of Turing machines. Moreover, both extended tabled simple eco-grammar systems (using the original derivation mode, not the weak one) and weak extended simple eco-grammar systems (without tables) are enough to generate all recursively enumerable languages. (We note that the definition of a weak derivation step we use in this article is a slightly different one compared to [11], therefore only the first result is stronger than the one in [11].)

More precisely, we consider two variants of extended simple eco-grammar systems for which the question of their generative power will be answered.

The first part of the article deals with extended tabled simple eco-grammar systems, where instead of one $0L$ system the environment can be represented by more than one $0L$ system, called tables. Thus the environment can vary its behaviour step by step. Allowing this possibility, all recursively enumerable languages can be obtained even with one agent.

In the second part of the article we present weak extended simple eco-grammar systems. In this case the definition of the derivation step is different from the original definition of an extended simple eco-grammar system. In this modified version the derivation is not blocked if some agents cannot perform any action on the sentential form. We show that the generative power of Turing machines can be reached also in this case.

2 Preliminaries

Here we present the notions and notations used in this article, for further information the reader is referred to [9], [8] and [7].

The set of all non-empty words over a finite alphabet V is denoted by V^+, the empty word is denoted by λ; $V^* = V^+ \cup \{\lambda\}$. For a set V, we denote by *card* (V) the cardinality of V. For a word x, we denote by $|x|$ the length of x. If $x = x_1 \cdots x_n$ is a word over an alphabet V, $x_j \in V$ for $1 \leq j \leq n$, $[x, i, k]$ denotes the word

$[x_1, i, k] \cdots [x_n, i, k]$ for $1 \leq i, k$.

By a context-free production or by a context-free rule (a CF rule, for short) over an alphabet V we mean a production of the form of $a \to u$, where $a \in V$ and $u \in V^*$. A CF rule is a λ-rule (or a deletion rule) if $u = \lambda$.

We also use the following notations: for a set of CF rules R, $dom(R)$ denotes the set of all letters appearing in the left-hand side of a rule in R. For a word x over an alphabet V, $alph(x)$ denotes the set of all letters appearing in x.

A $0L$ system is a triplet $H = (V, P, \omega)$, where V is a finite alphabet, P is a set of context-free rules over V, and $\omega \in V^*$ is the axiom. Moreover, P has to be complete, that is for each symbol a from V there must be at least one rule in P with this letter in the left-hand side. $0L$ systems use parallel derivations: we say that x directly derives y in a $0L$ system $H = (V, P, \omega)$, written as $x \overset{0L}{\Longrightarrow}_H y$, if $x = x_1 x_2 \cdots x_n$, $y = y_1 y_2 \cdots y_n$, where $x_i \in V$, $y_i \in V^*$, and the rules $x_i \to y_i$ are in P for $1 \leq i \leq n$.

A $T0L$ system is a triplet $H = (V, T, \omega)$, where V is a finite alphabet, $T = \{T_1, \ldots, T_k\}$ is a set of tables over V, where each table T_i for $1 \leq i \leq k$ is a complete set of CF rules over V, and $\omega \in V^*$ is the axiom. We say that x directly derives y in a $T0L$ system $H = (V, T, \omega)$, written as $x \overset{T0L}{\Longrightarrow}_H y$, if $x \overset{0L}{\Longrightarrow}_{H_i} y$ for some i , $1 \leq i \leq k$, with the $0L$ system $H_i = (V, T_i, \omega)$.

An $ET0L$ system is a quadruple $H = (V, T, \Delta, \omega)$, where $H' = (V, T, \omega)$ is a $T0L$ system, and $\Delta \subseteq V$ is the terminal alphabet. In an $ET0L$ system $H = (V, T, \Delta, \omega)$ x directly derives y, written as $x \overset{ET0L}{\Longrightarrow}_H y$, if $x \overset{T0L}{\Longrightarrow}_{H'} y$.

The transitive and reflexive closure of $\overset{ET0L}{\Longrightarrow}_H$ is denoted by $\overset{ET0L}{\Longrightarrow}{}^*_H$.

The generated language of the $ET0L$ system H (denoted by $L(H)$) is

$$L(H) = \{\, w \in \Delta^* \mid \omega \overset{ET0L}{\Longrightarrow}{}^*_H w \,\}.$$

That is, in an $ET0L$ system only words over a distinguished subalphabet are in the generated language. A language is said to be an $ET0L$ language if there is an $ET0L$ system which generates it.

$T0L$ and $0L$ systems are special cases of $ET0L$ systems: $\Delta = V$ stands in both cases; moreover, in the case of $0L$ systems $T = \{T_1\}$ also holds. Therefore the above definition gives the generated language for these systems as well.

A random-context grammar is a quadruple $G = (N, T, P, S)$ where N is the set of nonterminals, T is the set of terminals, S is the axiom, and P is a finite set of random-context rules, that is triplets of the form of $(C \to \alpha, Q, R)$, where $C \to \alpha$ is a CF rule over $N \cup T$, where $C \in N$, and Q and R are subsets of N. For $x, y \in (N \cup T)^*$, we write $x \overset{rc}{\Longrightarrow} y$ iff $x = x_1 C x_2$, $y = x_1 \alpha x_2$ for some $x_1, x_2 \in (N \cup T)^*$, $(C \to \alpha, Q, R)$ is a triplet in P, all symbols of Q appear and no symbol of R appears in $x_1 C x_2$ (Q is called the permitting context, and R is called the forbidding context of the rule $C \to \alpha$. If Q and/or R are empty, no check is necessary.) This is a slightly modified but equivalent version of the definition presented in [7].

If the forbidding context is empty for every rule, we speak about a random-context grammar without appearance checking, otherwise the grammar is with appearance checking. For the sake of brevity we will refer to a random-context

grammar with appearance checking and with λ-rules as a random-context grammar.

The generated language of a random-context grammar consists of all the words which can be generated in some steps from axiom S. The class of languages which can be generated by random-context grammars is denoted by $\mathcal{RC}_{ac}^{\lambda}$.

Now we present the definition of an extended simple eco-grammar system, as introduced in [6].

Definition 2.1 *An extended simple eco-grammar system is a construct* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$, *where*

- V_E *is a finite, non-empty alphabet,*
- P_E *is a complete set of CF rules over* V_E, *i.e. for each letter of* V_E *there exists at least one rule in* P_E *with this letter in the left-hand side,*
- R_i *is a non-empty set of CF rules over* V_E *for* $1 \leq i \leq n$,
- $\omega \in V_E{}^*$, *and*
- $\Delta \subseteq V_E$.

In this construct V_E is the alphabet and P_E is the set of the evolution rules of the environment. The ith agent is represented by R_i, $1 \leq i \leq n$, its set of action rules. The current state of the environment, which is also the state of the eco-grammar system, is the current sentential form. String ω is the initial state. Δ is the terminal alphabet and shortly we will see that only words over Δ are in the generated language of Σ.

The system changes its state by a simultaneous action of the agents and by a parallel rewriting according to P_E.

Definition 2.2 *Consider an extended simple eco-grammar system* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$. *We say that* x *directly derives* y *in* Σ *(with* $x \in V_E{}^+$ *and* $y \in V_E{}^*$, *written as* $x \overset{eco}{\Longrightarrow}_\Sigma y$), *if*

- $x = x_1 Z_1 x_2 Z_2 \cdots x_n Z_n x_{n+1}$, *with* $Z_i \in V_E$, $x_j \in V_E{}^*$, $1 \leq i \leq n$, *and* $1 \leq j \leq n+1$,
- $y = y_1 w_1 y_2 w_2 \cdots y_n w_n y_{n+1}$, *with* $y_i, w_j \in V_E{}^*$, $1 \leq i \leq n$, *and* $1 \leq j \leq n+1$,
- *there exists a permutation of the agents, namely* $R_{j_1}, R_{j_2}, \ldots, R_{j_n}$, *such that* $Z_i \to w_i \in R_{j_i}$, *for* $1 \leq i \leq n$, *and*
- $x_i = \lambda$ *or* $x_i \overset{0L}{\Longrightarrow}_E y_i$, *for* $1 \leq i \leq n$, *where* $E = (V_E, P_E, \omega)$ *is the* $0L$ *system of the environment.*

We denote the transitive and reflexive closure of $\overset{eco}{\Longrightarrow}_\Sigma$ by $\overset{eco}{\Longrightarrow}{}^*_\Sigma$.
The generated language consists of the words over Δ which can be obtained in some derivation steps starting from the axiom.

Definition 2.3 *Consider an extended simple eco-grammar system* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$. *The generated language of* Σ *is the following:*

$$L(\Sigma) = \{ v \in \Delta^* \mid \omega \overset{eco}{\Longrightarrow}{}^*_\Sigma v \}.$$

3 Extended tabled simple eco-grammar systems

In this section a modified version of an extended simple eco-grammar system, called an extended tabled simple eco-grammar system, is investigated. In such a system the environment can be represented by more than one $0L$ system. We show that this device is as powerful as Turing machines.

Definition 3.1 *An extended tabled simple eco-grammar system (an ETEG system, for short) is a construct* $\Sigma = (V_E, T_E, R_1, \ldots, R_n, \omega, \Delta)$, *where*

- $(V_E, T_E, \Delta, \omega)$ *is an* $ET0L$ *system, and*
- R_i *is a non-empty set of* CF *rules over* V_E *for* $1 \leq i \leq n$.

Here V_E, R_i, ω and Δ have the same meaning as in Definition 2.1, namely they are the alphabet of the system, the production sets representing the agents, the axiom, and the terminal alphabet. T_E is the set of tables of the environment.

In an extended tabled eco-grammar system the environment can choose a $0L$ system in each derivation step to perform a parallel rewriting.

Definition 3.2 *Consider an extended tabled simple eco-grammar system* $\Sigma = (V_E, T_E, R_1, \ldots, R_n, \omega, \Delta)$. *We say that* x *directly derives* y *in* Σ *(with* $x \in V_E{}^+$ *and* $y \in V_E{}^*$, *written as* $x \overset{t-eco}{\Longrightarrow}_\Sigma y$*) if* $x \overset{eco}{\Longrightarrow}_{\Sigma_i} y$ *for the extended simple eco-grammar system* $\Sigma_i = (V_E, T_i, R_1, \ldots, R_n, \omega, \Delta)$ *for some* $1 \leq i \leq k$.

We denote the transitive and reflexive closure of $\overset{t-eco}{\Longrightarrow}_\Sigma$ by $\overset{t-eco}{\Longrightarrow}{}^*_\Sigma$.

Definition 3.3 *The generated language of an extended tabled simple eco-grammar system* $\Sigma = (V_E, T_E, R_1, \ldots, R_n, \omega, \Delta)$ *is the following:*

$$L(\Sigma) = \{ v \in \Delta^* \mid \omega \overset{t-eco}{\Longrightarrow}{}^*_\Sigma v \}.$$

The class of languages which can be generated by an $ETEG$ system with n agents is denoted by $\mathcal{L}(\mathcal{ETEG}, \backslash)$.

Now we present an example to illustrate the power of $ETEG$ systems.

Example 3.1 *Let* $\Sigma = (V_E, T_E, R_1, \omega, \Delta)$ *be the following* $ETEG$ *system:*

- $V_E = \{ S, B, N, D, S', a, b \}$,
- $T_E = \{ T_1, T_2, T_3, T_4 \}$, *where*
  $T_1 = \{ S \to D, N \to N, a \to a, b \to b, D \to D, S' \to S', B \to B \}$,
  $T_2 = \{ S \to \lambda, N \to N, a \to a, b \to b, D \to D, S' \to D, B \to B \}$,
  $T_3 = \{ S \to D, N \to N, a \to a, b \to b, D \to D, S' \to D, B \to bB \}$,
  $T_4 = \{ S \to D, N \to N, a \to a, b \to b, D \to D, S' \to D, B \to b \}$,
- $R_1 = \{ S \to aBNS, N \to \lambda, S' \to \lambda \}$,
- $\omega = SS'$, *and*

- $\Delta = \{ a, b \}$.

We show that the generated language of this system is

$$L(\Sigma) = \{ (ab^n)^m \mid 1 \leq n \leq m \} \cup \{\lambda\}.$$

First we note the following: S can be deleted only by the environment and S' can be deleted only by the agent. These two events must happen in the same derivation step because of the following reasons. If the environment deletes S, that is uses table T_2, and the agent does not apply the rule $S' \to \lambda$, then the environment rewrites S' to D, which "blocks" the derivation since this D will never disappear from the sentential form. If the agent deletes S' in a derivation step and the environment does not delete S, that is does not use table T_2, it introduces symbol D again.

We have seen that S and S' disappear from the sentential form in the same derivation step. Before this step the agent has to use the rule $S \to aBNS$, or otherwise the environment introduces a D; the environment has to use the first table T_1, or otherwise S' would be rewritten to D. Thus either the derivation is $SS' \overset{t-eco}{\Longrightarrow} \lambda$, or the first few steps are $SS' \overset{t-eco}{\Longrightarrow}{}^* (aBN)^m SS' \overset{t-eco}{\Longrightarrow} (aBN)^m$. After these steps the only possibility for the agent to work is by using rule $N \to \lambda$. During these steps the environment can use any of its tables, therefore it can introduce b letters before all the B's or it can rewrite either all B's to B or all B's to b's. Because there are only m symbols N in the sentential form, the derivation lasts exactly m more steps. Hence at the end of the derivation, when there are no more N symbols left, there are at least one but at most m symbols b after each a.

The above explication showes that all non-empty generated words are of the form of $(ab^n)^m$, $1 \leq n \leq m$. It follows from the construction that all words of this form as well as the empty word, λ, are in the generated language, which is thus indeed:

$$L(\Sigma) = \{ (ab^n)^m \mid 1 \leq n \leq m \} \cup \{\lambda\}.$$

This example shows that even a very simple extended tabled simple eco-grammar system with only one agent is able to produce a quite complicated language, namely a language which is not an $ET0L$ one (see [8]).

We show that the generative capacity of these systems reaches that of Turing machines. This is a direct consequence of the following lemma.

Lemma 3.1 $\mathcal{RC}_{ac}^{\lambda} \subseteq \mathcal{L}(\mathcal{ETEG}, \infty)$

Proof Let $G = (N, T, S, P)$ be a random-context grammar. Without loss of generality, we can asssume that the rules in P have the form $(C \to \alpha, Q, R)$, with $C \in N$, $\alpha \in (N \cup T)^*$, $card(Q) \leq 1$, and $card(R) \leq 1$ (see [7]). Moreover, we can assume that there are no rules with $Q = R$, $R = \{C\}$ or $Q = \{C\}$, because rules of the first two types are not applicable and in the last case the rule is equivalent to the rule $(C \to x, \emptyset, R)$.

We denote by r the number of rules in P and by V the set $N \cup T$. The rules of P are enumerated as $p_i = (C_i \to \alpha_i, Q_i, R_i)$. We will refer to the components of the ith rule as C_i, α_i, Q_i, and R_i.

Now we will construct a simple extended tabled eco-grammar system $\Sigma = (V_E, T_E, R_1, \omega, \Delta)$ such that $L(\Sigma) = L(G)$.

Let

- $V_E = V \cup \{[X,i] \mid X \in V, 1 \le i \le r\}$
  $\cup \{[X',i] \mid X \in V, 1 \le i \le r\}$
  $\cup \{D, U, Z, Z'\}$
  $\cup \{[U,i] \mid 1 \le i \le r, Q_i = \emptyset\}$
  $\cup \{[U',i] \mid 1 \le i \le r\} \cup$
  $\cup \{[\widehat{U},i] \mid 1 \le i \le r, Q_i \ne \emptyset\}$,
  where $D, U, Z \notin V$,

- $T_E = \{T_{(i)}, T'_{(i)}, T''_{(i)} \mid 1 \le i \le r\}$, where

  $T_{(i)} = \{X \to [X,i] \mid X \in V\}$
  $\cup \{U \to [U,i] \mid Q_i = \emptyset\}$
  $\cup \{U \to [\widehat{U},i] \mid Q_i \ne \emptyset\}$,
  for $1 \le i \le r$,

  $T'_{(i)} = \{[X,i] \to [X',i] \mid X \in V, \{X\} \ne R_i\}$
  $\cup \{[B,i] \to D \mid \{B\} = R_i\}$
  $\cup \{Z' \to Z'\}$
  $\cup \{[\widehat{U},i] \to [U',i] \mid Q_i \ne \emptyset\}$
  for $1 \le i \le r$,

  $T''_{(i)} = \{[X',i] \to X \mid X \in V\}$
  $\cup \{Z' \to Z, Z' \to \lambda, [U',i] \to U, [U',i] \to \lambda\}$
  for $1 \le i \le r$,

- $R_1 = \{Z \to Z'\}$
  $\cup \{[U,i] \to [U',i] \mid 1 \le i \le r, Q_i = \emptyset\}$
  $\cup \{[A,i] \to [A',i] \mid 1 \le i \le r, Q_i = \{A\}\}$
  $\cup \{[C'_i,i] \to \alpha_i \mid 1 \le i \le r\}$,

- $\omega = SUZ$, and

- $\Delta = T$.

Those symbols which are not mentioned above in the tables are rewritten into D; these rules make the tables complete.

We introduce different alphabets according to the rules of P in the following way. These alphabets are multiplied versions of V: for the ith rule of P we have alphabets $\{[X,i] \mid X \in V\}$ and $\{[X',i] \mid X \in V\}$. Moreover, we have some special additional

symbols in the ETEG system in order to coordinate the derivation. These are $\{D, U, Z, Z'\} \cup \{[U', i] \mid 1 \leq i \leq r\} \cup \{[U, i] \mid 1 \leq i \leq r, Q_i = \emptyset\} \cup \{[\widehat{U}, i] \mid 1 \leq i \leq r, Q_i \neq \emptyset\}$. By D the derivation is "blocked": if this symbol appears, the derivation never results in a terminal word. Symbol U allows the agent to work when $Q_i = \emptyset$.

First we show how a derivation step of G can be simulated by Σ. During the simulation the sentential form has the form wUZ, where the word w corresponds to the sentential form of G, while U and Z coordinate the simulation. Let us suppose that in a derivation step with sentential form x rule $(C_i \to \alpha_i, Q_i, R_i)$ is used. The simulation in the ETEG system is as follows.

In the first step the environment applies table $T_{(i)}$ to rewrite xU into $[x, i][U, i]$ or into $[x, i][\widehat{U}, i]$ depending on whether or not $Q_i = \emptyset$; the agent rewrites Z into Z'. This is the only role of Z: it allows the agent to work during the first step of the simulation.

In the second step the agent applies the rule $[U, i] \to [U', i]$ if $Q_i = \emptyset$ or applies the rule $[A, i] \to [A', i]$ if $Q_i = \{A\}$. The environment rewrites the remaining letters by using table $T'_{(i)}$.

In the third simulation step the agent applies its rule corresponding to the rule of G, namely rule $[C'_i, i] \to \alpha_i$, while the environment rewrites the remaining letters by using table $T''_{(i)}$. During this last step, the environment can delete the special symbols Z' and $[U', i]$, thus allowing the possibility of finishing the derivation if the sentential form would be a terminal word.

Now we have showed that we can simulate the derivation steps of the random-context grammar. It follows from the construction of the simulating $ETEG$ system that the behaviour described above is the only one which can result in a terminal word. The only possibility to start a derivation from a word over $V \cup \{U, Z\}$ is to use one of the tables $T_{(i)}$ and the rule $Z \to Z'$ of the agent. If the sentential form contains some forbidding letters from R_i, the environment blocks the derivation in the next step by introducing a D; if the permitting symbol referring to the non-empty set Q_i does not appear in the sentential form, the derivation is blocked because the agent cannot work. (It cannot use the other rule $[U, i] \to [U', i]$, because this symbol appears in the sentential form iff $Q_i = \emptyset$.) In the next step the agent has to use the rule $[C'_i, i] \to \alpha_i$ and the environment has to use table $T''_{(i)}$. These three consecutive steps simulate the application of one of the rules of P. □

Using the fact that $\mathcal{RC}^{\lambda}_{ac} = RE$ and the fact that we can construct a Turing machine simulating an extended tabled simple eco-grammar system, we obtain the following theorem:

Theorem 3.1 $\mathcal{L}(\mathcal{ETEG}, \infty) = \mathcal{RE}$

4 Weak extended simple eco-grammar systems

In this section another variant of extended simple eco-grammar systems is studied: the weak extended simple eco-grammar system. This variant has the same compo-

nents as the extended simple eco-grammar system but it works in a different way. Informally speaking, in a weak system the derivation is not blocked if there are some agents which cannot perform any action.

Compared to [11], the definition of a weak rewriting step is slightly different. There, in a weak rewriting step, the derivation is blocked if no agent can work, whereas here we allow this possibility.

Definition 4.1 *A weak extended simple eco-grammar system (a wEEG system, for short) is a construct* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$, *where*

- *all the components are the same as in Definition 2.1.*

In a weak extended simple eco-grammar system $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ x *directly derives* y *(with* $x \in V_E{}^+$ *and* $y \in V_E{}^*$, *written as* $x \overset{w-eco}{\Longrightarrow}_\Sigma y$*) if*

- $x = x_1 Z_1 x_2 Z_2 \cdots x_k Z_k x_{k+1}$, *with* $Z_i \in V_E$, $x_j \in V_E{}^*$, $0 \leq k \leq n$, $1 \leq i \leq k$, *and* $1 \leq j \leq k+1$,
- $y = y_1 w_1 y_2 w_2 \cdots y_k w_k y_{k+1}$, *with* $y_i, w_j \in V_E{}^*$, $0 \leq k \leq n$, $1 \leq i \leq k$, *and* $1 \leq j \leq k+1$,
- *there exists a permutation of some agents, namely* $R_{j_1}, R_{j_2}, \ldots, R_{j_k}$, *such that* $Z_i \rightarrow w_i \in R_{j_i}$, *for* $1 \leq i \leq k$,
- $\{dom(R_t) \mid 1 \leq t \leq n, t \neq j_i, \text{ for } 1 \leq i \leq k\} \cap alph(x_1 x_2 \cdots x_{k+1}) = \emptyset$, *and*
- $x_i = \lambda$ *or* $x_i \overset{0L}{\Longrightarrow}_E y_i$, *for* $1 \leq i \leq k+1$, *where* $E = (V_E, P_E, \omega)$ *is the* $0L$ *system of the environment.*

We denote by $\overset{w-eco}{\Longrightarrow}{}^*_\Sigma$ the transitive and reflexive closure of $\overset{w-eco}{\Longrightarrow}_\Sigma$.

That is, in a weak extended simple eco-grammar system we choose some agents to perform a common action in the following way: the chosen agents can perform an action together and there is no symbol among the remaining letters where any of the other agents could act. The chosen agents perform their actions and the remaining letters are rewritten by the environment. In the particular case when there is only one agent in the system, this definition implies that the agent has to work if it is able to but if no letter can be rewritten by the agent it is the environment itself that continues the derivation.

Definition 4.2 *For a weak extended simple eco-grammar system* $\Sigma = (V_E, P_E, R_1, \ldots, R_n, \omega, \Delta)$ *the generated language is the following:*

$$L(\Sigma) = \{ v \in \Delta^* \mid \omega \overset{w-eco}{\Longrightarrow}{}^*_\Sigma v \}.$$

We denote by $wEEG(n)$ the class of languages which can be generated by weak extended simple eco-grammar systems with n agents.

In the following we show that $wEEG$ systems can generate all recursively enumerable languages. The result is based on the following lemma.

Lemma 4.1 $\mathcal{RC}^{\lambda}_{ac} \subseteq wEEG(1)$

Proof For a random-context grammar $G = (N, T, P, S)$ we will give a weak extended simple eco-grammar system $\Sigma = (V_E, P_E, R_1, \Delta, \omega)$ such that $L(G) = L(\Sigma)$.

First we present the definition of this system Σ and explain its functioning. Similar to Lemma 3.1, the notation V stands for $N \cup T$, r denotes the number of rules in P, the rules in P are enumerated as $p_i = (C_i \to \alpha_i, Q_i, R_i)$, we assume there are no rules with $Q = R$, $Q = \{C\}$ or $R = \{C\}$, and we refer to the components of the ith rule as C_i, α_i, Q_i, and R_i.

Let

- $$\begin{aligned} V_E = V \cup\ & \{[X,i,j] \mid X \in V, 1 \le i \le r, 1 \le j \le 5\} \\ & \cup \{[Z,i,j], [\widehat{Z},i,k] \mid 1 \le i \le r, 1 \le j \le 5, 1 \le k \le 4\} \\ & \cup \{[\widehat{C}_i,i,k] \mid 1 \le i \le r, 1 \le k \le 4\} \\ & \cup \{[U,i,j] \mid 1 \le i \le r, Q_i = \emptyset, 1 \le j \le 2\} \\ & \cup \{D\} \\ & \cup \{[D,i,3] \mid 1 \le i \le r\} \text{ where } U, D, Z \notin V, \end{aligned}$$

- $$\begin{aligned} P_E = \{&[X,i-1,5] \to [X,i,1] \mid 2 \le i \le r, X \in V \cup \{Z\}\} \\ & \cup \{[X,r,5] \to [X,1,1] \mid X \in V \cup \{Z\}\} \\ & \cup \{[X,i,1] \to [X,i,2] \mid 1 \le i \le r, X \in V \cup \{Z, \widehat{Z}\}\} \\ & \cup \{[\widehat{C}_i,i,1] \to [\widehat{C}_i,i,2] \mid 1 \le i \le r\} \\ & \cup \{[U,i,1] \to [U,i,2] \mid 1 \le i \le r, Q_i = \emptyset\} \\ & \cup \{[X,i,2] \to [X,i,3] \mid 1 \le i \le r, X \in V \cup \{Z, \widehat{Z}\}\} \\ & \cup \{[\widehat{C}_i,i,2] \to [\widehat{C}_i,i,3] \mid 1 \le i \le r\} \\ & \cup \{[U,i,2] \to D \mid 1 \le i \le r, Q_i = \emptyset\} \\ & \cup \{[X,i,3] \to [X,i,4] \mid 1 \le i \le r, X \in V \cup \{Z, \widehat{Z}\}\} \\ & \cup \{[\widehat{C}_i,i,3] \to [\widehat{C}_i,i,4] \mid 1 \le i \le r\} \\ & \cup \{[D,i,3] \to D \mid 1 \le i \le r\} \\ & \cup \{[X,i,4] \to [X,i,5] \mid 1 \le i \le r, X \in V \cup \{Z\}\} \\ & \cup \{[\widehat{Z},i,4] \to [Z,i,5] \mid 1 \le i \le r\} \\ & \cup \{[\widehat{C}_i,i,4] \to [C_i,i,5] \mid 1 \le i \le r\} \\ & \cup \{[X,i,5] \to X \mid 1 \le i \le r, X \in V \cup \{Z\}\} \\ & \cup \{X \to D \mid X \in V \cup \{D\}\}, \end{aligned}$$

- $$\begin{aligned} R_1 = \{&[Z,i-1,5] \to [\widehat{Z},i,1] \mid 2 \le i \le r\} \\ & \cup \{[Z,r,5] \to [\widehat{Z},1,1]\} \\ & \cup \{[C_i,i-1,5] \to [\widehat{C}_i,i,1] \mid 2 \le i \le r, Q_i \ne \emptyset\} \\ & \cup \{[C_1,r,5] \to [\widehat{C}_1,1,1] \mid Q_1 \ne \emptyset\} \\ & \cup \{[C_i,i-1,5] \to [\widehat{C}_i,i,1][U,i,1] \mid 2 \le i \le r, Q_i = \emptyset\} \\ & \cup \{[C_1,r,5] \to [\widehat{C}_1,1,1][U,1,1] \mid Q_1 = \emptyset\} \\ & \cup \{[\widehat{Z},i,1] \to [\widehat{Z},i,2] \mid 1 \le i \le r\} \end{aligned}$$

$$\cup \{[B,i,1]\to D \mid 1 \le i \le r, \{B\} = R_i \ne \emptyset\}$$
$$\cup \{[A,i,2]\to[A,i,3][D,i,3] \mid 1 \le i \le r, \{A\} = Q_i \ne \emptyset\}$$
$$\cup \{[U,i,2]\to[D,i,3] \mid 1 \le i \le r, Q_i = \emptyset\}$$
$$\cup \{[\widehat{Z},i,2]\to[\widehat{Z},i,3] \mid 1 \le i \le r\}$$
$$\cup \{[D,i,3]\to\lambda\}$$
$$\cup \{[\widehat{C}_i,i,3]\to D \mid 1 \le i \le r,\}$$
$$\cup \{[\widehat{C}_i,i,4]\to[\alpha_i,i,5] \mid 1 \le i \le r, \alpha_i \ne \lambda\}$$
$$\cup \{[\widehat{C}_i,i,4]\to\lambda \mid 1 \le i \le r, \alpha_i = \lambda\}$$
$$\cup \{[Z,i,5]\to\lambda \mid 1 \le i \le r\},$$

- $\Delta = T$, and
- $\omega = [S,r,5][Z,r,5]$.

The main point of the simulation is that we simulate the application of the rules in their order from 1 to r, each time either simulating the rule or skipping the rule. After having simulated or skipped the rth rule we continue with the first one.
We do the simulation of a rule by introducing five different alphabets for each rule of G: for the ith rule we introduce the alphabets $[V,i,j]$ for $1 \le j \le 5$. We start the simulation or the skipping of the ith rule with a word over the alphabet $[V,i-1,5]$, then during the simulation we go through the alphabets $[V,i,j]$ for $1 \le j \le 4$, and finish with a word over the alphabet $[V,i,5]$. Consequently we can finish the whole derivation or we can continue with the next the rule.

There are more additional alphabets for coordinating the simulation: the letters $[Z,i,j]$ and $[\widehat{Z},i,k]$ for $1 \le i \le r$, $1 \le j \le 5$, and $1 \le k \le 4$ make it possible to skip the ith rule of G; the symbols $[\widehat{C}_i,i,j]$ let the agent simulate the ith rule of G; the symbols $[U,i,j]$ are introduced only if $Q_i = \emptyset$ and make it possible to deal with this case; the symbols $[D,i,3]$ ensure that the derivation is blocked if the agent simulates the ith rule of G while the non-empty permitting condition is missing.

In the following, we first show how the application of a rule of G can be simulated and we also show how the application can be skipped. Then we show why the construction of the above $wEEG$ system guarantees that only those derivations that follow a derivation of the random-context grammar G result in a terminal word.

Let us suppose that we want to simulate the application of the first rule of G: $(C_1\to\alpha_1, Q_1, R_1)$ (the case of the other rules is similar) and let us first suppose that $Q_1 \ne \emptyset$. Before the simulation the sentential form in Σ is over $\{[W,r,5] \mid W \in V \cup \{Z\}\}$.

In the first step the agent "decides" whether the current rule (in this case the first rule of G) will be simulated or will be skipped. Let us suppose that the rule is to be simulated. In this case the agent uses the rule $[C_1,r,5]\to[\widehat{C}_1,1,1]$. The other letters are rewritten by the environment, using the rules $\{[X,r,5]\to[X,1,1] \mid X \in V \cup \{Z\}\}$.

In the next step the agent checks whether or not the forbidding context is present in the sentential form. This is done in the following way: the agent introduces a D if $[B,1,1]$ is present (where $\{B\} = R_1$), while otherwise the agent does not work because $[\widehat{Z},1,1]$ is not present in the sentential form. The environment increases the second index of the symbols from 1 to 2 in this step.

In the third step the agent uses its rule $[A,1,2]\to[A,1,3][D,1,3]$ for $\{A\} = Q_1$; the environment increases the second indices from 2 to 3 in the other symbols.

In the fourth step the agent deletes $[D,1,3]$ while the environment increases the second indices from 3 to 4. In the fifth and final step the agent applies the rule $[\widehat{C}_1,1,4]\to[\alpha_1,1,5]$ or the rule $[\widehat{C}_1,1,4]\to\lambda$, which correspond to the first rule of G; the environment increases the second indices. Therefore we obtain a word over $\{[W,1,5] \mid W \in V \cup \{Z\}\}$.

If $Q_1 = \emptyset$, that is when the permitting condition is empty, the simulation is different. While the environment does the same as in the previous case, the agent applies different rules. The rule the agent uses in the first step is $[C_1,r,5]\to[\widehat{C}_1,1,1][U,1,1]$ and thus $[U,1,1]$ is introduced. In the third step this symbol is used to introduce $[D,1,3]$ and from that point the simulation continues in the same way as described above, that is when $Q_1 \neq \emptyset$.

Now we show how we can do the skipping of the first rule (the case of the other rules is the same). Let us suppose again that we have a word over the alphabet $\{[W,r,5] \mid W \in V \cup \{Z\}\}$.

The environment works in the same way as it did in the previous case, the only difference is in the behaviour of the agent. In the first step the agent chooses the rule $[Z,r,5]\to[\widehat{Z},1,1]$, in the next step the rule $[\widehat{Z},1,1]\to[\widehat{Z},1,2]$, and in the third step the rule $[\widehat{Z},1,2]\to[\widehat{Z},1,3]$. In the fourth and the fifth step the agent no longer has any rule to apply, hence it does not perform any action. By the end of these five steps we have the same word as we had before, apart from the first indices in the symbols: we have the same word over the alphabet $\{[W,1,5] \mid W \in V \cup \{Z\}\}$.

At this point the simulation or the skipping of the second rule can start and can be carried out in the previous manner. We can continue this process until the last rule, the rth one, when we can restart the whole procedure with the first rule again.

In order to finish the derivation, after having finished the simulation of a rule of G the agent chooses the rule of the form of $[Z,i,5]\to\lambda$ while the environment rewrites the remaining letters according to its rules $[X,i,5]\to X$.

Thus we have seen that $L(G) \subseteq L(\Sigma)$.

In the following we show that the eco-grammar system must follow one of the sequences of steps presented above, or otherwise the derivation would never terminate.

In the first step, when the sentential form is over $[W,i-1,5]$, the agent can work because either the left-hand side of the current rule of G is present (and thus the agent can rewrite $[C_i,i-1,5]$) or the symbol $[Z,i-1,5]$ can be rewritten. (At the end of the proof we explain why we can suppose that Z has not yet disappeared from the sentential form.)

Therefore, in this first step the agent marks a place where it can perform the application of the current rule or it can mark Z. If it marks a place for the current rule in the next steps it must check the appearance of the forbidding and the permitting context. The derivation can result in a word not containing letters D only if the check is successful. This is done in the following way: the derivation is blocked by the rule $[B, i, 1] \rightarrow D$ if the forbidding context is present, or by the rule $[\widehat{C}_i, i, 3] \rightarrow D$ if the non-empty permitting condition is missing. In the last step the agent must apply the rule corresponding to the rule of G.

Thus, we have seen that if the agent decides to mark a place for applying the current rule, then he must check whether or not the rule is applicable, and he must simulate it during the five steps. If the agent chooses the other possibility and marks Z, then in the next two steps he must increase the second index of $[\widehat{Z}, i, j]$ from 1 to 2 and from 2 to 3. In the next two steps the agent cannot work. Hence if the agent chooses to mark $[Z, i, 5]$, then the work of the whole system follows the strategy of skipping the current rule, or otherwise the derivation would be blocked.

As far as the end of the derivation is concerned, the environment has to apply the rules of the form $[X, i, 5] \rightarrow X$ for all the letters in the same derivation step, or otherwise the derivation is blocked in the next step. It can happen that the agent deletes Z before the end of the derivation but this fact does not allow any new word to be generated, so we can safely assume that the deletion of Z happens in the same derivation step as the rewritings $[X, i, 5] \rightarrow X$.

We have seen the other direction of the inclusion, $L(\Sigma) \subseteq L(G)$, which completes the proof of the lemma. □

Because $\mathcal{RC}_{ac}^{\lambda} = RE$ and because weak extended simple eco-grammar systems can be simulated by Turing machines, we obtain the following theorem:

Theorem 4.1 $\mathcal{L}(\sqsupseteq\mathcal{EEG}, \infty) = \mathcal{RE}$

5 Conclusions

In this article we presented two variants of extended simple eco-grammar systems. In both cases we have found that the modifications lead to systems withlarge generative power: all recursively enumerable languages can be obtained in these ways with only one agent.

The question of the generative power of the original model, the extended simple eco-grammar system, remains open.

Acknowledgement

The author expresses her thanks to two anonymous referees who suggested a series of improvements of the article.

References

[1] J. Csima. On extended simple eco-grammar systems. *Acta Cybernetica*, 13(4):359–373, 1998.

[2] E. Csuhaj-Varjú, J. Kelemen, A. Kelemenová, and Gh. Păun. Eco-Grammar Systems: A Grammatical Framework for Studying Lifelike Interactions. *Artificial Life*, 3:1–28, 1997.

[3] E. Csuhaj-Varjú and A. Kelemenová. Team Behaviour in Eco-Grammar Systems. *Theoretical Computer Science*, (209):213–224, 1998.

[4] E. Csuhaj-Varjú, Gh. Păun, and A. Salomaa. Conditional Tabled Eco-Grammar Systems. In Gh. Păun, editor, *Artificial Life: grammatical models*, pages 227–239. Black Sea Univ. Press, Bucharest, 1995.

[5] E. Csuhaj-Varjú, Gh. Păun, and A. Salomaa. Conditional Tabled Eco-Grammar Systems versus (E)T0L Systems. *Journal of Universal Computer Science*, 5(1):252–268, 1995.

[6] J. Dassow and V. Mihalache. Eco-Grammar Systems, Matrix Grammars and E0L Systems. In Gh. Păun, editor, *Artificial Life: grammatical models*, pages 210–226. Black Sea Univ. Press, Bucharest, 1995.

[7] J. Dassow and Gh. Păun. *Regulated Rewriting in Formal Language Theory.* Springer-Verlag, 1989.

[8] Grzegorz Rozenberg and Arto Salomaa. *The Mathematical Theory of L-systems.* Academic Press, 1980.

[9] A. Salomaa. *Formal languages.* Academic Press, 1973.

[10] P. Sosík. Eco-Grammar Systems, Decidability and the Tiling Problem. In A. Kelemenová, editor, *Proceedings of the MFCS'98 Satellite Workshop on Grammar Systems*, pages 195–213. Silesian University, Opava, 1998.

[11] M. H. ter Beek. Simple Eco-Grammar Systems with Prescribed Teams. In Gh. Păun and A. Salomaa, editors, *Grammatical Models of Multi-Agent Systems*, pages 113–135. Gordon and Breach Science Publishers, 1999.

Received December, 1999

Acta Cybernetica 14 (2000) 583-595.

On Kleene Algebras of Ternary Co-Relations

Igor Dolinka *

Abstract

In this paper we investigate identities satisfied by a class of algebras made of ternary co-relations – contravariant ("arrow-reversed") analogues of binary relations. These algebras are equipped with the operations of union, co-relational composition, iteration, converse and the empty co-relation and the so-called diagonal co-relation as constants. Our first result is that the converse-free part of the corresponding equational theory consists precisely of Kleenean equations for relations, or, equivalently, for (regular) languages. However, the rest of the equations, involving the symbol of the converse, are relatively axiomatized by involution axioms only, so that the co-relational converse behaves more like the reversal of languages, rather than the relational converse. Actually, the language reversal is explicitely used to prove this result. Therefore, we conclude that co-relations can offer a better framework than relations for the mathematical modeling of formal languages, as well as many other notions from computer science.

1. Introduction

The study of the equational theory of Kleene algebras dates back to sixties, and since then it has a vivid history. However, the term 'Kleene algebra' is of more recent date, while the above equational theory was in the first place considered as the collection of *regular identities*: pairs of regular expressions denoting the same language. It was Redko [23] who proved first that regular identities have no finite base of equational axioms, but that result became available for a larger audience only with the famous booklet of Conway [6] in 1971. Conway's model-theoretic argument is probably the best known proof of Redko's result so far.

What is even more important, Conway's ideas eventually led to further progress in the field. However, the explicite determination of a nontrivial equational base of Kleene algebras had to wait until the last decade, when Krob [19] and Bloom and Ésik [3] solved the problem: the axiomatization from [19] was based on the discovery of a beautiful connection between regular languages and finite groups, while the one in [3] came out from some deep investigations in category theory and

*Institute of Mathematics, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Yugoslavia, e-mail: dockie@unsim.ns.ac.yu

its applications in computer science (see [2]). These approaches were quite recently unified in [5, 12].

It was realized in the late seventies by relation algebraists that language algebras and Kleene relation algebras are very closely related: they satisfy precisely the same (regular) identities, so that both of these two classes generate the same variety (of Kleene algebras). Moreover, the algebras of regular languages turned out to be just the free Kleene algebras, as proved by Kozen [18] in 1979 (although this result was originally formulated in the context of dynamic algebras).

But when one considers the operations of the converse of relations and the reversal of languages, respectively, the above symmetry between languages and relations is lost. Namely, the involution axioms suffice to capture the equational properties of the reversal of languages [4], while for the converse of relations one should involve an additional identity [13], which *does not* hold for languages. Therefore, relations are not 'good enough' to model the language reversal.

On the other hand, the concept of a co-relation is quite new. Yet, it belongs to the collection of 'co-algebraic' phenomena, which have been studied for some time. Roughly speaking, the main idea is to dualize the notion of an algebra and the main algebraic constructions. The pioneering papers along this line were the ones of Eilenberg and Moore [11] and Kleisli [17], but it was Aczel and Mendler [1] who opened new directions in applying co-algebra in computer science. With this approach at hand, they managed to model (binary) trees, different deterministic and nondeterminstic transition systems, etc. Since then, co-algebraic concepts were widely applied e.g. in object-oriented programming [24] and program verification [14]. For basic notions of co-algebra, see [15, 25].

In 1971, Drbohlav [10] started to investigate co-operations on a set, which one obtains from the notion of an operation by reversing arrows and replacing products by coproducts in the category of sets. Later, this inspired Csákány [9] to introduce clones of co-operations (see also [20]). But as the classical clone theory needs its 'relational part' in order to develop full strength, so the theory of clones of co-operations needs appropriate co-objects as invariants. Hence, Pöschel and Rößiger [22] proposed the concept of a co-relation. While an n-ary relation on X can be thought of as a family of n-ary vectors over X, that is, functions $n \to X$, an *n-ary co-relation on X* is a collection of functions $X \to n$ (n-ary co-vectors on X), which should be imagined just as colourings (partitions) of X in n colours (into n classes).

In [20], the operation of composition was defined for arbitrary co-relations; however, the result of the composition of two n-ary co-relations is again an n-ary co-relation if and only if $n = 3$. Of course, binary co-relations quite clearly correspond to unary relations (subsets). Thus it is natural to expect that the role and importance of binary relations is inherited by ternary co-relations on a set.

In this paper, we consider algebras consisting of ternary co-relations, endowed with the operations of union, composition, iteration (in the sense of the complete union of composition powers), co-relational converse and with two distinguished constants. Our main result is that such algebras generate the same variety as the language algebras equipped with the operations of union, concatenation, Kleene star, reversal and the empty lanugage and the language containing the empty word

only, as constants. In particular, it follows that the converse-free reducts of these co-relation algebras are indeed Kleene algebras, justifying the title of the paper. Therefore, we are going to eventually conclude that, from the equational point of view, co-relations model (the operations on) languages better than relations.

For basics of universal algebra we refer to [21] and for the theory of binary relations to [16]. The same references hold for all undefined notions throughout the paper.

2. Preliminaries

2.1. Kleene algebras

Let X be any set. Consider the following algebra:

$$\mathbf{Rel}(X) = \langle \mathcal{P}(X \times X), \cup, \circ, ^{\mathrm{rtc}}, \emptyset, \Delta_X \rangle,$$

where $\cup$ is the union, $\circ$ is the composition of relations, ^{rtc} is the formation of the reflexive-transitive closure, while Δ_X is the diagonal relation on X. The algebra $\mathbf{Rel}(X)$ is called the *full Kleene algebra of relations on* X. Any algebra which can be embedded into some full Kleene relation algebra is called *representable* (or *standard*) *Kleene algebra*. The variety generated by all algebras $\mathbf{Rel}(X)$ we denote by $\mathcal{KA}$. A *Kleene algebra* is just any member of $\mathcal{KA}$.

Beyond algebras of relations, the most important example of Kleene algebras is the *language algebra* over an alphabet Σ:

$$\mathbf{Lang}(\Sigma) = \langle \mathcal{P}(\Sigma^*), +, \cdot, ^*, \emptyset, \{\lambda\} \rangle,$$

where Σ^* is the free monoid on Σ (which consists of all words over Σ), $+$ denotes the union, $\cdot$ is the concatenation, * is the Kleene star (iteration), and finally, λ denotes the empty word. The fact that language algebras indeed belong to $\mathcal{KA}$ is a consequence of a more general observation.

Lemma 1. *Let M be any monoid. Then $\mathbf{K}(M) = \langle \mathcal{P}(M), \cup, \cdot, ^*, \emptyset, \{1\} \rangle$, where $\cdot$ is the complex multiplication, * the generation of a submonoid, and 1 the unit of M, is a Kleene algebra.*

Proof (in outline). Consider the mapping $\xi : \mathcal{P}(M) \to \mathcal{P}(M \times M)$ defined for every $A \subseteq M$ by

$$\xi(A) = \{\langle x, xa \rangle : x \in M, a \in A\} = \bigcup_{a \in A} \varrho_a,$$

where ϱ_a denotes the right translation of the monoid M. It is a routine matter to show that ξ is an embedding of $\mathbf{K}(M)$ into $\mathbf{Rel}(M)$. □

By taking $M = \Sigma^*$, from the above lemma we immediately obtain that $\mathbf{Lang}(\Sigma) = \mathbf{K}(\Sigma^*)$ is a Kleene algebra for any alphabet Σ.

The elements of the subalgebra of $\mathbf{Lang}(\Sigma)$ generated by the languages of the form $\{a\}$, $a \in \Sigma$ (or, equivalently, by all finite languages), are called the *regular languages over* Σ. This subalgebra is denoted by $\mathbf{Reg}(\Sigma)$. Now, the algebras of regular languages have the following remarkable property.

Proposition 2. (Kozen, [18]) $\mathbf{Reg}(\Sigma)$ *is the free Kleene algebra on* Σ, *freely generated by the map* $a \mapsto \{a\}$, $a \in \Sigma$.

Thus, it follows that an identity $p = q$ holds in $\mathcal{KA}$ if and only if the regular expressions p, q represent the same (regular) language. Also, the above proposition implies that if we denote by $\mathcal{L}$ the variety generated by all language algebras $\mathbf{Lang}(\Sigma)$, then $\mathcal{L} = \mathcal{KA}$.

We are not going to state here the well known nonfinite axiomatizability result for $\mathcal{KA}$, due to Redko [23], nor the explicite axiomatizations given by Krob [19] and Bloom and Ésik [3]. However, when one is concerned with Kleene algebras or relations and languages, it is quite customary to consider one more operation. First, we have the natural operation of the *reversal* of languages. If $w = a_1 a_2 \dots a_n \in \Sigma^*$ is a word, we define

$$w^\vee = a_n \dots a_2 a_1.$$

Now we have

$$L^\vee = \{w^\vee : w \in L\}.$$

By adding the operation of the reversal of languages to $\mathbf{Lang}(\Sigma)$ we obtain the algebra $\mathbf{Lang}^\vee(\Sigma)$. The variety generated by algebras of this form we denote by $\mathcal{L}^\vee$.

Proposition 3. (Bloom, Ésik and Stefanescu, [4]) *The variety* $\mathcal{L}^\vee$ *is axiomatized relatively to* $\mathcal{KA}$ *by the involution axioms, that is, by the following identities:*

$$\begin{aligned}
(x+y)^\vee &= x^\vee + y^\vee, && (1)\\
(xy)^\vee &= y^\vee x^\vee, && (2)\\
(x^*)^\vee &= (x^\vee)^*, && (3)\\
(x^\vee)^\vee &= x, && (4)\\
0^\vee &= 0, && (5)\\
1^\vee &= 1. && (6)
\end{aligned}$$

There is one more way to define an involutorial operation on language algebras, which can be useful in some applications. For an alphabet Σ, let Σ' denote a bijective copy of Σ, $\Sigma' = \{a' : a \in \Sigma\}$. For $w = b_1 b_2 \dots b_n \in (\Sigma \cup \Sigma')^*$ define

$$w' = b'_n \dots b'_2 b'_1,$$

where for all $1 \le i \le n$,

$$b'_i = \begin{cases} a', & b_i = a \in \Sigma, \\ a, & b_i = a' \in \Sigma'. \end{cases}$$

Finally, let $\mathbf{Lang}'(\Sigma, \Sigma')$ be the language algebra $\mathbf{Lang}(\Sigma \cup \Sigma')$ endowed with the unary operation $'$, where, of course, $L' = \{w' : w \in L\}$.

By Proposition 4.3 and Theorem 5.1 of [4], algebras $\mathbf{Lang}'(\Sigma, \Sigma')$ also generate the variety $\mathcal{L}^{\vee}$.

On the other hand, the operation which extends Kleene algebras of relations is the *converse*:

$$\varrho^{\vee} = \{\langle y, x\rangle : \langle x, y\rangle \in \varrho\}.$$

By equipping the algebra $\mathbf{Rel}(X)$ with $^{\vee}$, we obtain the algebra $\mathbf{Rel}^{\vee}(X)$. All algebras of the latter form generate the variety $\mathcal{KA}^{\vee}$, which turns out to be a *proper* subvariety of $\mathcal{L}^{\vee}$.

Proposition 4. (Ésik and Bernátsky, [13]) *The equations (1)–(6) and*

$$x + xx^{\vee}x = xx^{\vee}x \tag{7}$$

axiomatize the variety $\mathcal{KA}^{\vee}$ relatively to $\mathcal{KA}$.

Therefore, we may conclude that the equational properties of the language reversal are not faithfully modeled by the relational converse and hence, it is natural to look after a different setting which would allow to capture those properties, preserving at the same time the Kleenean equations.

2.2. Co-Relations

Clearly, an n-ary relation on X can be thought of as a collection of n-ary vectors over X, that is, functions $n \to X$. Dually, an *n-ary co-relation on X* is a set consisting of *n-ary co-vectors*, i.e. of functions $X \to n$. Of course, the notion of a n-ary co-vector is equivalent to the notion of a colouring of a given set in n colours. In particular, a *ternary co-relation* is a family of functions $X \to 3$. It is convenient to represent a ternary co-vector $f : X \to 3$ through the corresponding partition of X into $A = f^{-1}(0)$, $B = f^{-1}(1)$ and $C = f^{-1}(2)$, so that f is written as $\langle A, B, C\rangle^{\nabla}$ (we use the symbol $^{\nabla}$ to indicate that we are not dealing with a ternary *vector* whose elements are A, B, C). In order to introduce a more intuitive (and visualisable) terminology, we are going to call the colours $0, 1, 2$ (i.e. the elements from A, B, C) respectively *red*, *green* and *blue*.

In this paper, we deal with algebras of ternary co-relations of the form

$$\mathbf{cRel}^{\sqcup}(X) = \langle \mathcal{P}(3^X), \cup, \bullet, ^{\star}, ^{\sqcup}, \emptyset, \varepsilon_X\rangle$$

(the reduct without $^{\sqcup}$ is denoted by $\mathbf{cRel}(X)$), where the operations and the constants are defined below. First of all, $\cup$ is simply the set-theoretic union, while the constant ε_X is the co-relation consisting of all green-free colourings of X, that is,

$$\varepsilon_X = \{\langle A, \emptyset, X \setminus A\rangle^{\nabla} : A \subseteq X\}.$$

The definition of the co-relational composition $\bullet$ is the following:

$$\varrho \bullet \sigma = \{\langle A, B \cup E, F\rangle^{\nabla} : (\exists C, D \subseteq X)(\langle A, B, C\rangle^{\nabla} \in \varrho \wedge \langle D, E, F\rangle^{\nabla} \in \sigma \wedge C = X \setminus D)\}.$$

In other words, two co-vectors can be composed if the blue set of the first one coincides with the non-red part (green+blue) of the second one (or, equivalently, red+green elements of the first are precisely the red elements of the second co-vector). If that is the case, green elements are added together, the red colour is copied from the first and the blue from the second factor.

Since one can define arbitrary unions of co-relations, the unary operation * is just the co-relation analogue of the reflexive-transitive closure of relations, or of the Kleene iteration. If for a ternary co-relation ϱ and $n \geq 1$ we define

$$\varrho^n = \underbrace{\varrho \bullet \ldots \bullet \varrho}_{n}$$

and $\varrho^0 = \varepsilon_X$, then

$$\varrho^\star = \bigcup_{n \geq 0} \varrho^n.$$

Finally, the co-relational converse $^\sqcup$ is defined as the interchanging of the red and the blue colour:

$$\varrho^\sqcup = \{\langle C, B, A\rangle^\nabla : \langle A, B, C\rangle^\nabla \in \varrho\}.$$

In the sequel, we shall need the following fact (whose proof is omitted as being immediate).

Lemma 5. *For any set X, the algebra $\mathbf{cRel}^\sqcup(X)$ satisfies the identities (1)–(6).*

However, note that for all nonempty X, $\mathbf{cRel}^\sqcup(X)$ does not satisfy (7), because for $\varrho = \{\langle \emptyset, X, \emptyset\rangle^\nabla\}$ we have $\varrho \bullet \varrho^\sqcup \bullet \varrho = \emptyset$.

3. The Results

First of all, we prove that the co-relation algebras $\mathbf{cRel}(X)$ are Kleene algebras. Moreover, all such algebras are representable.

Proposition 6. *For any set X, the algebra $\mathbf{cRel}(X)$ is isomorphic to a subalegbra of $\mathbf{Rel}(\mathcal{P}(X))$.*

Proof. Define a mapping $\Theta : \mathcal{P}(3^X) \to \mathcal{P}(\mathcal{P}(X) \times \mathcal{P}(X))$ by

$$\Theta(\varrho) = \{\langle A, A \cup B\rangle : \langle A, B, C\rangle^\nabla \in \varrho\}.$$

It is immediately clear that Θ is injective and completely additive. It remains to prove that for all $\varrho, \sigma \subseteq 3^X$ we have $\Theta(\varrho \bullet \sigma) = \Theta(\varrho) \circ \Theta(\sigma)$ (for then it follows from the the complete additivity that we have $\Theta(\varrho^\star) = (\Theta(\varrho))^{\mathrm{rtc}}$).

Indeed, let $\langle A, B\rangle \in \Theta(\varrho \bullet \sigma)$. Clearly, this is the same as saying that $A \subseteq B$ and $\langle A, B \setminus A, X \setminus B\rangle^\nabla \in \varrho \bullet \sigma$. The latter condition is just equivalent to the existence of

$B_1, B_2, C, D \subseteq X$ such that $C = X \setminus D$, $\langle A, B_1, C\rangle^\nabla \in \varrho$, $\langle D, B_2, X \setminus B\rangle^\nabla \in \sigma$ and $B_1 \cup B_2 = B \setminus A$. Note that from here C can be eliminated, namely $C = X \setminus (A \cup B_1)$, so that we arrive at $\langle A, A \cup B_1\rangle \in \Theta(\varrho)$ and $\langle A \cup B_1, A \cup B_1 \cup B_2\rangle \in \Theta(\sigma)$, where $B_1 \cup B_2 = B \setminus A$. But recall that $A \subseteq B$, so that $\langle A, B\rangle \in \Theta(\varrho \bullet \sigma)$ is the same as $\langle A, A \cup B_1\rangle \in \Theta(\varrho)$ and $\langle A \cup B_1, B\rangle \in \Theta(\sigma)$ for some $B_1 \subseteq B$, i.e. $\langle A, B\rangle \in \Theta(\varrho) \circ \Theta(\sigma)$, which finishes the proof. □

The combined effect of Lemma 5 and the above proposition is just as follows.

Corollary 7. *For all X, $\mathbf{cRel}^{\sqcup}(X) \in \mathcal{L}^\vee$.*

Now let Σ be an alphabet, $x \in \Sigma$, and let

$$w = a_1 a_2 \ldots a_n \in (\Sigma \cup \Sigma')^*$$

be any word (Σ' is, as in the previous section, a bijective copy of Σ). Define a mapping $\psi_w : \Sigma \to \mathcal{P}(3^{\underline{n}})$ (where we use the following notation: $\underline{n} = \{1, 2, \ldots, n\}$ and $\underline{0} = \emptyset$) by

$$\psi_w(x) = \{\langle \underline{i-1}, \{i\}, \underline{n} \setminus \underline{i}\rangle^\nabla : a_i = x\} \cup \{\langle \underline{n} \setminus \underline{i}, \{i\}, \underline{i-1}\rangle^\nabla : a_i = x'\}.$$

Since by Proposition 4.2 from [4] we have that $\mathbf{Lang}'(\Sigma, \Sigma')$ is the free object on Σ in the category of completely idempotent semirings with involution (to which $\mathbf{cRel}^{\sqcup}(X)$ certainly belongs, for all X), the mapping defined above can be extended (by identifying x and $\{x\}$ for all $x \in \Sigma$) to a morphism $\Psi_w : \mathbf{Lang}'(\Sigma, \Sigma') \to \mathbf{cRel}^{\sqcup}(\underline{n})$ (recall that $n = |w|$).

It is not difficult to see that the following assertions hold:

(a) $\Psi_w(\{w\}) = \Psi_w(\{a_1\}) \ldots \Psi_w(\{a_n\}) = \psi_w(a_1) \ldots \psi_w(a_n) = \{\langle \emptyset, \underline{n}, \emptyset\rangle^\nabla\}$.

(b) $\Psi_w(\{\lambda\}) = \varepsilon_{\underline{n}}$. In particular, $\Psi_\lambda(\{\lambda\}) = \varepsilon_{\underline{0}} = \{\langle \emptyset, \emptyset, \emptyset\rangle^\nabla\}$.

(c) If u is a nonempty subword of w, say $u = a_i \ldots a_j$, then, similarly as in (a),

$$\Psi_w(\{u\}) = \{\langle \underline{i-1}, \underline{j} \setminus \underline{i-1}, \underline{n} \setminus \underline{j}\rangle^\nabla\}.$$

Otherwise, $\Psi_w(\{u\}) = \emptyset$.

(d) If L is a language over $\Sigma \cup \Sigma'$, then

$$\Psi_w(L) = \Psi_w(\{u : u \text{ is a subword of } w \text{ such that } u \in L\}).$$

Therefore, for any word w, we have the following equivalence:

$$w \in L \iff \langle \emptyset, \underline{n}, \emptyset\rangle^\nabla \in \Psi_w(L). \tag{8}$$

Finally, let

$$\Psi : \mathbf{Lang}'(\Sigma, \Sigma') \to \prod_{w \in (\Sigma \cup \Sigma')^*} \mathbf{cRel}^{\sqcup}(\underline{|w|})$$

be the target tupling of the morphisms Ψ_w, that is, the (unique) function satisfying the condition $\Psi \circ \pi_w = \Psi_w$ for all $w \in (\Sigma \cup \Sigma')^*$ (where π_w is the projection of the above direct product corresponding to w).

Proposition 8. *Ψ is an embedding of completely idempotent semirings with involution (and thus, of Kleene algebras with involution).*

Proof. Since all functions Ψ_w are morphisms of completely idempotent semirings with involution, it suffices to prove that Ψ is injective. But this easily follows from the equivalence (8), because if L_1, L_2 are two different languages over $\Sigma \cup \Sigma'$ and $w_0 \in L_1 \setminus L_2$, $n_0 = |w_0|$, then by (8) we have that $\langle \emptyset, \underline{n_0}, \emptyset \rangle^{\nabla} \in \Psi_{w_0}(L_1) \setminus \Psi_{w_0}(L_2)$. Hence, $\Psi_{w_0}(L_1) \neq \Psi_{w_0}(L_2)$, and so $\Psi(L_1) \neq \Psi(L_2)$. □

As the algebras $\mathbf{Lang}'(\Sigma, \Sigma')$ generate the variety $\mathcal{L}^{\vee}$, we have just proved

Theorem 9. *The variety generated by all algebras of co-relations $\mathbf{cRel}^{\sqcup}(X)$ coincides with $\mathcal{L}^{\vee}$.*

Hence, we may say that the equational behaviour of the language reversal is modeled by ternary co-relations.

On the other hand, it is interesting to see how one obtains Kleene algebras of relations from those of co-relations, provided that we drop the converse operation. It turns out that we do not need the (slightly cumbersome) construction of the direct product: we shall prove that $\mathbf{Rel}(X) \in \mathsf{HS}(\mathbf{cRel}(\omega \times X))$ for all X, i.e. that $\mathbf{Rel}(X)$ is a quotient of a subalgebra of $\mathbf{cRel}(\omega \times X)$. It is worth noting that $\omega \times X$ is just the ω-copower of X (coproduct of ω copies of X) in the category of sets.

First of all, choose a linear order $\leq$ on X, so that $\langle X, \leq \rangle$ is a chain. Further, define a linear order relation $\preceq$ on $\omega \times X$ as follows:

$$\langle k, x_1 \rangle \preceq \langle \ell, x_2 \rangle \text{ if and only if } k < \ell \text{ or } k = \ell,\ x_1 \leq x_2.$$

A ternary co-vector over X (3-colouring of X) of the form $\mathbf{c}_{u,v}^{k,\ell} = \langle A, B, C \rangle^{\nabla}$,

$$\begin{aligned} A &= \{\langle n, x \rangle : \langle n, x \rangle \preceq \langle k, u \rangle\}, \\ B &= \{\langle n, x \rangle : \langle k, u \rangle \prec \langle n, x \rangle \preceq \langle \ell, v \rangle\}, \\ C &= \{\langle n, x \rangle : \langle \ell, v \rangle \prec \langle n, x \rangle\}, \end{aligned}$$

where $\langle k, u \rangle \preceq \langle \ell, v \rangle$, we call a *cutting* of the set $\omega \times X$. Now for $m \in \omega$ let

$$\chi_{u,v}^{m} = \{\mathbf{c}_{u,v}^{k,\ell} : \ \ell - k = m\}.$$

Note that $\chi_{u,v}^{0}$ is nonempty if and only if $u \leq v$.

A ternary co-relation on $\omega \times X$ is a *closed set of cuttings* if it is representable as a union of co-relations of the form $\chi_{u,v}^{m}$. Alternatively, we can define a closed set of cuttings as a family ϱ of cuttings satisfying, for all $u, v \in X$, the condition

$$(\exists p, q \in \omega)\ \mathbf{c}_{u,v}^{p,q} \in \varrho \ \Rightarrow\ (\forall n \in \omega)\ \mathbf{c}_{u,v}^{n,n+(q-p)} \in \varrho.$$

Finally, we call a ternary co-relation on $\omega \times X$ *good* if it is a union of a closed set of cuttings and a green-free co-relation (that is, a subset of $\varepsilon_{\omega \times X}$) which contains *no* cuttings. The set of all good co-relations on $\omega \times X$ is denoted by $G(X)$.

Lemma 10. *$G(X)$ is the universe of a subalgebra $\mathbf{G}(X)$ of* $\mathbf{cRel}(\omega \times X)$.

Proof. First of all, it is clear that $G(X)$ is closed for *arbitrary* unions and that $\emptyset \in G(X)$. Also, $\varepsilon_{\omega\times X} \in G(X)$, because

$$\varepsilon_{\omega\times X} = \left(\bigcup_{x\in X} \chi^0_{x,x}\right) \cup \varepsilon',$$

where ε' is the set of all green-free co-vectors over $\omega \times X$ which are not cuttings. Hence, the lemma will be proved if we show that the composition of two good co-relations remains good.

Therefore, let

$$\begin{aligned} \varrho_1 &= \left(\bigcup_{i\in I} \chi^{m_i}_{u_i,v_i}\right) \cup \varepsilon_1, \\ \varrho_2 &= \left(\bigcup_{j\in J} \chi^{m_j}_{u_j,v_j}\right) \cup \varepsilon_2, \end{aligned}$$

where $\varepsilon_1, \varepsilon_2$ are green-free co-relations containing no cuttings, and I, J are disjoint. Since $\bullet$ is completely distributive over $\cup$ (recall Proposition 6), we have:

$$\varrho_1 \bullet \varrho_2 = \left(\bigcup_{\langle i,j\rangle\in I\times J} (\chi^{m_i}_{u_i,v_i} \bullet \chi^{m_j}_{u_j,v_j})\right) \cup \left(\bigcup_{i\in I} (\chi^{m_i}_{u_i,v_i} \bullet \varepsilon_2)\right) \cup \left(\bigcup_{j\in J} (\varepsilon_1 \bullet \chi^{m_j}_{u_j,v_j})\right) \cup (\varepsilon_1 \bullet \varepsilon_2).$$

It is easy to see that the following holds:

$$\chi^k_{u,v} \bullet \chi^\ell_{z,t} = \begin{cases} \chi^{k+\ell}_{u,t}, & v = z, \\ \emptyset, & v \neq z. \end{cases}$$

Also, note that a co-vector which is a cutting can be composed (from the left or from the right) with a green-free co-vector only if the latter is a cutting, too (because the blue part of the considered green-free co-vector must coincide either with the green+blue part, or with the blue part of the cutting which it is composed with). Thus for all $u, v \in X$ and $m \in \omega$ we have

$$\chi^m_{u,v} \bullet \varepsilon_2 = \varepsilon_1 \bullet \chi^m_{u,v} = \emptyset.$$

Finally, it is not difficult to see that the composition of two green-free co-relations coincides with their intersection (because two green-free co-vectors can be composed if and only if they are equal), so

$$\varepsilon_1 \bullet \varepsilon_2 = \varepsilon_1 \cap \varepsilon_2,$$

which is a green-free co-relation containing no cuttings. We conclude that $\varrho_1 \bullet \varrho_2$ is a good co-relation. □

Now define a relation $\equiv$ on $G(X)$ by

$$\varrho_1 \equiv \varrho_2 \text{ if and only if } (\forall u, v \in X)((\exists k \in \omega)\ \chi^k_{u,v} \subseteq \varrho_1 \ \Leftrightarrow\ (\exists \ell \in \omega)\ \chi^\ell_{u,v} \subseteq \varrho_2).$$

Proposition 11. *The relation $\equiv$ is a congruence of $\mathbf{G}(X)$ and*

$$\frac{\mathbf{G}(X)}{\equiv} \cong \mathbf{Rel}(X).$$

Proof. Define a mapping $\Upsilon : G(X) \to \mathcal{P}(X \times X)$ by

$$\Upsilon(\varrho) = \{\langle u, v\rangle :\ (\exists k \in \omega)\ \chi^k_{u,v} \subseteq \varrho\}.$$

It is obvious that the kernel of Υ coincides with $\equiv$. Thus it remains to prove that Υ is a surjective morphism of complete semirings.

First, for $\sigma \subseteq X \times X$ define

$$\varrho_\sigma = \bigcup_{\langle u,v\rangle \in \sigma} \chi^1_{u,v}.$$

According to the above definition, $\Upsilon(\varrho_\sigma) = \sigma$. Hence, Υ is surjective.

Now we prove that Υ is completely additive. We have:

$$\Upsilon(\textstyle\bigcup_{i\in I} \varrho_i) = \{\langle u, v\rangle :\ (\exists k \in \omega)\ \chi^k_{u,v} \subseteq \bigcup_{i\in I} \varrho_i\}.$$

But all the co-relations ϱ_i are good, which means that if $\mathbf{c}^{p,q}_{u,v} \in \varrho_i$, then $\chi^{q-p}_{u,v} \subseteq \varrho_i$. Therefore, if $\varrho_i \cap \chi^k_{u,v} \neq \emptyset$, then $\chi^k_{u,v} \subseteq \varrho_i$, and so $\chi^k_{u,v} \subseteq \bigcup_{i\in I} \varrho_i$ implies $\chi^k_{u,v} \subseteq \varrho_{i_0}$ for some $i_0 \in I$. Clearly, the converse of the latter conclusion is true, which amounts to say that $\Upsilon(\bigcup_{i\in I} \varrho_i) = \bigcup_{i\in I} \Upsilon(\varrho_i)$.

Finally, let $\varrho_1 = \theta_1 \cup \varepsilon_1$ and $\varrho_2 = \theta_2 \cup \varepsilon_2$ be two good co-relations, where θ_1, θ_2 are closed sets of cuttings and $\varepsilon_1, \varepsilon_2$ are green-free co-relations containing no cuttings. As seen in the previous lemma, we have

$$\varrho_1 \bullet \varrho_2 = (\theta_1 \bullet \theta_2) \cup (\varepsilon_1 \bullet \varepsilon_2).$$

Now we have the following chain of equivalences:

$$\begin{aligned}
\langle u, v\rangle \in \Upsilon(\varrho_1 \bullet \varrho_2) \quad &\Leftrightarrow \quad (\exists k \in \omega)\ \chi^k_{u,v} \subseteq \varrho_1 \bullet \varrho_2 \\
&\Leftrightarrow \quad (\exists k \in \omega)\ \chi^k_{u,v} \subseteq \theta_1 \bullet \theta_2 \\
&\Leftrightarrow \quad (\exists z \in X)(\exists p, q \in \omega)(\chi^p_{u,z} \subseteq \theta_1 \subseteq \varrho_1 \ \wedge\ \chi^q_{z,v} \subseteq \theta_2 \subseteq \varrho_2) \\
&\Leftrightarrow \quad (\exists z \in X)(\langle u, z\rangle \in \Upsilon(\varrho_1) \ \wedge\ \langle z, v\rangle \in \Upsilon(\varrho_2)) \\
&\Leftrightarrow \quad \langle u, v\rangle \in \Upsilon(\varrho_1) \circ \Upsilon(\varrho_2)
\end{aligned}$$

So, $\Upsilon(\varrho_1 \bullet \varrho_2) = \Upsilon(\varrho_1) \circ \Upsilon(\varrho_2)$, and the proposition is proved. □

Finally, it is well known that any direct product of full Kleene relation algebras (possibly with converse) is a represetable Kleene algebra. Namely, such a direct product (say, of $\mathbf{Rel}^\vee(X_i)$, $i \in I$) can be embedded into the full Kleene algebra of the relations on $\coprod_{i\in I} X_i$, the coproduct (disjoint union) of the base sets X_i. In the last assertion of this paper, we note that the direct product of co-relation algebras $\mathbf{cRel}^\sqcup(X_i)$ can be in a similar fashion represented by co-relations.

Proposition 12. *Any direct product of algebras of the form* $\mathbf{cRel}^{\sqcup}(X)$ *is embeddable into an algebra of that form. More precisely, the direct product of algebras* $\mathbf{cRel}^{\sqcup}(X_i)$, $i \in I$, *is isomorphic to a subalgebra of* $\mathbf{cRel}^{\sqcup}(\coprod_{i\in I} X_i)$, *where* $\coprod_{i\in I} X_i$ *denotes the coproduct of the sets* X_i.

Proof (in outline). In order to relax the notation, we may assume that the sets X_i are already disjoint and argue that $\prod_{i\in I} \mathbf{cRel}^{\sqcup}(X_i)$ is embeddable into $\mathbf{cRel}^{\sqcup}(X)$, where $X = \bigcup_{i\in I} X_i$. Consider the mapping $\varphi : \prod_{i\in I} \mathcal{P}(3^{X_i}) \to \mathcal{P}(3^X)$ given by

$$\varphi(\langle \varrho_i : i \in I\rangle) = \bigcup_{i\in I} \varrho_i.$$

One shows in a routine way that φ is an injecitve morphism of complete semirings with involution. □

Therefore, the embedding Ψ from Proposition 8 composed with the embedding φ from the above proposition gives an embedding of the language algebra $\mathbf{Lang}'(\Sigma, \Sigma')$ into $\mathbf{cRel}^{\sqcup}(S)$, where S is a set of cardinality $|\Sigma| + \aleph_0$.

4. An Open Problem

The algebras $\mathbf{cRel}^{\sqcup}(X)$, whose identities were investigated in this paper, turned up as categorical duals of Kleene relation algebras (with converse). However, we can consider another kind of co-relation algebras which arise from the analogy with relation algebras of Tarski (by droping the operation of iteration and taking all of the Boolean operations):

$$\mathbf{cR}(X) = \langle \mathcal{P}(3^X), \cup, \cap, ^-, \emptyset, 3^X, \bullet, ^{\sqcup}, \varepsilon_X\rangle.$$

It is well known (Monk, 1964) that the variety generated by the corresponding relation algebras is not finitely axiomatizable. Also, several explicite axiomatizations are known. Here we raise the question whether the same is true for the variety determined by algebras of the form $\mathbf{cR}(X)$. First of all, it would be interesting to give any nontrivial equational axiomatization for this variety (or any other description of its equational theory). Of course, we have proved in the present paper that the equations of co-relation algebras $\mathbf{cR}(X)$ not involving $\cap, ^-, 3^X$, are just those of idempotent unitary semirings with involution. However, the equations of relation and co-relation algebras which contain the above symbols are *not* equal, since the famous *Tarski identity*:

$$(x^{\vee} \circ (\overline{x \circ y})) \cap y = \emptyset,$$

does not hold for co-relations (see [20]).

Aknowledgement. The author is grateful to Dragan Mašulović for providing a copy of [20] and for many valuable conversations we led concerning the theory of ternary co-relations.

References

[1] Aczel, P. and Mendler, N. P., *A final coalgebra theorem.* In: eds. D. H. Pitt et al., "Category Theory and Computer Science", Lecture Notes Comput. Sci., Vol. 389, pp. 357–365, Springer-Verlag, 1989.

[2] Bloom, S. L. and Ésik, Z., "Iteration Theories: The Equational Logic of Iterative Processes". EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1993.

[3] Bloom, S. L. and Ésik, Z., *Equational axioms for regular sets.* Math. Struct. Comput. Sci. **3** (1993), 1–24.

[4] Bloom, S. L., Ésik, Z. and Stefanescu, Gh., *Notes on equational theories of relations.* Algebra Universalis **33** (1995), 98–126.

[5] Bloom, S. L. and Ésik, Z., *The equational logic of fixed points.* Theoret. Comput. Sci. **179** (1997), 1–60.

[6] Conway, J. H., "Regular Algebra and Finite Machines". Chapman & Hall, 1971.

[7] Crvenković, S. and Madarász, R. Sz., *On Kleene algebras.* Theoret. Comput. Sci. **108** (1993), 17–24.

[8] Crvenković, S., Dolinka, I. and Ésik, Z., *The variety of Kleene algebras with conversion is not finitely based.* Theoret. Comput. Sci. **230** (2000), 235–245.

[9] Csákány, B., *Completeness in coalgebras.* Acta Sci. Math. (Szeged) **48** (1985), 75–84.

[10] Drbohlav, K., *On quasicovarieties.* Acta Fac. Rerum Natur. Univ. Comenian. Math., Special Issue (1971), 17–20.

[11] Eilenberg, S. and Moore, J. C., *Adjoint functors and triples.* Illinois J. Math. **9** (1965), 381–398.

[12] Ésik, Z., *Group axioms for iteration.* Inform. Comput. **148** (1999), 131–180.

[13] Ésik, Z. and Bernátsky, L., *Equational properties of Kleene algebras of relations with conversion.* Theoret. Comput. Sci. **137** (1995), 237–251.

[14] Jacobs, B., *Objects and classes, co-algebraically.* In: eds. B. Freitag et al., "Object-Orientation with Paralelism and Persistance", pp. 83–103, Kluwer Academic Publishers, 1996.

[15] Jacobs, B. and Rutten, J. J. M. M., *A tutorial on (co)algebra and (co)induction.* EATCS Bull. **62** (1997), 222–259.

[16] Jónsson, B., *The theory of binary relations.* In: eds. H. Andréka, J. D. Monk and I. Németi, "Algebraic Logic" (Budapest, 1988), Colloq. Math. Soc. János Bolyai, Vol. 54, pp. 245–292, North-Holland, 1991.

[17] Kleisli, H., *Every standard construction is induced by a pair of adjoint functors.* Proc. Amer. Math. Soc. **16** (1965), 544–546.

[18] Kozen, D., *A representation theorem for models of *-free PDL.* Report RC 7864, IBM Research, Yorktown Heights, 1979.

[19] Krob, D., *Complete systems of $\mathcal{B}$-rational identities.* Theoret. Comput. Sci. **89** (1991), 207–343.

[20] Mašulović, D., "The Lattice of Clones of Co-Operations" (in Serbian). Ph.D. thesis, viii+216 pp., University of Novi Sad, 1999.

[21] McKenzie, R. N., McNulty, G. F. and Taylor, W. F., "Algebras, Lattices, Varieties", Vol. I. Wadsworth & Brooks/Cole, 1987.

[22] Pöschel, R. and Rößiger, M., *A general Galois theory for co-functions and co-relations.* Preprint MATH-AL-11-1997, Technische Universität Dresden, 1997.

[23] Redko, V. N., *On defining relations for the algebra of regular events* (in Russian). Ukrainian Math. J. **16** (1964), 120–126.

[24] Reichel, H., *An approach to object semantics based on terminal co-algebras.* Math. Struct. Comput. Sci. **5** (1995), 129–152.

[25] Rutten, J. J. M. M., *Universal coalgebra: A theory of systems.* CWI Technical Report CS-R9652, 1996.

Received February, 2000

Acta Cybernetica 14 (2000) 597-605.

Results concerning E0L and C0L power series

Juha Honkala *

Abstract

By a classical result of Ehrenfeucht and Rozenberg the families of E0L and C0L languages are equal. We generalize this result for E0L and C0L power series satisfying the ε-condition which restricts the coefficients of the empty word.

1 Introduction

A celebrated result from classical theory of Lindenmayer systems states that the families of E0L languages and C0L languages are equal (see Ehrenfeucht and Rozenberg [1], Rozenberg and Salomaa [5,6]). In this paper we generalize this result for formal power series. We will work in the framework of morphically generated formal power series introduced in Honkala [2,3] and Honkala and Kuich [4].

In what follows A will always be a commutative ω-continuous semiring (see [4]). Suppose Σ is a finite alphabet. The set of *formal power series with noncommuting variables* in Σ and coefficients in A is denoted by $A \ll \Sigma^* \gg$. The subset of $A \ll \Sigma^* \gg$ consisting of all series with a finite support is denoted by $A < \Sigma^* >$. Series of $A < \Sigma^* >$ are referred to as *polynomials*. A semialgebra morphism $h : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ is specified by the polynomials $h(\sigma)$, $\sigma \in \Sigma$. If $h(\sigma)$ is quasiregular for all $\sigma \in \Sigma$, the semialgebra morphism h is called *propagating*. If Δ is a finite alphabet, a semialgebra morphism $h : A < \Sigma^* > \longrightarrow A < \Delta^* >$ is called a *coding* if for each $\sigma \in \Sigma$ there exist a nonzero $a \in A$ and a letter $x \in \Delta$ such that $h(\sigma) = ax$.

We are going to discuss 0L, P0L, E0L, EP0L and C0L power series. By definition, a power series $r \in A \ll \Sigma^* \gg$ is called a *0L power series* if there exist $a \in A$, $w \in \Sigma^*$ and a semialgebra morphism $h : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ such that

$$r = \sum_{n=0}^{\infty} ah^n(w). \tag{1}$$

If in (1) the semialgebra morphism h is propagating, r is called a *P0L power series*. E0L and EP0L power series are now defined in the natural way (see Honkala and

*Research supported by the Academy of Finland Department of Mathematics, University of Turku, FIN-20014 Turku, Finland, email: juha.honkala@utu.fi
and Turku Centre for Computer Science (TUCS), Lemminkäisenkatu 14, FIN-20520 Turku, Finland

Kuich [4]). A power series $r \in A \ll \Delta^* \gg$ is called an *E0L* (resp. *EP0L*) *power series* if there are a finite alphabet Σ and a 0L (resp. P0L) power series $s \in A \ll \Sigma^* \gg$ such that

$$r = s \odot \mathrm{char}(\Delta^*).$$

Finally, a power series $r \in A \ll \Delta^* \gg$ is called a *C0L power series* if there exist a finite alphabet Σ, a 0L power series $s \in A \ll \Sigma^* \gg$ and a coding $g : A < \Sigma^* > \longrightarrow A < \Delta^* >$ such that

$$r = g(s).$$

If $A = \mathbf{B}$ where $\mathbf{B} = \{0, 1\}$ is the Boolean semiring, $r \in \mathbf{B} \ll \Sigma^* \gg$ is a 0L (resp. P0L, E0L, EP0L, C0L) power series if and only if the support of r is a 0L (resp. P0L, E0L, EP0L, C0L) language. (Here the empty set is regarded as a 0L (resp. P0L, E0L, EP0L, C0L) language.)

In order to generalize the E0L=C0L theorem for formal power series it is useful to consider separately three parts of the result corresponding to different steps in its proof (see Rozenberg and Salomaa [5]; recall also that here two languages are regarded as equal if they contain the same nonempty words.)

Theorem 1 *Every C0L language is an E0L language.*

Theorem 2 *Every E0L language is an EP0L language.*

Theorem 3 *Every EP0L language is a C0L language.*

In the sequel we will generalize Theorems 1 and 3 for quasiregular power series over any commutative ω-continuous semiring A. To generalize Theorem 2 we have to introduce an additional condition. As a consequence we obtain a power series generalization of the E0L=C0L theorem.

2 C0L power series are E0L power series

In this section we prove a power series generalization of Theorem 1.

Theorem 4 *Suppose $r \in A \ll \Delta^* \gg$ is a quasiregular C0L power series. Then r is an E0L power series.*

Proof. Suppose

$$r = \sum_{n=0}^{\infty} agh^n(w)$$

where $h : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ is a semialgebra morphism, $g : A < \Sigma^* > \longrightarrow A < \Delta^* >$ is a coding, $a \in A$ and $w \in \Sigma^*$. Without restriction we assume that $\Sigma \cap \Delta = \emptyset$. Extend g and h to semialgebra morphisms $g, h : A < (\Sigma \cup \Delta)^* > \longrightarrow A < (\Sigma \cup \Delta)^* >$ by $g(x) = h(x) = 0$ if $x \in \Delta$. Next, choose a new letter $\$ \notin \Sigma \cup \Delta$ and define the semialgebra morphism $f : A < (\Sigma \cup \Delta \cup \$)^* > \longrightarrow A < (\Sigma \cup \Delta \cup \$)^* >$ by

$$f(x) = \$h(x) + g(x), \qquad f(\$) = \varepsilon,$$

$x \in \Sigma \cup \Delta$. We claim that there exist polynomials $r_n, p_n \in A < (\Sigma \cup \Delta \cup \$)^* >$, $n \geq 1$, such that

$$f^n(w) = r_n + gh^{n-1}(w) + p_n \tag{2}$$

and

$$\mathrm{proj}_{\Sigma\cup\Delta}(r_n) = h^n(w), \;\; p_n \odot \mathrm{char}((\Sigma \cup \$)^*) = 0, \;\; p_n \odot \mathrm{char}(\Delta^*) = 0 \tag{3}$$

if $n \geq 1$. (Here $\mathrm{proj}_{\Sigma\cup\Delta} : A < (\Sigma \cup \Delta \cup \$)^* > \longrightarrow A < (\Sigma \cup \Delta)^* >$ is the projection mapping \$ into ε and x into itself if $x \in \Sigma \cup \Delta$.) Clearly, there exist $r_1, p_1 \in A < (\Sigma \cup \Delta \cup \$)^* >$ such that (2) and (3) hold for $n = 1$. Suppose then that (2) and (3) hold for $n \geq 1$. Then

$$f^{n+1}(w) = f(r_n + gh^{n-1}(w) + p_n) = f(h^n(w)) = r_{n+1} + gh^n(w) + p_{n+1}$$

for suitable $r_{n+1}, p_{n+1} \in A < (\Sigma \cup \Delta \cup \$)^* >$ satisfying

$$\mathrm{proj}_{\Sigma\cup\Delta}(r_{n+1}) = h^{n+1}(w),$$

$$p_{n+1} \odot \mathrm{char}((\Sigma \cup \$)^*) = 0, \;\; p_{n+1} \odot \mathrm{char}(\Delta^*) = 0.$$

This concludes the proof of the existence of the polynomials r_n, p_n, $n \geq 1$.

Now, because

$$\sum_{n=0}^{\infty} af^n(w) \odot \mathrm{char}(\Delta^*) =$$

$$aw \odot \mathrm{char}(\Delta^*) + a\sum_{n=1}^{\infty}(r_n + gh^{n-1}(w) + p_n) \odot \mathrm{char}(\Delta^*) =$$

$$\sum_{n=0}^{\infty} agh^n(w) = r,$$

r is a E0L power series. □

3 E0L power series satisfying the ε-condition

In this section we generalize Theorem 2 for E0L power series satisfying the ε-condition. Suppose

$$r = \sum_{n=0}^{\infty} ag^n(w) \odot \mathrm{char}(\Delta^*)$$

is an E0L power series where $g : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ is a semialgebra morphism, $a \in A$, $w \in \Sigma^*$ and $\Delta \subseteq \Sigma$. We say that r satisfies the *ε-condition* if

$$(g(c), \varepsilon) = (g^n(c), \varepsilon)$$

for all $n \geq 1$, $c \in \Sigma$.

Theorem 5 *Suppose $r \in A \ll \Delta^* \gg$ is a quasiregular E0L power series satisfying the ε-condition. Then r is an EP0L power series.*

Proof. Suppose

$$r = \sum_{n=0}^{\infty} ag^n(w) \odot \mathrm{char}(\Delta^*)$$

where $g : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ is a semialgebra morphism, $a \in A$, $w \in \Sigma^*$ and $\Delta \subseteq \Sigma$. Define the semialgebra morphism $\beta : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ by $\beta(c) = (g(c), \varepsilon)\varepsilon$ for $c \in \Sigma$. Then we have $\beta(v) = (g(v), \varepsilon)\varepsilon$ for $v \in \Sigma^*$. Let $\overline{\Sigma} = \{\overline{c} \mid c \in \Sigma\}$ be a new alphabet. Define the mapping $\phi : A < \Sigma^* > \longrightarrow A < (\Sigma \cup \overline{\Sigma})^* >$ by

$$\phi(\varepsilon) = 0,$$

$$\phi(c_1 \ldots c_m) = c_1 \ldots c_m + [\beta(c_1) + \overline{c}_1] \ldots [\beta(c_m) + \overline{c}_m] - \overline{c}_1 \ldots \overline{c}_m - \beta(c_1 \ldots c_m)$$

if $m \geq 1$ and $c_1, \ldots, c_m \in \Sigma$, and

$$\phi(P) = \sum (P, w)\phi(w)$$

if $P \in A < \Sigma^* >$. (Here A is not a ring but the meaning of the subtraction above should be clear.) Next, define the propagating semialgebra morphism $h : A < (\Sigma \cup \overline{\Sigma})^* > \longrightarrow A < (\Sigma \cup \overline{\Sigma})^* >$ by

$$h(c) = h(\overline{c}) = \phi(g(c))$$

for $c \in \Sigma$. Finally, define the semialgebra morphism $\pi : A < (\Sigma \cup \overline{\Sigma})^* > \longrightarrow A < \Delta^* >$ by $\pi(c) = c$ if $c \in \Delta$ and $\pi(c) = 0$ if $c \notin \Delta$.

Now, we claim that

$$\pi h^n(c) + \beta(c) = \pi g^n(c), \tag{4}$$

$$\pi h^n(\overline{c}) + \beta(c) = \pi g^n(c) \tag{5}$$

and

$$\pi h^n(\phi(v)) + \beta(v) = \pi g^n(v) \tag{6}$$

for $c \in \Sigma$, $v \in \Sigma^+$ and $n \geq 1$. First, it is easy to see that (4) and (5) hold if $n = 1$. Suppose (4) and (5) hold for $n \geq 1$. Let $v = c_1 \ldots c_m$ where $m \geq 1$ and $c_1, \ldots, c_m \in \Sigma$. Then

$$\pi h^n(\phi(v)) + \beta(v) = \pi h^n(c_1 \ldots c_m) + \pi[\beta(c_1) + h^n(\overline{c}_1)] \ldots$$

$$\pi[\beta(c_m) + h^n(\overline{c}_m)] - \pi h^n(\overline{c}_1 \ldots \overline{c}_m) = \pi g^n(c_1 \ldots c_m).$$

Next, we have

$$g(c) = \beta(c) + \sum_{u \neq \varepsilon} (g(c), u)u.$$

Because $\beta(c) = (g(c), \varepsilon)\varepsilon = (g^2(c), \varepsilon)\varepsilon = \beta(g(c))$, we obtain

$$\beta(c) = \beta(c) + \sum_{u \neq \varepsilon} (g(c), u)\beta(u).$$

Hence

$$\pi h^{n+1}(c) + \beta(c) = \pi h^n(h(c)) + \beta(c) + \sum_{u \neq \varepsilon}(g(c), u)\beta(u) =$$

$$\pi h^n(\sum_{u \neq \varepsilon}(g(c), u)\phi(u)) + \beta(c) + \sum_{u \neq \varepsilon}(g(c), u)\beta(u) =$$

$$\pi g^n(\sum_{u \neq \varepsilon}(g(c), u)u) + \beta(c) = \pi g^n(g(c)) = \pi g^{n+1}(c).$$

Therefore (4) holds if n is replaced by $n+1$. A similar argument shows that (5) holds if n is replaced by $n+1$. This proves (4),(5) and (6) for all $n \geq 1$.

Let now \$ be a new letter and extend h and π by $h(\$) = \phi(w)$, $\pi(\$) = 0$. Then the extended h is propagating and

$$r = \sum_{n=0}^{\infty} a\pi g^n(w) = a\pi(w) + \sum_{n=1}^{\infty} a\pi h^n(\phi(w)) = \sum_{n=0}^{\infty} a\pi h^n(\$),$$

where we have used the fact that $a\beta(w) = 0$. Hence r is an EP0L power series. □

4 EP0L power series are C0L power series

To generalize Theorem 3 we need two lemmas.

Lemma 1 *If $a \in A$ and $w \in \Sigma^*$ is a nonempty word, the monomial aw is a C0L power series.*

Proof. Define the semialgebra morphism $h : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ by $h(c) = 0$ for all $c \in \Sigma$. Then

$$aw = \sum_{n=0}^{\infty} ah^n(w)$$

is a 0L power series. Hence aw is also a C0L power series. □

Note that the proof of Lemma 1, although very simple, is completely different than the language-theoretic proof that $\{w\}$ is a 0L language. In fact, the use of 0-images is unavoidable in Lemma 1. For example, if $\sigma \in \Sigma$, $\sigma \in \mathbf{N} \ll \Sigma^* \gg$ is not a 0-free C0L power series although it clearly is a 0-free EP0L power series.

Lemma 2 *If $r_1, \ldots, r_t \in A \ll \Delta^* \gg$ are quasiregular C0L power series, so is $r_1 + \ldots + r_t$.*

Proof. It suffices to consider the case $t = 2$. Let

$$r_j = \sum_{n=0}^{\infty} g_j h_j^n(a_j w_j)$$

where $h_j : A < \Sigma_j^* > \longrightarrow A < \Sigma_j^* >$ is a semialgebra morphism, $g_j : A < \Sigma_j^* > \longrightarrow A < \Delta^* >$ is a coding, $a_j \in A$ and $w_j \in \Sigma_j^*$, $j = 1, 2$. Without restriction we

suppose that $a_1 \neq 0$ and $\Sigma_1 \cap \Sigma_2 = \emptyset$. Denote $k = |w_1|$ and let $\$_1, \ldots, \$_k$ be new letters. Let h be the common extension of h_1 and h_2 satisfying

$$h(\$_1) = h_1(a_1 w_1) + a_2 w_2, \qquad h(\$_2) = \ldots = h(\$_k) = \varepsilon.$$

Finally, let g be the common extension of g_1 and g_2 satisfying

$$g(\$_1 \ldots \$_k) = a_1 g_1(w_1).$$

(The existence of g is clear if $a_1 g_1(w_1) \neq 0$ or $k \neq 1$. If $a_1 g_1(w_1) = 0$ and $k = 1$, we have to increase the value of k by 1.) Then

$$\sum_{n=0}^{\infty} gh^n(\$_1 \ldots \$_k) = g(\$_1 \ldots \$_k) + \sum_{n=0}^{\infty} gh^n(h_1(a_1 w_1) + a_2 w_2) =$$

$$a_1 g_1(w_1) + \sum_{n=0}^{\infty} g_1 h_1^{n+1}(a_1 w_1) + \sum_{n=0}^{\infty} g_2 h_2^n(a_2 w_2) = r_1 + r_2$$

showing that $r_1 + r_2$ is indeed a C0L power series. □

Theorem 6 *If $r \in A \ll \Delta^* \gg$ is a quasiregular EP0L power series then r is a C0L power series.*

Proof. Suppose

$$r = \sum_{n=0}^{\infty} ah^n(w) \odot \mathrm{char}(\Delta^*)$$

where $h : A < \Sigma^* > \longrightarrow A < \Sigma^* >$ is a propagating semialgebra morphism, $a \in A$ and $w \in \Sigma^*$. Without restriction we assume that $a = 1$.

For a letter $c \in \Sigma$, the *existential spectrum* of c, denoted by $\mathrm{espec}(c)$, is defined by

$$\mathrm{espec}(c) = \{n \geq 0 \mid h^n(c) \odot \mathrm{char}(\Delta^*) \neq 0\}.$$

If $c \in \Sigma$, the set $\mathrm{espec}(c)$ is ultimately periodic, see Rozenberg and Salomaa [5,6]. (Here we use König's Lemma to avoid the difficulties caused by products equal to zero.) The threshold and period of $\mathrm{espec}(c)$ are denoted by $\mathrm{thres}(\mathrm{espec}(c))$ and $\mathrm{per}(\mathrm{espec}(c))$, respectively. If $\mathrm{espec}(c)$ is infinite, then c is called a *vital letter*. The set of vital letters of Σ is denoted by $\mathrm{vit}(\Sigma)$.

The *uniform period* associated to r is the smallest positive integer p such that
(i) for all $j \geq p$, if c is not a vital letter, then $h^j(c) \odot \mathrm{char}(\Delta^*) = 0$;
(ii) if c is a vital letter, then $p > \mathrm{thres}(\mathrm{espec}(c))$ and $\mathrm{per}(\mathrm{espec}(c))$ divides p.

Let $0 \leq k < p$ and denote

$$\Sigma_k = \{c \in \Sigma \mid p + k \in \mathrm{espec}(c)\}.$$

Define the propagating semialgebra morphism $g_k : A < \Sigma_k^* > \longrightarrow A < \Sigma_k^* >$ by

$$g_k(c) = h^p(c) \odot \mathrm{char}(\Sigma_k^*),$$

$c \in \Sigma_k$. Furthermore, define the propagating semialgebra morphism $g_{p+k} : A < \Sigma_k^* > \longrightarrow A < \Delta^* >$ by

$$g_{p+k}(c) = h^{p+k}(c) \odot \mathrm{char}(\Delta^*),$$

$c \in \Sigma_k$. Note that $g_{p+k}(c) \neq 0$ for all $c \in \Sigma_k$. We claim that

$$h^{p+k}(h^p)^n h^p(P) \odot \mathrm{char}(\Delta^*) = g_{p+k} g_k^n [h^p(P) \odot \mathrm{char}(\Sigma_k^*)] \tag{7}$$

for any $n \geq 0$ and $P \in A < \Sigma^* >$. First,

$$h^{p+k} h^p(P) \odot \mathrm{char}(\Delta^*) = h^{p+k}[h^p(P) \odot \mathrm{char}(\Sigma_k^*)] \odot \mathrm{char}(\Delta^*) +$$

$$h^{p+k}[h^p(P) \odot \mathrm{char}(\Sigma^+ - \Sigma_k^*)] \odot \mathrm{char}(\Delta^*) =$$

$$h^{p+k}[h^p(P) \odot \mathrm{char}(\Sigma_k^*)] \odot \mathrm{char}(\Delta^*) = g_{p+k}[h^p(P) \odot \mathrm{char}(\Sigma_k^*)].$$

Hence (7) holds if $n = 0$. Suppose then that (7) holds for $n \geq 0$. Then

$$h^{p+k}(h^p)^{n+1} h^p(P) \odot \mathrm{char}(\Delta^*) = h^{p+k}(h^p)^n h^p[h^p(P) \odot \mathrm{char}(\Sigma_k^*)] \odot \mathrm{char}(\Delta^*) =$$

$$g_{p+k} g_k^n [h^p[h^p(P) \odot \mathrm{char}(\Sigma_k^*)] \odot \mathrm{char}(\Sigma_k^*)] = g_{p+k} g_k^{n+1}[h^p(P) \odot \mathrm{char}(\Sigma_k^*)].$$

Consequently, (7) holds for all $n \geq 0$. Therefore

$$r = \sum_{n=0}^{2p-1} h^n(w) \odot \mathrm{char}(\Delta^*) + \sum_{k=0}^{p-1} \sum_{n=0}^{\infty} h^{p+k}(h^p)^n h^p(w) \odot \mathrm{char}(\Delta^*) =$$

$$\sum_{n=0}^{2p-1} h^n(w) \odot \mathrm{char}(\Delta^*) + \sum_{k=0}^{p-1} \sum_{n=0}^{\infty} g_{p+k} g_k^n [h^p(w) \odot \mathrm{char}(\Sigma_k^*)].$$

By Lemmas 1 and 2 it suffices to prove that the series

$$s_{k,y} = \sum_{n=0}^{\infty} g_{p+k} g_k^n(y)$$

is a C0L power series if $0 \leq k < p$ and $y \in \Sigma_k^+$. For the proof fix k and y.

Next, choose nonzero polynomials P_x, $x \in \Sigma_k$, and a coding α such that

$$\alpha(P_x) = g_{p+k}(x),$$

no two of P_x, $x \in \Sigma_k$ contain a common variable, each variable of P_x has a unique occurrence in P_x and every nonzero coefficient of P_x equals 1, $x \in \Sigma_k$. Denote $P_\varepsilon = \varepsilon$ and $P_v = P_{v_1} P_{v_2} \ldots P_{v_m}$ if $m \geq 1$, $v = v_1 \ldots v_m$ and $v_i \in \Sigma_k$ for $1 \leq i \leq m$. By our choice of P_x, there exists a semialgebra morphism f such that

$$f(P_x) = \sum_{v \in \Sigma_k^*} (g_k(x), v) P_v,$$

if $x \in \Sigma_k$. Then

$$f(P_u) = \sum_{v \in \Sigma_k^*} (g_k(u), v) P_v \tag{8}$$

for any nonempty word $u \in \Sigma_k^*$. Indeed, (8) holds if $u \in \Sigma_k$ and, if (8) holds for $u \in \Sigma_k^+$ we have

$$f(P_{ux}) = f(P_u) f(P_x) =$$

$$\sum_{v_1 \in \Sigma_k^*} (g_k(u), v_1) P_{v_1} \cdot \sum_{v_2 \in \Sigma_k^*} (g_k(x), v_2) P_{v_2} = \sum_{v \in \Sigma_k^*} (g_k(ux), v) P_v$$

where $x \in \Sigma_k$.

Next, we claim that

$$f^n(P_y) = \sum_{v \in \Sigma_k^*} (g_k^n(y), v) P_v \tag{9}$$

for $n \geq 1$. First, if $n = 1$, (9) follows from (8). Suppose that (9) holds for $n \geq 1$. Then

$$f^{n+1}(P_y) = \sum_{u \in \Sigma_k^*} (g_k^n(y), u) f(P_u) =$$

$$\sum_{u \in \Sigma_k^*} (g_k^n(y), u) \sum_{v \in \Sigma_k^*} (g_k(u), v) P_v = \sum_{v \in \Sigma_k^*} (g_k^{n+1}(y), v) P_v.$$

Hence (9) holds for all $n \geq 1$. Therefore

$$\sum_{n=0}^{\infty} \alpha f^n(P_y) = g_{p+k}(y) + \sum_{n=1}^{\infty} \sum_{v \in \Sigma_k^*} (g_k^n(y), v) g_{p+k}(v) = \sum_{n=0}^{\infty} g_{p+k} g_k^n(y) = s_{k,y}.$$

This shows that $s_{k,y}$ is indeed a C0L power series. □

Now, Theorems 5 and 6 imply the following result.

Theorem 7 *If $r \in A \ll \Delta^* \gg$ is a quasiregular E0L power series satisfying the ε-condition, then r is a C0L power series.*

The necessity of the ε-condition in Theorem 7 is an open problem.

References

[1] A. Ehrenfeucht and G. Rozenberg, The equality of E0L languages and codings of 0L languages, *Intern. J. Comput. Math.* 4 (1974) 95-104.

[2] J. Honkala, On Lindenmayerian series in complete semirings, in: G. Rozenberg and A. Salomaa, eds., *Developments in Language Theory* (World Scientific, Singapore, 1994) 179-192.

[3] J. Honkala, On morphically generated formal power series, *RAIRO Theoretical Inform. and Appl.* **29** (1995) 105-127.

[4] J. Honkala and W. Kuich, On Lindenmayerian algebraic power series, *Theoret. Comput. Sci.* **183** (1997) 113-142.

[5] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L Systems* (Academic Press, New York, 1980).

[6] G. Rozenberg and A. Salomaa (eds.): *Handbook of Formal Languages, Vol. 1-3* (Springer, Berlin, 1997).

Received April, 2000

Acta Cybernetica 14 (2000) 607-617.

On commutative asynchronous nondeterministic automata *

B. Imreh † M. Ito ‡ A. Pukler §

Abstract

In this paper, we deal with nondeterministic automata, in particular, commutative asynchronous ones. Our goal is to give their isomorphic representation under the serial product or equivalently, under the α_0-product. It turns out that this class does not contain any finite isomorphically complete system with respect to the α_0-product. On the other hand, we present an isomorphically complete system for this class which consists of one monotone nondeterministic automaton of three elements.

1 Introduction

The study of the compositions of nondeterministic (n.d. for short) automata was initiated in the work [3], where the isomorphically complete systems with respect to the general product were characterized. In [4] it is proved that the general and cube products of n.d. automata are equivalent regarding the isomorphically complete systems. A further result on this line can be found in [7], where the isomorphically complete systems of n.d. automata with respect to the α_0-product are characterized.

In this work, a particular class of n.d. automata, the class of all commutative asynchronous n.d. automata, is studied. The isomorphic representation of the deterministic commutative asynchronous automata was studied in [8], where it turned out that every commutative asynchronous automaton can be embedded into a quasi-direct power of a suitable two-state commutative asynchronous automaton. We show here that this is not valid for the n.d. case, and what is more, it is not valid neither under the stronger α_0-product. On the other hand, it is proved that

*This work has been supported by the Japanese Ministry of Education, Mombusho International Scientific Research Program, Joint Research 10044098, the Hungarian National Foundation for Scientific Research, Grant T030143, and the Ministry of Culture and Education of Hungary, Grant FKFP 0704/1997.

†Department of Informatics, University of Szeged, Árpád tér 2, H-6720 Szeged, Hungary

‡Department of Mathematics, Faculty of Science, Kyoto Sangyo University, Kyoto 603-8555, Japan

§Department of Computer Science, István Széchenyi College, Hédervári út 3., H-9026 Győr, Hungary

every commutative asynchronous n.d. automaton can be embedded into a suitable α_0-power of a monotone n.d. automaton having three states.

The paper is organized as follows. First, in Section 2, we recall a few notions and notation and present some basic results necessary in the sequal. In Section 3, it is shown that there is no finite system of commutative asynchronous n.d. automata which is isomorphically complete for the class under consideration with respect to the α_0-product. Then we look for a finite isomorphically complete system in a larger class, namely, in the class of monotone n.d. automata, and we prove that every commutative asynchronous n.d. automaton can be embedded into a suitable α_0-power of a monotone n.d. automaton of three states.

2 Preliminaries

An automaton can be defined as an algebra $\mathbf{A} = (A, X)$ in which every input sign x is realized as a unary operation $x^{\mathbf{A}} : A \to A$. Then the n.d. automata can be introduced as generalized automata in which the unary operations are replaced by binary relations. Therefore, by *n.d. automaton* we mean a system $\mathbf{A} = (A, X)$, where A is a finite nonvoid set of *states*, X is a finite nonempty set of *input signs*, and every $x \in X$ is realized as a binary relation $x^{\mathbf{A}} (\subseteq A \times A)$ on A. For any $a \in A$ and $x \in X$, let $ax^{\mathbf{A}} = \{c : c \in A \text{ and } (a, c) \in x^{\mathbf{A}}\}$, *i.e.*, $ax^{\mathbf{A}}$ is the set of states into which $\mathbf{A}$ may enter from a by receiving the input sign x. For any $C \subseteq A$ and $x \in X$, we set $Cx^{\mathbf{A}} = \cup\{ax^{\mathbf{A}} : a \in C\}$. For a word $w \in X^*$, $Cw^{\mathbf{A}}$ can be defined inductively as follows:

(1) $C\varepsilon^{\mathbf{A}} = C$,

(2) $Cw^{\mathbf{A}} = (Cv^{\mathbf{A}})x^{\mathbf{A}}$ for $w = vx$, $v \in X^*$ and $x \in X$,

where ε denotes the empty word of X^*. An n.d. automaton is called *complete* if $ax^{\mathbf{A}} \neq \emptyset$, for all $a \in A$ and $x \in X$. Throughout this paper, by n.d. automaton we always mean a complete n.d. automaton. Let $\mathbf{A} = (A, X)$ be an n.d. automaton and $B \subseteq A$. Then one can define a subautomaton $\mathbf{B} = (B, X)$ of $\mathbf{A}$ by the realizations $x^{\mathbf{B}} = x^{\mathbf{A}} \cap (B \times B)$, $x \in X$. We note that a subautomaton of a complete n.d. automaton is not necessarily complete. Let $\mathbf{A} = (A, X)$ and $\mathbf{B} = (B, X)$ be two n.d. automata and μ a mapping of A onto B. The mapping μ is called a *homomorphism* of $\mathbf{A}$ onto $\mathbf{B}$ if $ax^{\mathbf{A}}\mu = a\mu x^{\mathbf{B}}$ is valid, for all $a \in A$ and $x \in X$. In this case, it is said that $\mathbf{B}$ is a *homomorphic image* of $\mathbf{A}$. If the homorphism μ is a one-to-one mapping, then it is called an *isomorphism* and in this case, it is said that $\mathbf{A}$ is *isomorphic* to $\mathbf{B}$. Furthermore, if $\mathbf{B}$ is isomorphic to some subautomaton of $\mathbf{A}$, then it is said that $\mathbf{B}$ *can be embedded* into $\mathbf{A}$.

Let $\mathbf{A} = (A, X)$ be an n.d. automaton and Θ an equivalence relation on A. For every $a \in A$, let us denote by $\Theta(a)$ the equivalence class containing a, or equivalently, the set of the elements which are equivalent to a. Then we can construct a factor n.d. automaton $\mathbf{A}/\Theta$ as follows. For any $\Theta(a) \in A/\Theta$ and $x \in X$, let

$\Theta(a)x^{\mathbf{A}/\Theta} = \{\Theta(b) : \Theta(b) \in A/\Theta$ and $\Theta(a)x^{\mathbf{A}} \cap \Theta(b) \neq \emptyset\}$. It is worth noting that $\mathbf{A}/\Theta$ is not a homomorphic image of $\mathbf{A}$ in general. In what follows, we shall use particular equivalence relations. To define them, let A be an arbitrary nonempty set and a, b its two different elements. Then the equivalence relation $\Theta(a, b)$ is defined as follows. For every $u, v \in A$,

$$u\Theta(a,b)v \text{ if and only if } \{a,b\} = \{u,v\} \text{ or } u = v.$$

An n.d. automaton $\mathbf{A} = (A, X)$ is called *commutative* if $a(xy)^{\mathbf{A}} = a(yx)^{\mathbf{A}}$ is valid, for every $a \in A$ and $x, y \in X$. By the definition of the commutativity, one can easily prove the following fact.

Lemma 1. *If an n.d. automaton* $\mathbf{A}$ *is commutative and* $\mathbf{B}$ *is a homomorphic image of* $\mathbf{A}$*, then* $\mathbf{B}$ *is commutative as well.*

An n.d. automaton $\mathbf{A} = (A, X)$ is called *asynchronous* if for every $a \in A$ and $x \in X$, $b \in ax^{\mathbf{A}}$ implies $bx^{\mathbf{A}} = \{b\}$. In particular, if $a \in ax^{\mathbf{A}}$, then $ax^{\mathbf{A}} = \{a\}$. Since we recall this property in more times, we express it by the following remark.

Remark 1. *If* $\mathbf{A} = (A, X)$ *is an asynchronous n.d. automaton and* $a \in ax^{\mathbf{A}}$ *for some* $a \in A$ *and* $x \in X$*, then* $ax^{\mathbf{A}} = \{a\}$.

¿From the definition of the asynchronous n.d. automata the following fact follows immediately.

Lemma 2. *If an n.d. automaton* $\mathbf{A}$ *is asynchronous and* $\mathbf{B}$ *is a homomorphic image of* $\mathbf{A}$*, then* $\mathbf{B}$ *is also asynchronous.*

We shall study the commutative asynchronous n.d. automata. Let us denote by $\mathcal{K}_{nd}$ the class of all commutative asynchronous automata. Then, by Lemmas 1 and 2, we obtain the following observation.

Corollary 1. *If* $\mathbf{A} \in \mathcal{K}_{nd}$ *and* $\mathbf{B}$ *is a homomorphic image of* $\mathbf{A}$*, then* $\mathbf{B} \in \mathcal{K}_{nd}$.

An important property of the n.d. automata in $\mathcal{K}_{nd}$ is presented by the next assertion.

Lemma 3. *If* $\mathbf{A} = (A, X) \in \mathcal{K}_{nd}$*, then its transition graph does not contain any directed cycle different from loop.*

Proof. Let $a \in aq^{\mathbf{A}}$ for some $a \in A$, $q \in X^+$ and let q be a minimum-length word with this property. Now, let us suppose that $|q| > 1$. Then $q = xp$ for some $x \in X$ and $p \in X^+$. By the commutativity of $\mathbf{A}$, $a \in ap^{\mathbf{A}}x^{\mathbf{A}}$. Therefore, there exists a state b such that $b \in ap^{\mathbf{A}}$ and $a \in bx^{\mathbf{A}}$. Let us distinguish now the following two cases depending on b.

Case 1. $a = b$. Then $a \in ax^{\mathbf{A}}$, and by Remark 1, $ax^{\mathbf{A}} = \{a\}$ contradicting the minimality of the word q.

Case 2. $a \neq b$. In this case, $a \in bx^{\mathbf{A}}$. Since $\mathbf{A}$ is an asynchronous n.d. automaton, $a \in bx^{\mathbf{A}}$ implies $ax^{\mathbf{A}} = \{a\}$ which contradicts the minimality of q again.

Consequently, the transition graph of $\mathbf{A}$ does not contain any directed cycle different from loop.

Let $\mathbf{A} = (A, X)$ be an arbitrary n.d. automaton. Let us define the *reachability relation* as follows. For a couple of states a, b, it is said that b is *reachable* from a, denoted by $a \leq b$, if there exists a word w such that $b \in aw^{\mathbf{A}}$. Obviously, that this relation is reflexive and transitive. In particular, if $\mathbf{A} \in \mathcal{K}_{nd}$, then by Lemma 3, this relation is antisymmetric, and thus, it is a partial ordering on A. Hence, we have the following statement.

Corollary 2. *For every* $\mathbf{A} \in \mathcal{K}_{nd}$, $(A, \leq)$ *is a partially ordered set.*

The more general composition, the general product of automata was introduced by V. M. Gluskov in [6]. This composition is extended to n.d. automata in [3]. Now, we recall this definition.

Let us consider the n.d. automata $\mathbf{A} = (X, A)$, $\mathbf{A}_j = (X_j, A_j)$, $j = 1, \ldots, k$, and let Φ be a family of mappings below

$$\varphi_j : A_1 \times \ldots \times A_k \times X \to X_j, \quad j = 1, \ldots, k.$$

It is said that $\mathbf{A}$ is the *general product* of $\mathbf{A}_j$ with respect to Φ if the following conditions are satisfied:

(1) $A = \prod_{j=1}^{k} A_j$,

(2) for any $(a_1, \ldots, a_k) \in \prod_{j=1}^{k} A_j$, and $x \in X$,

$$(a_1, \ldots, a_k)x^{\mathbf{A}} = a_1 x_1^{\mathbf{A}_1} \times \cdots \times a_k x_k^{\mathbf{A}_k},$$

where $x_j = \varphi_j(a_1, \ldots, a_k, x)$ for all $j \in \{1, \ldots, k\}$.

For the general product above we use the notation

$$\mathbf{A} = \prod_{j=1}^{k} \mathbf{A}_j(X, \Phi) .$$

The mappings φ_j, $j = 1, \ldots, k$ are called *feedback functions.*

Let $\mathcal{K}$ be a system of n.d. automata. $\mathcal{K}$ is *isomorphically complete* with respect to the general product if for any n.d. automaton $\mathbf{A}$, there exist automata $\mathbf{A}_j \in \mathcal{K}$, $j = 1, \ldots, k$, such that $\mathbf{A}$ can be embedded into a general product of $\mathbf{A}_j$, $j = 1, \ldots, k$.

Different compositions of automata can be obtain as a special case of the general product by using particular feedback functions. One of them is the serial composition of automata, where the automata form a chain and the input sign of a given

automaton of the chain depends on the input sign received by the composition and the current states of the previos automata in the chain. The formal definition can be given as follows.

Let $\mathbf{A}_j = (A_j, X_j)$, $j = 1, \ldots, k$ be arbitrary n.d. automata. Moreover, let X be a finite nonvoid set and Φ is a family of mappings:

$$\varphi_j : A_1 \times \cdots \times A_{j-1} \times X \to X_j, \quad j = 1, \ldots, k.$$

An n.d. automaton $\mathbf{A} = (A, X)$ is called the *serial product* or *α_0-product* of the n.d. automata considered, if $A = \prod_{j=1}^{k} A_j$ and for every $(a_1, \ldots, a_k) \in \prod_{j=1}^{k} A_j$ and $x \in X$,

$$(a_1, \ldots, a_k)x^{\mathbf{A}} = a_1 x_1^{\mathbf{A}_1} \times \cdots \times a_k x_k^{\mathbf{A}_k}$$

is valid, where $x_j = \varphi_j(a_1, \ldots, a_{j-1}, x)$, $j = 1, \ldots, k$. If the component n.d. automata $\mathbf{A}_j$ are equal, say $\mathbf{A}_j = \mathbf{B}$, $j = 1, \ldots, k$, then it is said that the α_0-product $\mathbf{A}$ is an *α_0-power* of $\mathbf{B}$. In particular, if the mappings φ_j, $j = 1, \ldots, k$ are independent of the states, *i.e.*, they have the forms $\varphi_j : X \to X_j$, $j = 1, \ldots, k$, then $\mathbf{A}$ is called the *quasi-direct product* of the n.d. automata under consideration.

It has to be mentioned here that as generalizations of the serial product of automata a family of products, the α_i-product, $i = 0, 1, \ldots$, was introduced in [1] for the deterministic case and some nice results concerning the α_i-products can be found in the monography [2].

By the definition of the α_0-product, one can easily prove the following statement.

Lemma 4. *If for every t, $t = 1, \ldots, n$, the n.d. automata $\mathbf{A}_t$ can be embedded into an α_0-product of n.d. automata $\mathbf{A}_{tj}$, $j = 1, \ldots, k_t$, then any α_0-product of the n.d. automata $\mathbf{A}_t$, $t = 1, \ldots, n$ can be embedded into an α_0-product of the n.d. automata $\mathbf{A}_{tj}$, $t = 1, \ldots, n$; $j = 1, \ldots, k_t$.*

Finally, we define the notion of isomorphically complete systems of n.d. automata for the α_0-product. For this purpose, let $\mathcal{K}$ be an arbitrary class of n.d. automata. A system $\mathcal{M}$ of n.d. automata is called *isomorphically complete for $\mathcal{K}$ with respect to the α_0-product* if any n.d. automaton in $\mathcal{K}$ can be embedded into an α_0-product of n.d. automata in $\mathcal{M}$.

3 Isomorphic representation

In this section, the isomorphic representation of the automata in $\mathcal{K}_{nd}$ are studied. The next statement shows that contrary to the deterministic case, the class $\mathcal{K}_{nd}$ does not contain any finite isomorphically complete system for $\mathcal{K}_{nd}$ with respect to the general product.

Proposition 1. *There is no finite system $\mathcal{M} \subseteq \mathcal{K}_{nd}$ of n.d. automata which is isomorphically complete for $\mathcal{K}_{nd}$ with respect to the general product.*

Proof. In our proof we shall use some particular automata. Namely, for all $n \geq 3$, let us define the n.d. automaton $\mathbf{C}_n = (\{1,\ldots,n\},\{x_2,\ldots,x_{n-1}\})$ as follows. For every $i \in \{1,\ldots,n\}$ and $x_k \in \{x_2,\ldots,x_{n-1}\}$, let

$$ix_k^{\mathbf{C}_n} = \begin{cases} \{k,k+1,\ldots,n\} & \text{if } i < k, \\ \{i\} & \text{otherwise.} \end{cases}$$

From the definition of $\mathbf{C}_n$ it follows that $\mathbf{C}_n$ is an asynchronous n.d. automaton. Now, we prove that $\mathbf{C}_n$ is commutative. For this reason, let $i \in \{1,\ldots,n\}$ and $x_j, x_k \in \{x_2,\ldots,x_{n-1}\}$ be arbitrary elements with $j \neq k$. Without loss of generality, we may suppose that $j < k$. Then, for the case $k \leq i$, we have that $ix_j^{\mathbf{C}_n}x_k^{\mathbf{C}_n} = \{i\} = ix_k^{\mathbf{C}_n}x_j^{\mathbf{C}_n}$. If $j \leq i < k$, then

$$ix_j^{\mathbf{C}_n}x_k^{\mathbf{C}_n} = \{i\}x_k^{\mathbf{C}_n} = \{k,k+1,\ldots,n\} = ix_k^{\mathbf{C}_n}x_j^{\mathbf{C}_n}.$$

Finally, if $i < j$, then

$$ix_j^{\mathbf{C}_n}x_k^{\mathbf{C}_n} = \{j,j+1,\ldots,n\}x_k^{\mathbf{C}_n} = \{k,k+1,\ldots,n\} = ix_k^{\mathbf{C}_n} = ix_k^{\mathbf{C}_n}x_j^{\mathbf{C}_n}.$$

These observations lead to the commutativity of $\mathbf{C}_n$. Consequently, $\mathbf{C}_n \in \mathcal{K}_{nd}$, for all integer $n \geq 3$.

For proving the statement, contrary, let us suppose that $\mathcal{M} \subseteq \mathcal{K}_{nd}$ is a finite isomorphically complete system for $\mathcal{K}_{nd}$ with respect to the general product. Then there exists an integer n such that $|A| < n$ is valid for every n.d. automaton $\mathbf{A} = (A,X) \in \mathcal{M}$. Since $\mathcal{M}$ is an isomorphically complete system for $\mathcal{K}_{nd}$ with respect to the general product and $\mathbf{C}_n \in \mathcal{K}_{nd}$, there are n.d. automata $\mathbf{A}_t \in \mathcal{M}$, $t = 1,\ldots,k$ such that $\mathbf{C}_n$ can be embedded into a general product $\prod_{t=1}^{k}\mathbf{A}_t(\{x_2,\ldots,x_{n-1}\},\Phi)$. Let μ denote a suitable isomorphism of $\mathbf{C}_n$ into the general product considered and let

$$i\mu = (a_{i1},a_{i2},\ldots,a_{ik}), \quad i = 1,\ldots,n.$$

Denote by r an integer for which $a_{n-1,r} \neq a_{nr}$. Such an integer exists. We shall now prove that the states $a_{1r},a_{2r},\ldots,a_{nr}$ are pairwise different. First, let us consider the state $a_{n-2,r}$. Since μ is an isomorphism, $a_{n-2,r}\varphi_r(a_{n-2,1},\ldots,a_{n-2,k},x_{n-1})^{\mathbf{A}_r} \cap \{a_{n-1,r},a_{n,r}\} = \{a_{n-1,r},a_{n,r}\}$. Thus, by $a_{n-1,r} \neq a_{nr}$ and Remark 1, we obtain that $a_{n-2,r} \notin \{a_{n-1,r},a_{n,r}\}$. Therefore, $a_{n-2,r},a_{n-1,r},a_{nr}$ are pairwise different. Now, if for some integer $2 \leq i \leq n-2$, the elements $a_{n-i,r},a_{n-i+1,r},\ldots,a_{nr}$ are pairwise different, then in a similar way as above, we get that

$$a_{n-i-1,r}\varphi_r(a_{n-i-1,1},\ldots,a_{n-i-1,k},x_{n-i})^{\mathbf{A}_r} \supseteq \{a_{n-i,r},a_{n-i+1,r},\ldots,a_{nr}\}.$$

This inclusion and Remark 1 yield that $a_{n-i-1,r} \notin \{a_{n-i,r},\ldots,a_{nr}\}$, and therefore, the elements $a_{n-i-1,r},a_{n-i,r},\ldots,a_{nr}$ are pairwise different. From these observations it follows immediately that the elements $a_{1r},a_{2r},\ldots,a_{nr}$ are pairwise different. This implies that $n \leq |A_r|$ contradicting the definition of n. Consequently,

there is no finite system $\mathcal{M} \subseteq \mathcal{K}_{nd}$ so that $\mathcal{M}$ is isomorphically complete for $\mathcal{K}_{nd}$ with respect to the general product.

Since the α_0-product is a particular case of the general product, we get the following observation.

Corollary 3. *There is no finite system $\mathcal{M} \subseteq \mathcal{K}_{nd}$ of n.d. automata which is isomorphically complete for $\mathcal{K}_{nd}$ with respect to the α_0-product.*

Corollary 3 shows that there is a significant difference between the isomorphic representations of deterministic and n.d. automata. The class of all deterministic automata denoted by $\mathcal{L}_d$ does not contain any finite system which is isomorphically complete for $\mathcal{L}_d$ with respect to the α_0-product (see [5]). On the other hand, the class of all n.d. automata denoted by $\mathcal{L}_{nd}$ contains finite isomorphically complete system for $\mathcal{L}_{nd}$ with respect to the α_0-product (*cf.* [7]). Therefore, this pair of classes is an example for the case when the deterministic class does not contain any finite isomorphically complete system while the n.d. class contains a finite isomorphically complete system with respect to the α_0-product. The pair of the classes of the commutative asynchronous deterministic and n.d. automata denoted by $\mathcal{K}_d$ and $\mathcal{K}_{nd}$, respectively, is an example for the opposite case. Indeed, in [8], it is proved that $\mathcal{K}_d$ contains finite isomorphically complete systems for $\mathcal{K}_d$ with respect to the quasi-direct product. Since the quasi-direct product is a particular case of the α_0-product, this result yields that this class contains some finite isomorphically complete systems for $\mathcal{K}_d$ with respect to the α_0-product. On the other hand, by Proposition 1, it is not valid for the class $\mathcal{K}_{nd}$. Consequently, the pair of classes $\mathcal{K}_d$ and $\mathcal{K}_{nd}$ is an example for the case when the determinstic class conatins a finite isomorphically system while the n.d. class does not do it.

Of course there are finite isomorphically complete systems for $\mathcal{K}_{nd}$ with respect to the α_0-product, but they are not contained in $\mathcal{K}_{nd}$. Proposition 2 shows that there are finite isomorphically complete systems for $\mathcal{K}_{nd}$ with respect to the α_0-product such that they contain monotone n.d. automata in that sense that the transition graphs of these automata do not contain any directed cycle different from a loop. Moreover, it turns out that there exists such an isomorphically complete system for $\mathcal{K}_{nd}$ with respect to the α_0-product which consists of a monotone n.d. automaton having three states.

The n.d. automaton what we need is denoted by $\mathbf{B} = (\{0,1,2\}, \{x,y,u,v\})$ and it is defined as follows:

$$0x^{\mathbf{B}} = \{0,1,2\},\ ix^{\mathbf{B}} = \{i\},\ i = 1,2,$$
$$0y^{\mathbf{B}} = \{0,1\},\ iy^{\mathbf{B}} = \{i\},\ i = 1,2,$$
$$0u^{\mathbf{B}} = \{0,2\},\ iu^{\mathbf{B}} = \{i\},\ i = 1,2,$$
$$iv^{\mathbf{B}} = \{2\},\ i = 0,1,2.$$

It is easy to check that $\mathbf{B}$ is monotone, *i.e.*, its transition graph does not contain any directed cycle different from loop.

Proposition 2. *Any system $\mathcal{M}$, containig such an n.d. automaton* $\mathbf{A}$ *that* $\mathbf{B}$ *can be embedded into an α_0-product of* $\mathbf{A}$ *with a single factor, is isomorphically complete for $\mathcal{K}_{nd}$ with respect to the α_0-product.*

Proof. By Lemma 4, it is sufficient to prove that any n.d. automaton from $\mathcal{K}_{nd}$ can be embedded into a suitable α_0-power of $\mathbf{B}$.

We shall prove this statement by induction on the number of states of the n.d. automata. It is worth noting that for every positive integer n, $\mathcal{K}_{nd}$ contains automata having n states.

One can easily check that if $\mathbf{A} \in \mathcal{K}_{nd}$ and $|A| \leq 2$, then $\mathbf{A}$ can be embedded into an α_0-product of $\mathbf{B}$ with a single factor. Now, let $n \geq 2$ be an arbitrary integer and let us suppose that the statement is valid for every $\mathbf{A} \in \mathcal{K}_{nd}$ with $|A| \leq n$. Let us consider an arbitrary n.d. automaton $\mathbf{A} = (A, X) \in \mathcal{K}_{nd}$ with $|A| = n + 1$. Corollary 2 provides that the reachability relation is a partial ordering on the set A. Since $\mathbf{A}$ is finite, $(A, \leq)$ contains maximal elements. We distinguish two cases depending on the number of the maximal elements.

Case 1. The number of the maximal elements in $(A, \leq)$ is not less than 2. Then there are at least 2 maximal elements, which are denoted by c, d. Now, let us define the α_0-product $\mathbf{D} = \mathbf{A}/\Theta(c,d) \times \mathbf{B}(X, \Phi)$ as follows.

For every $z \in X$ and $a \in A \setminus \{c, d\}$, let

$$\varphi_1(z) = z$$

$$\varphi_2(\{a\}, z) = \begin{cases} y & \text{if } az^{\mathbf{A}} \cap \{c,d\} = \{c\}, \\ u & \text{if } az^{\mathbf{A}} \cap \{c,d\} = \{d\}, \\ x & \text{otherwise,} \end{cases}$$

$$\varphi_2(\{\{c,d\}\}, z) = x.$$

Let us define the mapping $\mu : A \to A/\Theta(c,d) \times \{0,1,2\}$ as follows:

$c\mu = (\{c,d\}, 1)$,
$d\mu = (\{c,d\}, 2)$,
$a\mu = (\{a\}, 0)$, for all $a \in A \setminus \{c,d\}$.

and let $S = \{(\{a\}, 0) : a \in A \setminus \{c,d\}\} \cup \{(\{c,d\}, 1), (\{c,d\}, 2)\}$.

We prove that μ is an isomorphism of $\mathbf{A}$ into the α_0-product $\mathbf{D}$, more precisely, $\mathbf{A}$ is isomorphic to the subautomaton of $\mathbf{D}$ which is determined by the subset S.

First, let $a \in A \setminus \{c,d\}$ and $z \in X$ be arbitrary state and input sign, respectively. If $az^{\mathbf{A}} \cap \{c,d\} = \emptyset$, then $az^{\mathbf{A}}\mu = a\mu z^{\mathbf{D}} \cap S = a\mu z^{\mathbf{S}}$ is obviously valid. If $az^{\mathbf{A}} \cap \{c,d\} \neq \emptyset$, then let us investigate separately the three cases corresponding to the elements of the intersection. For the sake of simplicity, let us denote by Q the set $\{c,d\}$ and for every $R \subseteq A \setminus Q$, let $R' = \{(\{r\}, 0) : r \in R\}$.

(1) $az^{\mathbf{A}} = R \cup \{c\}$, where $R \subseteq A \setminus Q$. Then $az^{\mathbf{A}}\mu = R' \cup \{(Q, 1)\}$. On the other hand,

$$(\{a\}, 0)z^{\mathbf{D}} = \{a\}z^{\mathbf{A}/\Theta(c,d)} \times \{0,1\} = (R' \cup \{Q\}) \times \{0,1\}.$$

But $((R' \cup \{Q\}) \times \{0,1\}) \cap S = R' \cup \{(Q,1)\}$, and hence, $az^{\mathbf{A}}\mu = a\mu z^{\mathbf{S}}$ is valid for the case under consideration.

(2) $az^{\mathbf{A}} = R \cup \{d\}$, where $R \subseteq A \setminus Q$. Then $az^{\mathbf{A}}\mu = R' \cup \{(Q,2)\}$. Furthermore,

$$(\{a\},0)z^{\mathbf{D}} = \{a\}z^{\mathbf{A}/\Theta(c,d)} \times \{0,2\} = (R' \cup \{Q\}) \times \{0,2\}.$$

Now, $((R' \cup \{Q\}) \times \{0,2\}) \cap S = R' \cup \{(Q,2)\}$, and therefore, $az^{\mathbf{A}}\mu = a\mu z^{\mathbf{S}}$ is valid for this case as well.

(3) $az^{\mathbf{A}} = R \cup Q$, with $R \subseteq A \setminus Q$. In this case, $az^{\mathbf{A}}\mu = R' \cup \{(Q,1),(Q,2)\}$. Furthermore,

$$(\{a\},0)z^{\mathbf{D}} = \{a\}z^{\mathbf{A}/\Theta(c,d)} \times \{0,1,2\} = (R' \cup \{Q\}) \times \{0,1,2\}.$$

Now, $((R' \cup \{Q\}) \times \{0,1,2\}) \cap S = R' \cup \{(Q,1),(Q,2)\}$, and hence, $az^{\mathbf{A}}\mu = a\mu z^{\mathbf{S}}$ is valid for the case considered.

Finally, it is easy to see that $cz^{\mathbf{A}}\mu = c\mu z^{\mathbf{S}}$ and $dz^{\mathbf{A}}\mu = d\mu z^{\mathbf{S}}$. By the cases considered above, we get that μ is an isomorphism of $\mathbf{A}$ into the α_0-product $\mathbf{D}$. On the other hand, it is easy to check that $\mathbf{A}/\Theta(c,d)$ is a homomorphic image of $\mathbf{A}$, and thus, Corollary 1, Lemma 4 and the induction hypothesis result in that $\mathbf{A}$ can be embedded into an α_0-power of $\mathbf{B}$.

Case 2. $(A, \leq)$ has only one maximal element which is denoted by c. Then the partially ordered set $(A \setminus \{c\}, \leq)$ contains at least one maximal element. Let us denote it by b. For the sake of simplicity, let Q denote the set $\{b,c\}$. Now, let us define the α_0-product $\mathbf{A}/\Theta(b,c) \times \mathbf{B}(X,\Phi)$ as follows.

For every $z \in X$ and $a \in A \setminus Q$, let

$$\varphi_1(z) = z,$$

$$\varphi_2(\{a\},z) = \begin{cases} u & \text{if } az^{\mathbf{A}} \cap Q = \{c\}, \\ y & \text{if } az^{\mathbf{A}} \cap Q = \{b\}, \\ x & \text{otherwise,} \end{cases}$$

$$\varphi_2(Q,z) = \begin{cases} y & \text{if } bz^{\mathbf{A}} = \{b\}, \\ v & \text{otherwise.} \end{cases}$$

Define the mapping of A into $A/\Theta(b,c) \times \{0,1,2\}$ as follows:

$c\mu = (Q,2)$,
$b\mu = (Q,1)$,
$a\mu = (\{a\},0)$, for all $a \in A \setminus Q$,

and let $S = \{(\{a\},0) : a \in A \setminus Q\} \cup \{(Q,1),(Q,2)\}$.

Then it can be seen that μ is an isomorphism of $\mathbf{A}$ into the α_0-product considered, namely, $\mathbf{A}$ is isomorphic to the subautomaton determined by the set S. On the other hand, $\mathbf{A}/\Theta(b,c)$ is a homomorphic image of $\mathbf{A}$. Then Corollary 1, Lemma

4 and the induction hypothesis yield that $\mathbf{A}$ can be embedded into an α_0-power of $\mathbf{B}$ which ends the proof of Proposition 2.

It is interesting to note that we need the monotone n.d. automaton of three states not only for the convenience. This assertion is vitnissed by a commutative asynchronous n.d. automaton which can not be embedded into any general product of two-state monotone n.d. automata.

Let us consider the n.d. automaton $\mathbf{A} = (\{0,1,2,3\}, \{x,y,z\})$ which is defined in the following way:

$$0x^{\mathbf{A}} = \{1,3\},\ ix^{\mathbf{A}} = \{i\},\ i = 1,2,3,$$
$$0y^{\mathbf{A}} = \{2,3\},\ 1y^{\mathbf{A}} = \{2\},\ iy^{\mathbf{A}} = \{i\},\ i = 2,3,$$
$$iz^{\mathbf{A}} = \{3\},\ i = 0,1,2,3.$$

It is easy to check that $\mathbf{A} \in \mathcal{K}_{nd}$. Now, we prove that $\mathbf{A}$ can not be embedded into any general product of two-state monotone n.d. automata. Contrary, let us suppose that $\mathbf{A}$ can be embedded into a general product $\mathbf{D} = \prod_{j=1}^{k} \mathbf{A}_j(\{x,y,z\},\Phi)$ of two-state monotone n.d. automata. Without loss of generality, we may assume that the states of the n.d. automaton $\mathbf{A}_j$ are 0 and 1, moreover, there is no edge from 1 into 0 in the corresponding transition graph, for all j, $j = 1,\ldots,k$. Let μ denote a suitable isomorphism and let $i\mu = (e_{i1},\ldots,e_{ik})$, $i = 0,1,2,3$. Obviously, the vectors $(e_{i1},\ldots,e_{ik})$, $i = 0,1,2,3$ are binary vectors. The isomorphism and the monotone property of the components imply that $0\mu \le 1\mu \le 2\mu \le 3\mu$. Let us investigate the equality $0x^{\mathbf{A}}\mu = 0\mu x^{\mathbf{D}} \cap \{(e_{i1},\ldots,e_{ik}) : 0 \le i \le 3\}$. The left side is obviously $\{(e_{11},\ldots,e_{1k}),(e_{31},\ldots,e_{3k})\}$. By the definition of the general product, the right side is equal to the following set:

$$W = (\{e_{11},e_{31}\} \times \{e_{12},e_{32}\} \times \cdots \times \{e_{1k},e_{3k}\}) \cap \{(e_{i1},\ldots,e_{ik}) : 0 \le i \le 3\}.$$

Since $e_{1j} \le e_{2j} \le e_{3j}$, $j = 1,\ldots,k$ and $e_{ij} \in \{0,1\}$, for all $i = 1,2,3$; $j = 1,\ldots,k$, $(e_{21},\ldots,e_{2k}) \in W$ which is a contradiction.

By the observation above, we obtain the following statement.

Corollary 4. *There is no isomorphically complete system for $\mathcal{K}_{nd}$ with respect to the general product which consists of two-state monotone n.d. automata.*

Summarizing, the results presented here illustrate that although $\mathcal{K}_{nd}$ is a small and very particular class, the characterization of the isomorphically complete systems for $\mathcal{K}_{nd}$ with respect to the α_0-product can be very difficult. Proposition 1 shows that some isomorphically complete systems for $\mathcal{K}_{nd}$ must be infinite, while Proposition 2 implies that there are some finite isomorphically complete systems for $\mathcal{K}_{nd}$.

Acknowledgement. The authors thank Professor Ferenc Gécseg for his valuable observations on this paper.

References

[1] Gécseg, F., Composition of automata, Proceedings of the 2nd Colloquium on Automata, Languages and Programming, Saarbrücken, LNCS **14** (1974), 351-363.

[2] Gécseg, F., *Products of Automata*, Springer-Verlag, Berlin - Heidelberg-New York - Tokyo (1986).

[3] Gécseg, F., B. Imreh, On completeness of nondeterministicautomata, *Acta Math. Hungar.* **68** (1995), 151-159.

[4] Gécseg F., B. Imreh, On the cube-product of nondeterministic automata, *Acta Sci. Math. (Szeged)* **60** (1995), 321-327.

[5] Imreh, B., On α_i-products of automata, *Acta Cybernetica* **3** (1978), 301-307.

[6] Glushkov, V. M., Abstract theory of automata, *Uspekhi Mat. Nauk*, **16:5 101** (1961), 3-62 (in Russian).

[7] Imreh, B., M., Ito, On α_i-products of nondeterministic automata, *Algebra Colloquium*, **4** (1997), 195-202.

[8] Imreh, B., M. Ito, A. Pukler, On commutative asynchronous automata, Proceedings of The Third International Colloquium on Words, Languages, and Combinatorics, Kyoto, 2000, to appear.

Received September, 2000

Acta Cybernetica 14 (2000) 619-630.

Difference Functions of Dependence Spaces

Jouni Järvinen *

Abstract

Here the reduction problem is studied in an algebraic structure called dependence space. We characterize the reducts by the means of dense families of dependence spaces. Dependence spaces defined by indiscernibility relations are also considered. We show how we can determine dense families of dependence spaces induced by indiscernibility relations by applying indiscernibility matrices. We also study difference functions which connect the reduction problem to the general problem of identifying the set of all minimal Boolean vectors satisfying an isotone Boolean function.

1 Introduction

Z. Pawlak introduced his notion of information systems in the early 1980's [11]. Information concerning properties of objects is the basic knowledge included in information systems, and it is given in terms of attributes and values of attributes. For example, we may express statements concerning the color of objects if the information system includes the attribute "color" and a set of values of this attribute consisting of "green", "yellow", etc. It should be noted that relational databases can be viewed as information systems in the sense of Pawlak.

In an information system each subset of attributes defines an indiscernibility relation, which is an equivalence on the object set such that two objects are equivalent when their values of all attributes in the set are the same. It may turn out that a proper subset of a set of attributes classifies the objects with the same accuracy as the original set, which means that some attributes may be omitted. An attribute set C is a reduct of an attribute set B, if C is a minimal subset of B which defines the same indiscernibility relation as B. The reduction problem means that we want to enumerate all reducts of a given subset of attributes.

This work is devoted to the reduction problem in a dependence space. It is based on some papers of the same author, in particular on [5]. The fundamental notion appearing in the present paper is the concept of a dense family of a dependence space. We prove that our definition of dense families agrees with the definition presented earlier in the literature [10]; this result appeared also in [7]. Proposition 4.1 characterizes reducts in dependence spaces by the means of dense families. Also

*Turku Centre for Computer Science TUCS, Lemminkäisenkatu 14, FIN-20520 Turku, Finland. Email: jjarvine@cs.utu.fi

difference functions are defined by using dense families (cf. [5]). Proposition 5.2 characterizes reducts by the means of difference functions and Proposition 6.1 contains a construction of a dense family in the dependence space of an information system starting with its indiscernibility matrix; this result appeared also in [6].

As stated above, this paper gives a survey of some results concerning reducts and their construction, but the presented formulations are simpler than the formulations published earlier. It also completes proofs of some theorems published without proofs in the quoted papers.

2 Preliminaries

All general lattice theoretical and algebraic notions used in this paper can be found in [2, 4], for example. An oredered set (many authors use the shorthand poset) $(P, \leq)$ is a *join-semilattice* if the join $a \vee b$ exists for all $a, b \in P$. An equivalence relation Θ on P is a *congruence* relation on the semilattice $(P, \vee)$ if $a_1 \Theta b_1$ and $a_2 \Theta b_2$ imply $(a_1 \vee a_2)\Theta(b_1 \vee b_2)$ for all $a_1, a_2, b_1, b_2 \in P$. We denote by a/Θ the *congruence class* of a, that is, $a/\Theta = \{b \in P \mid a\Theta b\}$.

An ordered set $(P, \leq)$ is a *lattice* if $a \vee b$ and $a \wedge b$ exist for all $a, b \in P$. Let us consider a lattice $(P, \leq)$. An element $a \in P$ is *meet-irreducible* if $a = b \wedge c$ implies $a = b$ or $a = c$. We denote the set of all meet-irreducible elements $a \neq 1$ (in case P has a unit) of $(P, \leq)$ by $\mathcal{M}(P)$. The following lemma can be found in [2], for example.

Lemma 2.1. *If $(P, \leq)$ is a finite lattice, then*

$$a = \bigwedge\{x \in \mathcal{M}(P) \mid a \leq x\}$$

for all $a \in P$. □

Let $(P, \leq)$ be an ordered set. A subset S of P is *meet-dense* (see e.g. [2]), if for all $x \in P$ there exists a subset X of S such that $x = \bigwedge_P X$. Now the following lemma holds.

Lemma 2.2. *If $(P, \leq)$ is a finite lattice, then $S(\subseteq P)$ is meet-dense if and only if $\mathcal{M}(P) \subseteq S$.*

Proof. Let $S \subseteq P$ be meet-dense and $a \in \mathcal{M}(P)$. Since S is meet-dense and $a \neq 1$, there exists a finite nonempty subset $X = \{a_1, \ldots, a_n\}$ of S such that $a = a_1 \wedge \cdots \wedge a_n$. Because a is meet-irreducible, we obtain that $a \in X$ and so $a \in S$. Hence, $\mathcal{M}(P) \subseteq S$ holds.

Conversely, suppose that $\mathcal{M}(P) \subseteq S \subseteq P$. Then for all $a \in P$,

$$\{x \in \mathcal{M}(P) \mid a \leq x\} \subseteq \{x \in S \mid a \leq x\} \subseteq \{x \in P \mid a \leq x\},$$

which implies

$$\begin{aligned} a &= \bigwedge\{x \in \mathcal{M}(P) \mid a \leq x\} \geq \bigwedge\{x \in S \mid a \leq x\} \\ &\geq \bigwedge\{x \in P \mid a \leq x\} = a. \end{aligned}$$

Hence $a = \bigwedge\{x \in S \mid a \leq x\}$. This means that S is meet-dense. □

Let $(P, \leq)$ be an ordered set and $a, b \in P$. We say that a is *covered by* b, and write $a \prec b$, if $a < b$ and $a \leq c < b$ implies $a = c$. It is known (see e.g. [2]) that in a finite lattice $(P, \leq)$ the set of the elements of $(P, \leq)$ covered by exactly one element of P is $\mathcal{M}(P)$. Thus, by Lemma 2.2, a subset of a finite lattice $(P, \leq)$ is meet-dense if and only if it contains all elements of P which are covered by exactly one element of P.

A family $\mathcal{L}$ of subsets of a set A is said to be a *closure system* on A if $\mathcal{L}$ is closed under intersections, which means that for all $\mathcal{H} \subseteq \mathcal{L}$, we have $\bigcap \mathcal{H} \in \mathcal{L}$. We denote by $\wp(A)$ the *power set* of A, i.e., the set of all subsets of A. A *closure operator* on a set A is an extensive, idempotent and isotone map $\mathcal{C}\colon \wp(A) \to \wp(A)$; that is to say, $B \subseteq \mathcal{C}(B)$, $\mathcal{C}(\mathcal{C}(B)) = \mathcal{C}(B)$, and $B \subseteq C$ implies $\mathcal{C}(B) \subseteq \mathcal{C}(C)$ for all $B, C \subseteq A$. A subset B of A is *closed* (with respect to $\mathcal{C}$) if $\mathcal{C}(B) = B$. A closure system $\mathcal{L}$ on A defines a closure operator $\mathcal{C}_{\mathcal{L}}$ on A by the rule

$$\mathcal{C}_{\mathcal{L}}(B) = \bigcap\{X \in \mathcal{L} \mid B \subseteq X\}.$$

Conversely, if $\mathcal{C}$ is a closure operator on A, then the family

$$\mathcal{L}_{\mathcal{C}} = \{B \subseteq A \mid \mathcal{C}(B) = B\}$$

of closed subsets of A is a closure system. The relationship between closure systems and closure operators is bijective; the closure operator induced by the closure system $\mathcal{L}_{\mathcal{C}}$ is $\mathcal{C}$ itself, and the closure system induced by the closure operator $\mathcal{C}_{\mathcal{L}}$ is $\mathcal{L}$. It is well-known that if $\mathcal{L}$ is a closure system on A, then the ordered set $(\mathcal{L}, \subseteq)$ is a lattice in which

$$X \wedge Y = X \cap Y \quad \text{and} \quad X \vee Y = \mathcal{C}_{\mathcal{L}}(X \cup Y)$$

for all $X, Y \in \mathcal{L}$.

Next we consider meet-dense subsets of the lattice $(\mathcal{L}, \subseteq)$, where $\mathcal{L}$ is a closure system on a finite set.

Proposition 2.3. *Let $\mathcal{T}$ be a meet-dense subset of a lattice $(\mathcal{L}, \subseteq)$, where $\mathcal{L}$ is a closure system on a finite set A.*

(a) *For all $B \subseteq A$, $\mathcal{C}_{\mathcal{L}}(B) = \bigcap\{X \in \mathcal{T} \mid B \subseteq X\}$.*

(b) *For all $B, C \subseteq A$ the following three conditions are equivalent:*

(i) $\mathcal{C}_{\mathcal{L}}(B) \subseteq \mathcal{C}_{\mathcal{L}}(C)$;

(ii) *for all $X \in \mathcal{T}$, $C \subseteq X$ implies $B \subseteq X$;*

(iii) *for all $X \in \mathcal{T}$, $B - X \neq \emptyset$ implies $C - X \neq \emptyset$.*

Proof. (a) Because $\mathcal{C}_{\mathcal{L}}(B) \in \mathcal{L}$, and $B \subseteq X$ if and only if $\mathcal{C}_{\mathcal{L}}(B) \subseteq X$ for all $X \in \mathcal{L}$, we obtain by Lemmas 2.1 and 2.2 that

$$\begin{aligned} \mathcal{C}_{\mathcal{L}}(B) &= \bigcap\{X \in \mathcal{M}(\mathcal{L}) \mid \mathcal{C}_{\mathcal{L}}(B) \subseteq X\} = \bigcap\{X \in \mathcal{M}(\mathcal{L}) \mid B \subseteq X\} \\ &\supseteq \bigcap\{X \in \mathcal{T} \mid B \subseteq X\} \supseteq \bigcap\{X \in \mathcal{L} \mid B \subseteq X\} = \mathcal{C}_{\mathcal{L}}(B). \end{aligned}$$

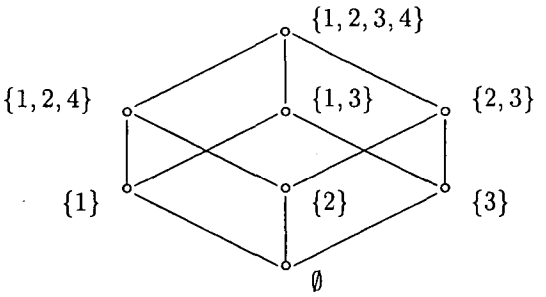

Figure 1: The closure lattice $(\mathcal{L}_{\mathcal{D}}, \subseteq)$

Hence, $\mathcal{C}_{\mathcal{L}}(B) = \{X \in \mathcal{T} \mid B \subseteq X\}$.

(b) Let $\mathcal{C}_{\mathcal{L}}(B) \subseteq \mathcal{C}_{\mathcal{L}}(C)$. If $X \in \mathcal{T}$ and $C \subseteq X$, then $B \subseteq \mathcal{C}_{\mathcal{L}}(B) \subseteq \mathcal{C}_{\mathcal{L}}(C) \subseteq \mathcal{C}_{\mathcal{L}}(X) = X$. Conversely, if for all $X \in \mathcal{T}$, $C \subseteq X$ implies $B \subseteq X$, then $\{X \in \mathcal{T} \mid C \subseteq X\} \subseteq \{X \in \mathcal{T} \mid B \subseteq X\}$. Hence by (a), $\mathcal{C}_{\mathcal{L}}(B) = \bigcap\{X \in \mathcal{T} \mid B \subseteq X\} \subseteq \bigcap\{X \in \mathcal{T} \mid C \subseteq X\} = \mathcal{C}_{\mathcal{L}}(C)$. Thus, (i) and (ii) are equivalent. Also (ii) and (iii) are equivalent since for all $X, Y \subseteq A$, $Y \subseteq X$ if and only if $Y - X = \emptyset$. □

3 Dense families of dependence spaces

We recall Novotný's and Pawlak's [9] definition of dependence spaces. We note that in [7] Järvinen studied infinite dependence spaces.

Definition. If A is a finite nonempty set and Θ is a congruence on the semilattice $(\wp(A), \cup)$, then the ordered pair $\mathcal{D} = (A, \Theta)$ is said to be a *dependence space*.

Let $\mathcal{D} = (A, \Theta)$ be a dependence space. Recalling the finiteness of A, it is clear that for every $B(\subseteq A)$, the congruence class B/Θ has a greatest element $\mathcal{C}_{\mathcal{D}}(B) = \bigcup B/\Theta$. It was noted in [8] that for all $B, C \subseteq A$,

$$B\Theta C \quad \text{if and only if} \quad \mathcal{C}_{\mathcal{D}}(B) = \mathcal{C}_{\mathcal{D}}(C).$$

In [8] it was also observed that $\mathcal{C}_{\mathcal{D}}\colon \wp(A) \to \wp(A), B \mapsto \bigcup B/\Theta$ is a closure operator on A. We denote by $\mathcal{L}_{\mathcal{D}}$ the closure system corresponding to the closure operator $\mathcal{C}_{\mathcal{D}}$. Hence, the family $\mathcal{L}_{\mathcal{D}}$ consists of the greatest elements of the Θ-classes.

Example 3.1. Let $A = \{1, 2, 3, 4\}$ and Θ be the congruence relation on $(\wp(A), \cup)$ whose congruence classes are $\{\emptyset\}$, $\{\{1\}\}$, $\{\{2\}\}$, $\{\{3\}\}$, $\{\{4\}, \{1, 2\}, \{1, 4\}, \{2, 4\}, \{1, 2, 4\}\}$, $\{\{1, 3\}\}$, $\{\{2, 3\}\}$ and $\{\{3, 4\}, \{1, 2, 3\}, \{1, 3, 4\}, \{2, 3, 4\}, \{1, 2, 3, 4\}\}$. The closure lattice $(\mathcal{L}_{\mathcal{D}}, \subseteq)$ corresponding to the dependence space $\mathcal{D} = (A, \Theta)$ is presented in Figure 1. Moreover, $\mathcal{M}(\mathcal{L}_{\mathcal{D}}) = \{\{1, 2, 4\}, \{1, 3\}, \{2, 3\}\}$.

Dense families of dependence spaces were introduced in [10]. Here we define them differently as meet-dense subsets of the lattice $(\mathcal{L}_{\mathcal{D}}, \subseteq)$; recall that in $(\mathcal{L}_{\mathcal{D}}, \subseteq)$, $X \wedge Y = X \cap Y$ for all $X, Y \in \mathcal{L}_{\mathcal{D}}$. We will also show that our definition agrees with Novotný's definition of dense families.

Definition. Let $\mathcal{D} = (A, \Theta)$ be a dependence space. A family $\mathcal{T} \subseteq \wp(A)$ is *dense* in $\mathcal{D}$ if it is a meet-dense subset of the lattice $(\mathcal{L}_{\mathcal{D}}, \subseteq)$.

By Lemma 2.2, a family $\mathcal{T}$ is dense in $\mathcal{D}$ if and only if it is a subfamily of $\mathcal{L}_{\mathcal{D}}$ which contains all elements of the lattice $(\mathcal{L}_{\mathcal{D}}, \subseteq)$ which are covered by exactly one element of $\mathcal{L}_{\mathcal{D}}$.

Example 3.2. Let us consider the dependence space $\mathcal{D} = (A, \Theta)$ of Example 3.1. The Hasse diagram of $(\mathcal{L}_{\mathcal{D}}, \subseteq)$ is given in Figure 1. The dense families of $\mathcal{D}$ are the 32 families $\mathcal{T}$ such that $\mathcal{M}(\mathcal{L}_{\mathcal{D}}) \subseteq \mathcal{T} \subseteq \mathcal{L}_{\mathcal{D}}$.

Let A be a set. Each family $\mathcal{T} \subseteq \wp(A)$ defines a binary relation $\Gamma(\mathcal{T})$ on $\wp(A)$:

$$(B, C) \in \Gamma(\mathcal{T}) \text{ if and only if } (\forall X \in \mathcal{T})\ B \subseteq X \iff C \subseteq X.$$

We note that in [10] dense families were defined by the condition presented in the next proposition.

Proposition 3.3. *Let $\mathcal{D} = (A, \Theta)$ be a dependence space. A family $\mathcal{T} \subseteq \wp(A)$ is dense in $\mathcal{D}$ if and only if $\Gamma(\mathcal{T}) = \Theta$.*

Proof. Let $\mathcal{T}$ be dense and $B\Theta C$. Then $\mathcal{C}_{\mathcal{D}}(B) = \mathcal{C}_{\mathcal{D}}(C)$, which implies by Proposition 2.3(b) that for all $X \in \mathcal{T}$, $B \subseteq X$ iff $C \subseteq X$. Thus, $\Theta \subseteq \Gamma(\mathcal{T})$. Conversely, if $(B, C) \in \Gamma(\mathcal{T})$, then

$$\mathcal{C}_{\mathcal{D}}(B) = \bigcap\{X \in \mathcal{T} \mid B \subseteq X\} = \bigcap\{X \in \mathcal{T} \mid C \subseteq X\} = \mathcal{C}_{\mathcal{D}}(C),$$

which is equivalent to $B\Theta C$. Hence, also $\Gamma(\mathcal{T}) \subseteq \Theta$.

On the other hand, let $\Gamma(\mathcal{T}) = \Theta$. We will show that $\mathcal{M}(\mathcal{L}_{\mathcal{D}}) \subseteq \mathcal{T} \subseteq \mathcal{L}_{\mathcal{D}}$, which implies by Lemma 2.2 that $\mathcal{T}$ is a meet-dense subset of $(\mathcal{L}_{\mathcal{D}}, \subseteq)$. Suppose that $X \in \mathcal{T}$. Because $X\Theta\mathcal{C}_{\mathcal{D}}(X)$ and $X \subseteq X$, we obtain $\mathcal{C}_{\mathcal{D}}(X) \subseteq X$, which implies $X \in \mathcal{L}_{\mathcal{D}}$. Hence, $\mathcal{T} \subseteq \mathcal{L}_{\mathcal{D}}$.

Assume that $\mathcal{M}(\mathcal{L}_{\mathcal{D}}) \not\subseteq \mathcal{T}$. This means that there exists a $Y \in \mathcal{M}(\mathcal{L}_{\mathcal{D}})$ such that $Y \notin \mathcal{T}$. Since $Y \in \mathcal{M}(\mathcal{L}_{\mathcal{D}})$, there exists exactly one $Z \in \mathcal{L}_{\mathcal{D}}$ such that $Y \prec Z$ holds in $\mathcal{L}_{\mathcal{D}}$. For all $X \in \mathcal{T}$, $Z \subseteq X$ implies obviously that $Y \subseteq X$. Suppose that there is an $X \in \mathcal{T}$ such that $Y \subseteq X$ but $Z \not\subseteq X$. Since $X, Z \in \mathcal{L}_{\mathcal{D}}$, we get $X \cap Z \in \mathcal{L}_{\mathcal{D}}$ and $Y \subseteq X \cap Z \subset Z$. The fact that $Y \prec Z$ holds in $\mathcal{L}_{\mathcal{D}}$ implies $Y = X \cap Z$. Because Y is meet-irreducible, we obtain $Y = X$ or $Y = Z$. Obviously both of these equalities lead to a contradiction! Hence, for all $X \in \mathcal{T}$, also $Y \subseteq X$ implies $Z \subseteq X$. Thus, $(Y, Z) \in \Gamma(\mathcal{T}) = \Theta$, which means that $Y = \mathcal{C}_{\mathcal{D}}(Y) = \mathcal{C}_{\mathcal{D}}(Z) = Z$, a contradiction! Therefore, also $\mathcal{M}(\mathcal{L}_{\mathcal{D}}) \subseteq \mathcal{T}$ holds. $\square$

4 Independent sets and reducts

In this section we review independent sets and reducts defined in dependence spaces. Further references can be found in [8, 9, 10], for example. Our main result of this section gives a characterization of the reducts of a given subset of a dependence space in terms of dense families.

Let $\mathcal{D} = (A, \Theta)$ be a dependence space. A subset $B(\subseteq A)$ is called *independent*, if B is minimal with respect to inclusion in its Θ-class. We denote the set of all independent subsets of $\mathcal{D}$ by $IND_{\mathcal{D}}$.

The notion of reducts is important in the theory of Pawlak's information systems. Here we study reducts in the more algebraic setting of dependence spaces. For any $B(\subseteq A)$ a set $C(\subseteq A)$ is called a *reduct* of B, if $C \subseteq B$, $B\Theta C$ and $C \in IND_{\mathcal{D}}$. The set of all reducts of B will be denoted by $RED_{\mathcal{D}}(B)$. In the other words, a subset C $(\subseteq B)$ is a reduct of B, if C is minimal in B/Θ with respect to inclusion. Because A is finite, it is obvious that every set has at least one reduct.

Finding all reducts of a given set is called the *reduction problem.* Our next proposition, which appears without a proof also in [6], characterizes the reducts of a given set by the means of dense families.

Proposition 4.1. *Let $\mathcal{T}$ be a dense family in a dependence space $\mathcal{D} = (A, \Theta)$. If $B \subseteq A$, then $C \in RED_{\mathcal{D}}(B)$ if and only if C is minimal set with respect to the property of containing an element from each nonempty difference $B - X$, where $X \in \mathcal{T}$.*

Proof. Let C be a minimal set which contains an element from each nonempty difference $B - X$, $X \in \mathcal{T}$. First we show that $C \subseteq B$. If $C \not\subseteq B$, then $B \cap C \subset C$ and $(B \cap C) \cap (B - X) = C \cap (B - X) \neq \emptyset$ whenever $B - X \neq \emptyset$, a contradiction! Thus, $C \subseteq B$. Now $C - X = (B \cap C) - X = C \cap (B - X) \neq \emptyset$ for all $X \in \mathcal{T}$ such that $B - X \neq \emptyset$. This implies by Proposition 2.3(b) that $\mathcal{C}_{\mathcal{D}}(B) \subseteq \mathcal{C}_{\mathcal{D}}(C)$. The inclusion $\mathcal{C}_{\mathcal{D}}(C) \subseteq \mathcal{C}_{\mathcal{D}}(B)$ is obvious. Hence, $B\Theta C$. Assume that $C \notin IND_{\mathcal{D}}$. Then there exists a $D \subset C$ such that $C\Theta D$. Since Θ is transitive, we obtain $B\Theta D$ and in particular $\mathcal{C}_{\mathcal{D}}(B) \subseteq \mathcal{C}_{\mathcal{D}}(D)$. This implies by Proposition 2.3(b) that $D \cap (B - X) = D - X \neq \emptyset$ whenever $B - X \neq \emptyset$, a contradiction! Hence, C is independent and thus C is a reduct of B.

On the other hand, suppose $C \in RED_{\mathcal{D}}(B)$. Then $C \subseteq B$, $B\Theta C$, $C \in IND_{\mathcal{D}}$, and especially $\mathcal{C}_{\mathcal{D}}(B) \subseteq \mathcal{C}_{\mathcal{D}}(C)$. This implies that $C \cap (B - X) = (B \cap C) - X = C - X \neq \emptyset$ for all $X \in \mathcal{T}$ which satisfy $B - X \neq \emptyset$. Assume that there exists a $D \subset C$ which contains an element from each nonempty difference $B - X$, where $X \in \mathcal{T}$. Then $D - X = (B \cap D) - X = D \cap (B - X) \neq \emptyset$ for all $X \in \mathcal{T}$ such that $B - X \neq \emptyset$. Hence, $\mathcal{C}_{\mathcal{D}}(B) \subseteq \mathcal{C}_{\mathcal{D}}(D)$. Since $D \subset B$ also $\mathcal{C}_{\mathcal{D}}(D) \subseteq \mathcal{C}_{\mathcal{D}}(B)$ holds. This implies $B\Theta D$, and because $C\Theta B$ we obtain $C\Theta D$, a contradiction! □

Example 4.2. Let us consider the dependence space $\mathcal{D} = (A, \Theta)$ defined in Example 3.1. We have already noted that $\mathcal{M}(\mathcal{L}_{\mathcal{D}}) = \{\{1, 2, 4\}, \{1, 3\}, \{2, 3\}\}$ is the smallest dense family.

Next we find the reducts of A. The differences $A - X$, where $X \in \mathcal{M}(\mathcal{L}_{\mathcal{D}})$, are

$$A - \{1,2,4\} = \{3\},\ A - \{1,3\} = \{2,4\}, \text{ and } A - \{2,3\} = \{1,4\}.$$

They are all nonempty. Because the reducts of A must contain an element from all of these differences, each reduct must include 3. It can be easily seen that $\{1,2,3\}$ and $\{3,4\}$ are the reducts of A.

5 The difference function

In this section we study the notion of difference function. Difference functions were introduced in [5]. Here we give an equivalent, but a clearer definition. First we recall some notions concerning Boolean functions (see e.g. [1], where further references can be found). A *Boolean function*, or a *function* for short, is a mapping $f : \{0,1\}^n \to \{0,1\}$. An element $v \in \{0,1\}^n$ is called a *Boolean vector* (a *vector* for short). If $f(v) = 1$ (resp. 0), then v is called a *true* (resp. *false*) vector of f. The set of all true vectors (resp. false vectors) of f is denoted by $T(f)$ (resp. $F(f)$).

Let $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$ be vectors. We set $u \leq v$ if and only if $u_i \leq v_i$, for $1 \leq i \leq n$. A function f is *isotone* if $u \leq v$ always implies $f(u) \leq f(v)$.

In the sequel we assume that f is an isotone function. A true vector v of f is *minimal* if there is no true vector w such that $w < v$, and let $\min T(f)$ denote the set of all minimal true vectors of f. A *maximal* false vector is symmetrically defined and $\max F(f)$ denotes the set of all maximal false vectors of f.

Let $\mathcal{D} = (A, \Theta)$ be a dependence space such that $A = \{a_1, \ldots, a_n\}$ and let $\mathcal{T}$ be dense in $\mathcal{D}$. For any $B \subseteq A$, let $\delta(B)$ denote the disjunction of all variables y_i, where $a_i \in B$. We define the *difference function* $f_B^{\mathcal{T}}(y_1, \ldots, y_n)$ as the conjunction

$$\bigwedge_{\substack{X \in \mathcal{T} \\ B - X \neq \emptyset}} \delta(B - X).$$

Clearly, the function $f_B^{\mathcal{T}}$ is isotone. A function $\chi : \wp(A) \to \{0,1\}^n$ is defined by

$$B \mapsto (\chi_1(B), \ldots, \chi_n(B)),$$

where

$$\chi_i(B) = \begin{cases} 0 & \text{if } a_i \notin B \\ 1 & \text{if } a_i \in B \end{cases}$$

for all i, $1 \leq i \leq n$. The value $\chi(B)$ is called the *characteristic vector* of B.

Now the following lemma holds.

Lemma 5.1. *Let $\mathcal{T}$ be a dense family in a dependence space $\mathcal{D} = (A, \Theta)$. For all $B, C \subseteq A$, the following conditions are equivalent:*

(a) $\chi(C) \in T(f_B^{\mathcal{T}})$;

(b) *C contains an element from each nonempty difference $B - X$, $X \in \mathcal{T}$.*

Proof. Let $B, C \subseteq A$ and $\{X \in \mathcal{T} \mid B - X \neq \emptyset\} = \{X_1 \ldots X_k\}$.

(a) $\Rightarrow$ (b) Assume that $f_B^{\mathcal{T}}(\chi(C)) = 1$. If $C \cap (B - X_i) = \emptyset$ for some i, $1 \leq i \leq k$, then obviously the disjunction $\delta(B - X_i)$ has the value 0 for $\chi(C)$. This implies that

also the conjunction $\bigwedge_{1\leq i\leq k}\delta(B-X_i)$ has the value 0 for $\chi(C)$, a contradiction! Hence, $C\cap(B-X_i)\neq\emptyset$ for all i, $1\leq i\leq k$.

(b) $\Rightarrow$ (a) Suppose that $C\cap(B-X_i)\neq\emptyset$ for all $1\leq i\leq n$. Then for all $1\leq i\leq n$, the disjunction $\delta(B-X_i)$ has the value 1 for $\chi(C)$. This implies that the conjunction $\bigwedge_{1\leq i\leq k}\delta(B-X_i)$ has the value 1 for $\chi(C)$, i.e., $f_B^{\mathcal{T}}(\chi(C))=1$. □

Now we can write the following proposition. Note that for any $X\subseteq A$, $X^{\complement}=A-X$ is the *complement* of X.

Proposition 5.2. *Let $\mathcal{T}$ be a dense family in a dependence space $\mathcal{D}=(A,\Theta)$. If $B\subseteq A$, then*

(a) $\min T(f_B^{\mathcal{T}})=\{\chi(C)\mid C\in RED_{\mathcal{D}}(B)\}$ *and*

(b) $\max F(f_B^{\mathcal{T}})=\max\{\chi((B-X)^{\complement})\mid X\in\mathcal{T},B-X\neq\emptyset\}$.

Proof. Let us denote $f_B^{\mathcal{T}}$ simply by f.

(a) Let $v\in\min T(f)$ and let C be the subset of A which satisfies $\chi(C)=v$. By Lemma 5.1 C contains an element from each nonempty difference $B-X$, where $X\in\mathcal{T}$. Assume that C is not minimal set with respect to that property, that is, there exist a $D\subset C$ which also contains an element from each nonempty difference $B-X$, where $X\in\mathcal{T}$. By Lemma 5.1 this implies that $\chi(D)\in T(f)$. But $D\subset C$ implies $\chi(D)<\chi(C)$ and hence $\chi(C)\notin\min T(f)$, a contradiction! Therefore, C is minimal set with respect to the property of containing an element from each nonempty difference $B-X$, where $X\in\mathcal{T}$. This implies that C is a reduct of B by Proposition 4.1.

On the other hand, suppose that C is a reduct of B. Then C contains an element from each nonempty difference $B-X$, where $X\in\mathcal{T}$, and thus $\chi(C)\in T(f)$. Suppose that $\chi(C)\notin\min T(f)$. This means that there exists a vector $v\in T(f)$ such that $v<\chi(C)$. Let D be the subset of A which satisfies $\chi(D)=v$. Then obviously D is a set which contains an element from each nonempty difference $B-X$, where $X\in\mathcal{T}$. Since $D\subset C$, C is not a reduct of B, a contradiction! Hence, $\chi(C)\in\min T(f)$.

(b) By Lemma 5.1 it is obvious that $f(\chi(C))=0$ if and only if there exist an $X\in\mathcal{T}$ such that $B-X\neq\emptyset$ and $C\cap(B-X)=\emptyset$. This is equivalent to the condition that $f(\chi(C))=0$ if and only if there exist an $X\in\mathcal{T}$ such that $B-X\neq\emptyset$ and $C\subseteq(B-X)^{\complement}$.

Suppose that $\chi(C)\in\max F(f)$. This implies that $C\subseteq(B-X)^{\complement}$ for some $X\in\mathcal{T},B-X\neq\emptyset$, and hence $\chi(C)\leq\chi((B-X)^{\complement})$. Assume that $\chi(C)<\chi((B-X)^{\complement})$. Since $\chi((B-X)^{\complement})\in F(f)$, this implies $\chi(C)\notin\max F(f)$, a contradiction! Hence, $\chi(C)\in\{\chi((B-X)^{\complement})\mid X\in\mathcal{T},B-X\neq\emptyset\}$. Assume that there exists a $\chi(D)\in\{\chi((B-X)^{\complement})\mid X\in\mathcal{T},B-X\neq\emptyset\}$ such that $\chi(C)<\chi(D)$. Clearly, this implies that $\chi(D)\in F(f)$ and hence $\chi(C)\notin\max F(f)$, a contradiction! Thus, $\chi(C)\in\max\{\chi((B-X)^{\complement})\mid X\in\mathcal{T},B-X\neq\emptyset\}$.

Conversely, suppose that $\chi(C)\in\max\{\chi((B-X)^{\complement})\mid X\in\mathcal{T},B-X\neq\emptyset\}$. Then obviously $\chi(C)\in F(f)$. Assume that there exists a $\chi(D)\in F(f)$ such that $\chi(C)<\chi(D)$. This implies that there exists an $X\in\mathcal{T}$ such that $D\subseteq(B-X)^{\complement}$

and $B - X \neq \emptyset$. We obtain that $\chi(C) < \chi(D) \leq \chi((B-X)^{\complement})$ for some $X \in \mathcal{T}$ such that $B - X \neq \emptyset$, a contradiction! Hence, $\chi(C) \in \max F(f)$. □

Hence, the minimal true vectors of the difference function of $B(\subseteq A)$ are the characteristic vectors of the reducts of B.

Example 5.3. Let us consider the dependence space $\mathcal{D} = (A, \Theta)$ defined in Example 3.1. The family $\mathcal{T} = \{\{1,2,4\},\{1,3\},\{2,3\}\}$ is known to be dense in $\mathcal{D}$. The differences $A - X$ are all nonempty for all $X \in \mathcal{T}$. Hence,

$$\begin{aligned} f_A^{\mathcal{T}} &= \delta(A - \{1,2,4\}) \wedge \delta(A - \{1,3\}) \wedge \delta(A - \{2,3\}) \\ &= 3 \wedge (2 \vee 4) \wedge (1 \vee 4), \end{aligned}$$

where i stands for y_i. The function $f_A^{\mathcal{T}}$ has the minimal true vectors $(0,0,1,1)$ and $(1,1,1,0)$, which implies by Proposition 5.2 that $RED_{\mathcal{D}}(A) = \{\{3,4\},\{1,2,3\}\}$.

The *dual* of a Boolean function f, denoted by f^d, is defined by $f^d(x) = \overline{f}(\overline{x})$, where $\overline{f}$ and $\overline{x}$ denote the complements of f and x, respectively. It is well-known that $(f^d)^d = f$ and that the DNF expression of f^d is obtained from that of f by exchanging $\vee$ and $\wedge$ as well as constants 0 and 1, and then expanding the resulting formula. For example, the dual of $g = 3 \vee (1 \wedge 4) \vee (2 \wedge 4)$ is $g^d = 3 \wedge (1 \vee 4) \wedge (2 \vee 4) = (3 \wedge 4) \vee (1 \wedge 2 \wedge 3)$.

It is known (see e.g. [1]) that for any isotone Boolean function f, $\min T(f^d) = \{\overline{v} \mid v \in \max F(f)\}$. Let us denote $f_B^{\mathcal{T}}$ simply by f. By Proposition 5.2:

$$\begin{aligned} v \in \min T(f^d) &\iff \overline{v} \in \max F(f) \\ &\iff \overline{v} \in \max\{\chi((B-X)^{\complement}) \mid X \in \mathcal{T}, B - X \neq \emptyset\} \\ &\iff v \in \min\{\chi(B - X) \mid X \in \mathcal{T}, B - X \neq \emptyset\}. \end{aligned}$$

The family $\mathcal{T} = \{\{1,2,4\},\{1,3\},\{2,3\}\}$ is known to be dense in the dependence space $\mathcal{D}$ of Example 3.1. Let us denote by f the difference function of the set A. Then

$$\begin{aligned} \min(f^d) &= \min\{\chi(A - X) \mid X \in \mathcal{T}, A - X \neq \emptyset\} \\ &= \{(0,0,1,0),(0,1,0,1),(1,0,0,1)\}. \end{aligned}$$

This means that $f^d = 3 \vee (1 \wedge 4) \vee (2 \wedge 4)$ and $f = (f^d)^d = (3 \wedge 4) \vee (1 \wedge 2 \wedge 3)$. Hence, $\min T(f) = \{(0,0,1,1),(1,1,1,0)\}$, as stated in Example 5.3.

Remark. Let $f = f(x_1, \ldots, x_n)$ and $g = g(x_1, \ldots, x_n)$ be a pair of isotone Boolean functions given by their minimal true vectors $\min T(f)$ and $\min T(g)$, respectively. Let us consider the following problem; test whether f and g are mutually dual. In [3] Fredman and Khachiyan showed that this problem can be solved in time $k^{o(\log k)}$, where $k = |\min T(f)| + |\min T(g)|$.

This implies that for an isotone Boolean function f given by its minimal true vectors and for a subset $G \subset \min T(f^d)$, a new vector $v \in \min T(f^d) - G$ can be

computed in time $nk^{o(\log k)}$, where $k = |\min T(f)| + |G|$ (see [1], for example). This means also that for any isotone Boolean function f given by its minimal true vectors, f^d can be computed in time $nk^{o(\log k)}$, where $k = |\min T(f)| + |\min T(f^d)|$.

6 An application to information systems

An *information system* is a triple $\mathcal{S} = (U, A, \{V_a\}_{a \in A})$, where U is a set of *objects*, A is a set of *attributes*, and $\{V_a\}_{a \in A}$ is an indexed set of *value sets of attributes*. All these sets are assumed to be finite and nonempty. Each attribute is a function $a: U \to V_a$ which assigns a value of the attribute a to objects (see e.g. [9, 10, 11]).

For any $B \subseteq A$, the *indiscernibility relation* of B is defined by

$$I(B) = \{(x, y) \in U^2 \mid a(x) = a(y) \text{ for all } a \in B\}.$$

It is known that $I(B)$ is an equivalence relation on U such that its equivalence classes consist of objects which are indiscernible with respect to all attributes in B. Let us define the following binary relation $\Theta_{\mathcal{S}}$ on the set $\wp(A)$:

$$(B, C) \in \Theta_{\mathcal{S}} \iff I(B) = I(C).$$

So, two subsets of attributes are in the relation $\Theta_{\mathcal{S}}$ if and only if they define the same indiscernibility relation. It is known (see e.g. [8, 9]) that $\Theta_{\mathcal{S}}$ is a congruence on the semilattice $(\wp(A), \cup)$. Hence, the pair $\mathcal{D}_{\mathcal{S}} = (A, \Theta_{\mathcal{S}})$ is a dependence space. It can be easily seen that C ($\subseteq A$) is a reduct of B ($\subseteq A$) in the dependence space $\mathcal{D}_{\mathcal{S}}$ if and only if C is a minimal subset of B which defines the same indiscernibility relation as B.

Assume that $U = \{x_1, \ldots, x_m\}$. Then the *indiscernibility matrix* of $\mathcal{S}$ is an $m \times m$-matrix $\mathbf{M}_{\mathcal{S}} = (c_{ij})_{m \times m}$ such that

$$c_{ij} = \{a \in A \mid a(x) = a(y)\}$$

for all $1 \leq i, j \leq m$. Thus, the entry c_{ij} consists of the attributes which do not discern objects x_i and x_j (cf. discernibility matrices defined in [12]). It is now trivial that

$$(x_i, x_j) \in I(B) \iff B \subseteq c_{ij}.$$

Next we show how matrices of preimage relations induce dense families.

Proposition 6.1. *If $\mathcal{S} = (U, A, \{V_a\}_{a \in A})$ is an information system and $\mathbf{M}_{\mathcal{S}} = (c_{ij})_{m \times m}$ is the indiscernibility matrix of $\mathcal{S}$, then the family*

$$\mathcal{T}_{\mathcal{S}} = \{c_{ij} \mid 1 \leq i, j \leq m\}$$

is dense in the dependence space $\mathcal{D}_{\mathcal{S}} = (A, \Theta_{\mathcal{S}})$.

Proof. By Proposition 3.3 it suffices to show that $\Gamma(\mathcal{T}_S) = \Theta_S$. If $(B, C) \in \Theta_S$, then for all $1 \leq i, j \leq m$, $B \subseteq c_{ij}$ iff $(x_i, x_j) \in I(B)$ iff $(x_i, x_j) \in I(C)$ iff $C \subseteq c_{ij}$, which implies $(B, C) \in \Gamma(\mathcal{T}_S)$. Hence, $\Theta_S \subseteq \Gamma(\mathcal{T}_S)$.

If $(B, C) \in \Gamma(\mathcal{T}_S)$, then for all $1 \leq i, j \leq m$, $(x_i, x_j) \in I(B)$ iff $B \subseteq c_{ij}$ iff $C \subseteq c_{ij}$ iff $(x_i, x_j) \in I(C)$, which implies $I(B) = I(C)$. Thus, also $\Gamma(\mathcal{T}_S) \subseteq \Theta_S$ and hence $\Gamma(\mathcal{T}_S) = \Theta_S$. □

We conclude this paper by an example.

Example 6.2. Let us consider an information system $\mathcal{S} = (U, A, \{V_A\}_{a \in A})$, where the object set $U = \{1, 2, 3, 4, 5\}$ consists of five persons, the attribute set A consists of the attributes Age, Eyes, and Height, and the corresponding value sets are $V_{\text{Age}} = \{\text{Young, Middle, Old}\}$, $V_{\text{Eyes}} = \{\text{Blue, Brown, Green}\}$, and $V_{\text{Height}} = \{\text{Short, Normal, Tall}\}$.

Let the values of the attributes be defined as in the following table.

	Age	Eyes	Height
1	Young	Blue	Short
2	Young	Brown	Normal
3	Middle	Brown	Tall
4	Old	Green	Normal
5	Young	Brown	Normal

For example, the indiscernibility relation $I(A)$ of the attribute set A is an equivalence on U which has the equivalence classes $\{1\}$, $\{2, 5\}$, $\{3\}$, and $\{4\}$.

If we denote a = Age, b = Eyes, and c = Height, then the indiscernibility matrix of $\mathcal{S}$ is the following:

$$\mathbf{M}_{\mathcal{S}} = \begin{pmatrix} A & \{a\} & \emptyset & \emptyset & \{a\} \\ \{a\} & A & \{b\} & \{c\} & A \\ \emptyset & \{b\} & A & \emptyset & \{b\} \\ \emptyset & \{c\} & \emptyset & A & \{c\} \\ \{a\} & A & \{b\} & \{c\} & A \end{pmatrix}$$

By Proposition 6.1, the family $\mathcal{T}_S = \{\emptyset, \{a\}, \{b\}, \{c\}, A\}$ consisting of the entries of $\mathbf{M}_S$ is dense in the dependence space $\mathcal{D}_S = (A, \Theta_S)$. Let us denote by f the difference function of the set A. Then $\min(f^d) = \min\{\chi(A - X) \mid X \in \mathcal{T}, A - X \neq \emptyset\} = \{(0,1,1), (1,0,1), (1,1,0)\}$. This means that $f^d = (b \wedge c) \vee (a \wedge c) \vee (a \wedge b)$ and $f = (b \vee c) \wedge (a \vee c) \wedge (a \vee b) = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c)$. Obviously, $(1,1,0)$, $(1,0,1)$ and $(0,1,1)$ are the minimal true vectors of f. Thus, $\{a, b\}$, $\{a, c\}$, and $\{b, c\}$ are the reducts of A in $\mathcal{D}_S$.

Acknowledgement

The author thanks the referee for his valuable comments and suggestions.

References

[1] J. C. BIOCH, T. IBARAKI, *Complexity of identification and dualization of positive Boolean functions*, Information and Computation 123 (1995), 50–63.

[2] B. A. DAVEY, H.A. PRIESTLEY, *Introduction to lattices and order*, Cambridge University Press, Cambridge, 1990.

[3] M. FREDMAN, L. KHACHIYAN, *On the complexity of dualization of monotone disjunctive normal forms*, Journal of Algorithms 21 (1996), 618–628.

[4] G. GRÄTZER, *Lattice theory: first concepts and distributive lattices*, W. H. Freeman and company, San Francisco, 1971.

[5] J. JÄRVINEN, *A representation of dependence spaces and some basic algorithms*, Fundamenta Informaticae 29 (1997), 369–382.

[6] J. JÄRVINEN, *Preimage relations and their matrices.* In L. POLKOWSKI, A. SKOWRON (eds.), *Rough sets and current trends in computing*, Lecture Notes in Artificial Intelligence 1424, Springer-Verlag, Berlin/Heidelberg, 139–146, 1998.

[7] J. JÄRVINEN, *Knowledge representation and rough sets*, PhD Dissertation, University of Turku, Department of Mathematics, March 1999 (available at `http://www.cs.utu.fi/jjarvine`).

[8] J. NOVOTNÝ, M. NOVOTNÝ, *Notes on the algebraic approach to dependence in information systems*, Fundamenta Informaticae 16 (1992), 263–273.

[9] M. NOVOTNÝ, Z. PAWLAK, *Algebraic theory of independence in information systems*, Fundamenta Informaticae 14 (1991), 454–476.

[10] M. NOVOTNÝ, *Applications of dependence spaces.* In E. ORŁOWSKA (ed.), *Incomplete information: rough set analysis*, Physica–Verlag, Heidelberg/New York, 247–289, 1998.

[11] Z. PAWLAK, *Information systems. Theoretical foundations*, Informations Systems 6 (1981), 205–218.

[12] A. SKOWRON, C. RAUSZER, *The discernibility matrices and functions in information systems.* In R. SLOWINSKI (ed.), *Intelligent decision support. Handbook of applications and advances of the rough set theory*, Kluwer Academic Publisher, Dordrecht, 331–362, 1991.

Received November, 1999

Acta Cybernetica 14 (2000) 631-652.

Elementary decomposition of soliton automata*

Miklós Bartha † Miklós Krész ‡

Abstract

Soliton automata are the mathematical models of certain possible molecular switching devices. In this paper we work out a decomposition of soliton automata through the structure of their underlying graphs. These results lead to the original aim, to give a characterization of soliton automata in general case.

1 Introduction

One of the most important goals of research in bioelectronics is to develop a molecular computer (see e.g. [3]). The soliton automaton introduced in [4] is the mathematical model of so-called "soliton valves" having the potential to serve as a molecular switching device in such a computer architecture.

The underlying object of a soliton automaton is a soliton graph, which is the topological model of a hydrocarbon molecule-chain in which the appropriate soliton waves travel along. Any soliton graph has a perfect internal matching, i.e. a matching that covers all the vertices with degree at least two. These vertices model the carbon atoms, whereas vertices with degree one (external vertices) represent an interface with the outside world. The states of the corresponding automaton – also called the states of the graph – are the perfect internal matchings of the underlying graph, while the transitions are realized by making soliton walks. Intuitively, a soliton walk is an alternating walk with respect to some state M of the graph G, which starts and ends at an external vertex. However, the status of each edge in the walk regarding its presence in M changes dynamically step by step while making the walk, so that by the time the walk is finished, a new state of G is reached.

The analysis of soliton automata is a very complex task. So far only a few special cases have been described. In [4], [5] and [6], the transition monoids were determined for strongly deterministic soliton automata, deterministic soliton automata with a single external vertex or with one cycle. Following a different approâch, in

*This work was partially supported by Soros Foundation. Presented at the Conference of PhD Students on Computer Science, July 20-23 2000, Szeged.

†Department of Informatics, University of Szeged, 6720, Szeged, Árpád tér 2, Hungary, e-mail: bartha@inf.u-szeged.hu

‡Department of Computer Science, University of Szeged, Faculty of Juhász Gyula Teacher Training College, 6725, Szeged, Boldogasszony sgt. 6, Hungary, e-mail: kresz@jgytf.u-szeged.hu

[8], the computational power of strongly deterministic soliton automata have been investigated by automata products. However, the general case is still open.

The main contribution of this paper is to reduce the general problem to a simpler one by working out a decomposition of soliton automata into elementary ones. For this goal we make use of the elementary structure of soliton graphs found in [2]. In Section 3 we describe the automata based on the internal parts of this decomposition, then characterize the relationship of component automata by α_0^ε-products. In Section 4 the self-transitions – transitions from a state to itself – induced by non-trivial walks are investigated. This problem will be analyzed also through the elementary decomposition.

2 Basic concepts and preliminaries

By a graph we mean, unless otherwise specified, a finite undirected graph in the most general sense, i.e. with multiple edges and loops allowed. For a graph G, $V(G)$ and $E(G)$ will denote the set of vertices and the set of edges of G, respectively. An edge $e = (v_1, v_2) \in E(G)$ connects two vertices $v_1, v_2 \in V(G)$, which are called the *endpoints* of e, and e is said to be *incident* with v_1 and v_2. If $v_1 = v_2$, then e is called a *loop* around v_1. Two edges sharing at least one endpoint are said to be *adjacent* in G. A subgraph G' of G is a graph such that $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. If $X \subseteq V(G)$ then $G[X]$ denotes the subgraph of G induced by X, i.e. $V(G[X]) = X$ and E(G[X]) consists of the edges of G having both endpoints in X. Moreover, we say that the set $E \subseteq E(G)$ *spans* the subgraph G' if $G' = G[X]$, where X is the set of vertices incident with some edge of E.

If the vertex set of a graph G can be partitioned into two disjoint non-empty sets A and B such that all edges of G join a vertex in A to a vertex in B, we call G *bipartite* and refer to (A, B) as the *bipartition* of G.

The *degree* of a vertex v in graph G is the number of occurences of v as an endpoint of some edge in $E(G)$. According to this definition, every loop around v contributes two occurences to the count. The vertex v is called *external* if its degree $d(v)$ is one, *internal* if $d(v) \geq 1$, and *isolated* otherwise. *External edges* are those that are incident with at least one external vertex, whereas an edge is *internal*, if it is not external. The sets of external and internal vertices of G will be denoted by $Ext(G)$ and $Int(G)$, respectively.

A *matching* M of graph G is a subset of $E(G)$ such that no vertex of G occurs more than once as an endpoint of some edge in M. Again, it is understood that loops are not allowed to participate in M. The endpoints of the edges contained in M are said to be covered by M. A *perfect internal matching* is one that covers all the internal vertices of G. An edge $e \in E(G)$ is *allowed (mandatory)* if e is contained in some (respectively, all) perfect internal matching(s) of G. *Forbidden edges* are those that are not allowed. We will also use the name *constant edge* as a common reference to forbidden and mandatory edges. A perfect internal matching in G will be also referred to as a *state* of G, and the set of states of G is denoted by $S(G)$. For a complete account on matching theory the reader is refered to [9].

To follow the matching theoretic terminology, a *soliton graph* G is defined as a graph having at least one external vertex and a perfect internal matching. (See [1]). A connected soliton graph G is said to be *essentially internal* if either G consists of one edge or every external edge of G is forbidden.

An *elementary component* C of soliton graph G is a maximal connected subgraph of G spanned by allowed edges only. Then C is called *external* or *internal* depending on whether it contains an external vertex or not. An elementary component is said to be *trivial*, if it contains at most one edge. *Elementary graphs* are those which consist of one elementary component. Note that the decomposition into elementary components determines a partition on $V(G)$.

Now let G' be a subgraph of soliton graph G. Then for any state M of G, by $M_{G'}$ we mean the restriction of M to G'. If, in addition, G' is also a soliton graph with $M_{G'} \in S(G')$, and either $Int(G') = V(G') \cap Int(G)$ or G' is essentially internal, then G' will be called a *soliton subgraph with respect to* M.

In a graph G, a *walk* of length n is a sequence $\alpha = v_0, e_1, \ldots, e_n, v_n$, $n \geq 0$, of alternating vertices and edges. This sequence indicates the starting point $v_0 \in V(G)$ of α and the vertex v_j, $j \in [n] = \{1, \ldots n\}$, that α reached after traversing the j-th edge e_j. The notation $\alpha[v_i, v_j]$ with $1 \leq i \leq j \leq n$ will be used for the subwalk of α between v_i and v_j, i.e., $\alpha[v_i, v_j] = v_i, e_{i+1}, \ldots, e_j, v_j$. Furthermore α^{-1} will represent the reverse of α. For every $j \in [n]$, $n_\alpha(j)$ will denote the number of occurences of the edge e_j in the prefix $v_0, e_1, \ldots, e_j$. By a *backtrack* in a walk we mean the traversal of the same edge twice in a consecutive way. However, as the only exception, the traversal of a looping edge in the above way is not considered to be a backtrack. If all edges in a walk are distinct, the walk is called a *trail*, and if, in addition, the vertices are also distinct, the trail is a *path*. We define a *cycle* to be a path together with an edge joining the first and the last vertex. Note, that a looping edge is also a (trivial) cycle according to the above definition. An *external trail (path)* is a trail (path) having an external endpoint , while a path between two external vertices is said to be *crossing*. *Internal trails (paths)* are those that are not external.

A trail $\alpha = v_0, e_1, \ldots, e_n, v_n$, $n \geq 0$ is an *alternating trail* with respect to state M (or M-alternating trail, for short) if for every $i \in [n-1]$, $e_i \in M$ iff $e_{i+1} \notin M$. If $v_0, v_n \in Int(G)$, then α is called *internal*, otherwise α is *external*. Moreover, α is said to be *positive* (*negative*) if either α is internal with $e_1, e_n \in M$ ($e_1, e_n \notin M$, respectively) or it is external with $v_n \in Int(G)$ such that $e_n \in M$ ($e_n \notin M$, respectively). Observe that at most the endpoints of α can be traversed twice by an alternating trail α. Based on the above fact, any maximal external M-alternating trail α starting from vertex v, different from a crossing, can be decomposed in the form $\alpha = \alpha_h + \alpha_c$, where α_h, the *handle of* α, is an external M-alternating path, whereas α_c, the *cycle of* α, is an M-alternating cycle. With these parameters, α is called an *alternating v-racket* or an *alternating v-loop* depending on whether α_c is even or odd.

We say that an internal vertex w is *accessible in state M from external vertex v* (or simply w is *M-accessible from v*) if there exists a positive external M-alternating path with endpoints v and w. We will call an edge e *viable* from external vertex

v in state M if e is traversed by an external M-alternating trail starting from v. *Impervious* edges are those which are not viable in any state from any external vertex. Furthermore, a cycle is said to be *M-accessible* from external vertex v if some of its edges are viable from v in state M.

For a state M of G, an M-alternating trail α is called *complete* if α is either a crossing or it is an even length cycle. An *alternating network* with respect to M (or M-alternating network, for short) is a set of nonempty, pairwise disjoint, complete M-alternating trails. Note that, although an M-alternating network Γ consists of nonempty trails only, the network Γ itself can be empty.

Let M be a state of graph G and α be a complete M-alternating trail. By *making* α in state M we mean exchanging the status of the edges in α regarding their being present or not being present in M, thus creating a new state M'. The state M' created in this way will be denoted by $S_G(M,\alpha)$ or simply $S(M,\alpha)$ if G is understood. Making an M-alternating network Γ in state M means making all the trails of Γ simultaneously in M. Since the trails of Γ do not intersect each other, the resulting state, denoted by $S_G(M,\Gamma)$, is well-defined. Finally, let G' be a subraph of G and $M \in S(G)$. Then by an $M_{G'}$-alternating trail (network) we mean one that is entirely contained in graph G'.

Now we quote two results from [1] related to alternating networks.

Theorem 2.1 *For any two states M_1, M_2 of graph G, there exists a unique mediator alternating network Γ between M_1 and M_2, i.e. $S_G(M_1,\Gamma) = M_2$ and $S_G(M_2,\Gamma) = M_1$*

Corollary 2.2 *An edge e is non-constant iff it is traversed by a complete M-alternating trail in each state M.*

In our decomposition results we will make use of the α_0^ε-products of finite automata, therefore we now recall the necessary definitions from [7]. An *alphabet* is a finite, non-empty set. If X is an alphabet, then X^* denotes the set of words over X, including the empty word ε. A *non-deterministic finite automaton* is a triple $\mathcal{A} = (S, X, \delta)$, where S is a non-empty finite set, the *set of states*, X is an alphabet, the *input alphabet*, and $\delta : A \times X \to 2^A$ is the *transition function*. We can extend δ in such a way that $\delta(s,\varepsilon) = s$ for all $s \in S$.
For $i = 1, 2$, let $\mathcal{A}_i = (S_i, X_i, \delta_i)$ be finite automata. An *isomorphism between $\mathcal{A}_1$ and $\mathcal{A}_2$* is a pair $\psi = (\psi_S, \psi_X)$ of bijective mappings $\psi_S : S_1 \to S_2$ and $\psi_X : X_1 \to X_2$ which satisfies the equation

$$\{\psi_S(s') \mid s' \in \delta_1(s,x)\} = \delta_2(\psi_S(s), \psi_X(x)),$$

for every $s \in S_1$ and every $x \in X_1$. The existence of an isomorphism between $\mathcal{A}_1$ and $\mathcal{A}_2$ is denoted by $\mathcal{A}_1 \cong \mathcal{A}_2$.

Definition 2.3 Let $\mathcal{A}_i = (S_i, X_i, \delta_i)$ $(i = 1, \ldots, k; k > 0)$ be a system of automata. Their *α_0^ε-product* with respect to alphabet X and feedback function ϕ — notation $\prod_{i=1}^k \mathcal{A}_i[X,\phi]$ — is the automaton

$$\mathcal{A} = (S, X, \delta), \text{ where}$$

(a) $S = S_1 \times \ldots \times S_k$

(b) $\phi = (\phi_1, \ldots, \phi_k)$ is a mapping, such that
$\phi_i : S_1 \times \ldots \times S_k \times X \to X_i \cup \{\varepsilon\}$, and ϕ_i is independent of its jth component whenever $i \leq j \leq k$, $(i = 1, \ldots, k)$

(c) $\delta((s_1, \ldots, s_k), x) =$
$$\delta_1(s_1, \phi_1(s_1, \ldots, s_k, x)) \times \ldots \times \delta_k(s_k, \phi_k(s_1, \ldots, s_k, x))$$
for every $x \in X$, $s_i \in S_i$ $(i = 1, \ldots, k)$

Moreover, if every ϕ_i $(1 \leq i \leq k)$ depends only on the input signal, then we speak of the *quasi-direct ε-product* of $\mathcal{A}_1 \ldots \mathcal{A}_k$.

The following definitions are the matching theoretic formalizations of soliton walk and soliton automata introduced in [4].

Definition 2.4 A *partial soliton walk* in graph G with respect to state M is a backtrack-free walk $\alpha = v_0, e_1, \ldots, e_n, v_n$ subject to the following conditions:

(a) v_0 is an external vertex

(b) for every $j \in [n-1]$, $n_\alpha(j)$ and $n_\alpha(j+1)$ have the same parity iff e_j and e_{j+1} are M-alternating, i.e., $e_j \in M$ iff $e_{j+1} \notin M$.

Furthermore if v_n is also external then α is called *total soliton walk*, or simply *soliton walk*.

Note that the case of $n = 0$ is also possible; then the soliton walk is called *trivial*.

Making the walk α in state M means creating $M' = S(M, \alpha)$ by setting for every $e \in E(G)$

$e \in M'$ iff $e \in M$ and e occurs an even number of times in α, or $e \notin M$ and e occurs an odd number of times in α.

In the light of [4, Lemma 3.3] it should be clear that $S(M, \alpha)$ is indeed a state.

In the rest of the paper we will use the following notation. If M is a state of graph G and $v_1, v_2 \in Ext(G)$, then

$\mathcal{S}_G(M, v_1, v_2) = \{S(M, \alpha) \mid \alpha$ is a soliton walk with respect to M, which starts at v_1 and ends at $v_2\}$

Definition 2.5 A *soliton automaton* with underlying graph G is a non-deterministic finite automaton

$$\mathcal{A}(G) = ((S(G), (X \times X), \delta)$$

subject to the following conditions:

(a) G is a soliton graph

(b) $S(G)$, the set of states of $\mathcal{A}(G)$, is the set of states of G

(c) $(X \times X)$ is the input alphabet, where $X = Ext(G)$

(d) $\delta : S(G) \times (X \times X) \to 2^{S(G)}$ is the transition function, such that
$\delta(M, (v_1, v_2)) = \mathcal{S}_G(M, v_1, v_2)$, if $\mathcal{S}_G(M, v_1, v_2) \neq \emptyset$
$\delta(M, (v_1, v_2)) = \{M\}$, otherwise
for any $M \in S(G)$ and $v_1, v_2 \in X$.

A soliton automaton is said to be *elementary* if its underlying object is an elementary graph.

Note that, without loss of generality, we can assume that all constant external edges of a soliton graph G are mandatory. Indeed, attaching an extra mandatory edge to each forbidden external edge of G results in a graph G^* for which $\mathcal{A}(G) \cong \mathcal{A}(G^*)$. We shall use this assumption throughout the paper without any further reference.

In [4] an edge is called *impervious* if it is not traversed by any partial soliton walk. The following proposition states that our definition of impervious edge is equivalent to this.

Proposition 2.6 *Let $\alpha = v_0, e_1, \dots, e_n, v_n$ be a partial soliton walk with respect to state M with $v_0 \neq v_n$. Then there exists an external M-alternating trail β from v_0 such that β terminates in e_n and $E(\beta) \subseteq E(\alpha)$. Furthermore, if $n_\alpha(n)$ is odd, then the other endpoint of β is v_n.*

Proof. First suppose that $e_n \notin M$. Extend G by new external edges $e = (v_{n-1}, v)$ and $e' = (v_n, v')$, such that $v, v' \notin V(G)$. Furthermore let e_m denote the last edge of α for which $e_m = e_n$ and $n_\alpha(m)$ is odd. Observe that $\alpha[v_0, v_{m-1}] + e$ or $\alpha[v_0, v_{m-1}] + e'$ is a total soliton walk in $G + e + e'$ depending on whether $v_{m-1} = v_{n-1}$ or $v_{m-1} = v_n$. Therefore, based on Theorem 2.1, there exists an M-alternating network Γ such that making Γ and making the appropriate part of the above walks results in the same state of $G + e + e'$. Clearly, Γ will contain an M-alternating crossing β' between v_0 and v (between v_0 and v', respectively). Then replacing e (respectively, e') in β' by e_n, we obtain the required M-alternating trail β.

Now consider the case when $e_n \in M$. Then $e_{n-1} \notin M$, thus we can construct the appropriate external alternating trail β described above, which terminates at e_{n-1}. If $e_n \in E(\beta)$, then we are ready. Otherwise $\beta + e_n$ will provide a suitable alternating path. Finally, based on the first part of the proof, observe that $e_n \notin E(\beta)$, when $n_\alpha(n) = n_\alpha(n-1)$ is odd, which makes the proof complete. □

It is clear that impervious edges have no effect on the operations of soliton automata. Thus, without loss of generality, we can restrict our investigation to soliton graphs without impervious edges. The above fact in more precise form is stated in [4, Proposition 4.5]. Therefore, throughout the paper, unless otherwise specified, G will denote a soliton graph without impervious edges.

In the rest of this section we summerize some results from [2].

Definition 2.7 For any two internal vertices $u, v \in V(G)$, $u \sim v$ if u and v belong to the same elementary component of G and the edge $e = (u, v)$ becomes forbidden in $G + e$.

For an elementary component C of G, $\sim_C$ will denote $\sim$ on C separately. Note that generally $\sim_C$ is not equal to the restriction of $\sim$ to C.

Theorem 2.8 *The relation $\sim$ is an equivalence on $Int(G)$.*

The classes of the partition determined by $\sim$ are called *canonical classes*. In particular a *canonical class of elementary component C* is a canonical class contained

in $V(C)$.

Proposition 2.9 *Let u and v be arbitrary vertices of a non-trivial internal elementary component C of G. Then $u \not\sim_C v$ iff for any state M of G there exists a positive internal M_C-alternating path connecting u and v.*

In the following we shall use the phrase "external alternating path γ enters elementary component C" in the strict sense, meaning that γ enters C for the first time. Note that in this case γ must be negative.

Definition 2.10 An internal elementary component C is *one-way* if all external alternating paths enter C in the same canonical class of C. This unique class is called *principal*. Further to this, every external elementary component is a priori one-way by the present definition (with no principal canonical class). An elementary component is *two-way* if it is not one-way.

Proposition 2.11 *There exists no edge connecting two internal vertices contained in principal canonical classes.*

Let C be an elementary component of G, and consider a state M in G. An *M-alternating C-loop* (just C-loop if M is understood) is a negative internal M-alternating path or odd M-alternating cycle in G having both endpoints, but no other vertices, in C. Note that the endpoints of a C-loop α must belong to the same canonical class which is called the *domain* of α. We say that α *covers* the elementary component D if some edge of D is traversed by α.

Definition 2.12 Let M and C be a state and an external elementary component of G, respectively. A *hidden edge* of C is an edge $e = (v_1, v_2)$, not necessarily in $E(G)$, for which v_1, v_2 are the endpoints of an M-alternating C-loop.

An elementary graph C consisting of an external elementary component and its hidden edges will also be considered elementary component throughout the paper. In this case we will call C *augmented external elementary component*. In [2] it was proved that the augmentation of a soliton graph G by its hidden edges preserves the elementary structure of G with the same canonical partition for each elementary component.

The hidden edges have important role in the external alternating paths, which is expressed below.

Proposition 2.13 *Let M be a state of G, $w \in V(G)$ and $v \in Ext(G)$. Furthermore let α be a positive (negative) M-alternating trail between v and w such that $E(\alpha)$ contains hidden edges. Then there exists a positive (respectively, negative) M-alternating trail between v and w which does not traverse any hidden edge.*

Elementary components are structured according to their accessibility by external alternating paths. The rest of this section is an extract of some results obtained in [2] relating to this structure.

Definition 2.14 Let C be an elementary component with a non-principal canonical class P. We say that the couple (C, P) are the *parents* of elementary component D if a C-loop with domain P covers D but there does not exist a C'-loop α for any

elementary component C' such that α covers both C and D. In that case C and P are called the *father* and the *mother* of D, respectively.

Theorem 2.15 *Each two-way elementary component has a unique father and a unique mother. One-way components have no parents.*

The following property of fathers will play important role in the paper.

Proposition 2.16 *Let (C, P) be the parents of elementary component D and let α be an alternating trail starting from external vertex v entering D at a vertex w. Then $\alpha[v, w]$ will go through C such that the last common vertex of $\alpha[v, w]$ and of C belongs to P.*

By Theorem 2.15, elementary components can be grouped into disjoint family trees according to the father-son relationship. Then a *family* $\mathcal{F}$ is defined as a block of elementary components belonging to the same family tree. The root, denoted by $r(\mathcal{F})$, is the ultimate forefather of $\mathcal{F}$. Then Theorem 2.15 implies the following result.

Theorem 2.17 *Every family contains a unique one-way elementary component, which is $r(\mathcal{F})$.*

A family $\mathcal{F}$ is *external* if $r(\mathcal{F})$ is such, otherwise it is internal. Moreover, for the family containing some elementary component C, the notation $\mathcal{F}_C$ will be used.

Now we describe the relationship of families with the help of a binary relation. For this we need the following observation.

Proposition 2.18 *Let e be a forbidden edge of G connecting two different families $\mathcal{F}_1$ and $\mathcal{F}_2$. Then exactly one endpoint of e belongs to the principal canonical class of the root of either $\mathcal{F}_1$ or $\mathcal{F}_2$.*

Making use of the above claim, the binary relation $\mapsto$ is defined in the following way.

Definition 2.19 For any two different families $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_1 \mapsto \mathcal{F}_2$ if there exists an edge e connecting $\mathcal{F}_1$ and $\mathcal{F}_2$ such that the principal endpoint of e is in $\mathcal{F}_2$. In this case we say that e points to family $\mathcal{F}_2$.

Let $\overset{*}{\mapsto}$ denote the reflexive and transitive closure of $\mapsto$.

Theorem 2.20 *The relation $\overset{*}{\mapsto}$ is a partial order on the collection of all families of G, by which the external families are maximal elements.*

Finally we give an important consequence of the above results, which will be used throughout the paper.

Corollary 2.21 *An edge e connecting two families is traversed by any external alternating trail α in G by reflecting the relation $\mapsto$, that is, if α enters family $\mathcal{F}_2$ from family $\mathcal{F}_1$, then e points to $\mathcal{F}_2$.*

Proof. Suppose by way of contradiction that $\mathcal{F}_2 \mapsto \mathcal{F}_1$ holds in the situation described in the statement of the corollary for some M-alternating trail α starting from an external vertex u. In that case let (v_1, v_2) denote the edge traversed by the above way with v_1 being contained in the principal canonical class P_1 of $r(\mathcal{F}_1)$. Now

let extend G by edge $f = (v_1, v_1)$. Then f will be clearly viable by $\alpha[u, v_1] + (v_1, v_1)$, thus $G + f$ has no impervious edges. Observe that P_1 will be a canonical class in $G + f$, too. Indeed, if we assumed that an extra edge g connecting two vertices $w_1, w_1' \in P_1$ would be allowed in $G + f + g$, then it is easy to see, by making use of Corollary 2.2, that there would exist in $G + f + g$ a complete M-alternating trail β containing both f and g. However, f is a loop, consequently the above situation is not possible. Therefore the elementary component $r(\mathcal{F}_1) + f$ is also one-way in $G + f$ with f being in its principal canonical class P_1; which is a contradiction in Proposition 2.11. □

3 The decomposition of soliton automata

It is a central question to establish the correspondence between alternating networks and soliton walks. The first result gives the characterization of this problem.

Definition 3.1 Let v, w be external vertices and M be a state of G. An *M-transition network Γ from v to w* is an M-alternating network with the following conditions:

(a) If $\Gamma = \emptyset$, then $v = w$.

(b) All elements of Γ, except one crossing from v to w if $v \neq w$, are alternating cycles accessible from v in M.

Let $\mathcal{T}_G(M, v, w)$ denote the set of M-transition networks from v to w in graph G.

Theorem 3.2 *Let M be a state of G, $v, w \in Ext(G)$, and Γ be an M-alternating network. Then $S_G(M, \Gamma) \in \mathcal{S}_G(M, v, w)$ iff $\Gamma \in \mathcal{T}_G(M, v, w)$.*

Proof. The "only if" part is straightforward from Theorem 2.1 and from Proposition 2.6. To prove the "if" part, let us construct for each cycle β of Γ an appropriate v-racket β' with respect to M, such that the length of the handle of β' is minimal. Let Γ' denote the set of the above v-rackets and , in the case of $v \neq w$, of the crossing of Γ.

We will show by an inductive argument on $\mid \Gamma' \mid$ that there exists a soliton walk α for which $E(\alpha) \subseteq E(\cup\Gamma')$ and $S(M, \alpha) = S(M, \Gamma)$. The basis step with Γ' being empty or a singleton is trivial.

Now let $\mid \Gamma' \mid > 1$ and assume that the assertion holds for each soliton walk set Γ_1' constructed in the described way from an appropriate alternating network Γ_1 with $\mid \Gamma_1' \mid < \mid \Gamma' \mid$. Let γ denote the v-racket with longest handle in Γ'. It is evident that γ_c is disjoint from $\cup(\Gamma' \backslash \{\gamma\})$. Now using the induction hypothesis consider a soliton walk $\alpha = v, e_1, v_1, \ldots, v_{n-1}, e_n, w$ traversing $\Gamma' \backslash \{\gamma\}$ by the required way. If $\gamma_h = v, f_1, w_1, \ldots, f_m, w_m$, then let w_i be the first vertex of γ_h such that $f_{k+1} \neq e_{k+1}$. Then it is easy to see that $\alpha[v, v_i] + \gamma_h[w_i, w_m] + \gamma_c + \gamma_h^{-1}[w_m, w_i] + \alpha[v_i, w]$ will be a soliton walk with the required properties, which makes the proof complete. □

Note that based on Theorem 3.2, the transitions of a soliton automaton can be effectively computed from its underlying soliton graph. Indeed, for any two states

M_1, M_2, the mediator alternating network between M_1 and M_2 is given by $\Gamma = M_1 \oplus M_2$, where $\oplus$ denotes the symmetric difference of M_1 and M_2, while the shortest handle for each alternating cycle in Γ can be found in a straightforward way. Moreover, the proof of Theorem 3.2 is constructive, it yields a soliton walk between two given states. The above facts are important from simulation point of view.

We will work out products of automata based on elementary components, thus first we characterize the automata constructed from these components.

Definition 3.3 For an elementary component C of graph G, the *component automaton* determined by C is the soliton automaton based on the graph C^*, where
$C^* = C$, if C is external
$C^* = C + (v, w), v \in C, w \notin V(G)$, if C is internal

Definition 3.3 might give the impression that an internal component automaton depends on the choice of vertex v. However, Theorem 3.5 will show that all component automata determined by the same internal elementary component are isomorphic, thus $\mathcal{A}(C^*)$ is unambiguous.

Definition 3.4 An $\mathcal{A} = (S, X, \delta)$ automaton is called *full*, if

(i) $X = \{x\}$
(ii) $\delta(s, x) = S$, for each $s \in S$

Theorem 3.5 *Every internal component automaton is a full automaton. Conversely, for any full automaton there exists an isomorphic internal component automaton.*

Proof. We start with proving the first statement. To this end let C be an internal elementary component, $v \in V(C)$ and (v, w) an extra external edge attached to C in order to form C^*. As any state of C^* has a transition to itself by a trivial soliton walk, we have to prove the "full-property" only for any two different states M_1,M_2 of C. If Γ is the mediator alternating network between M_1 and M_2, then clearly Γ consits of $M_1(M_2)$-alternating cycles. Any cycle β of Γ contains a vertex u for which $u \not\sim_C v$, thus there exists an internal positive $M_1(M_2)$-alternating path α between u and v in the graph C. Therefore β is accessible from w in $M_1(M_2)$ by $(w, v) + \alpha$. As v and β were arbitrary, we obtain the first claim with the help of Theorem 3.2.

To prove the second statement, we only have to show that there exists an internal elementary component with n states for every $n \in N$. The case $n = 1$ is satisfied by an elementary component consisting of one internal mandatory edge. If $n \geq 2$, then consider an even cycle β, two adjacent vertices $v, w \in V(\beta)$ and construct a graph G such that it has a representation in the form $G = \beta + \alpha_1 + \ldots + \alpha_{n-2}$, where

(i) α_i, $i \in [n-2]$ is an odd path with endpoints v and w
(ii) $V(\alpha_i) \cap V(\beta) = \{v, w\}$, $i \in [n-2]$
(ii) $V(\alpha_i) \cap V(\alpha_j) = \{v, w\}$, $i, j \in [n-2]$

Observe that for any edge e being incident with v, there is a unique state M of G such that $e \in M$. Thus, it is easy to see, that each edge of G is allowed and G has n states, as expected. □

For the description of the product automaton we need the following concepts.

Definition 3.6 Let P be a canonical class of some external component. Then the set ρ_P is the smallest set of elementary components such that:

(i) if C' is an internal elementary component and (v, w) is an edge for which $v \in P$ and $w \in V(C')$, then $C' \in \rho_P$.

(ii) if C_1, C_2 are internal elementary components such that $\mathcal{F}_{C_1} \overset{*}{\mapsto} \mathcal{F}_{C_2}$, $C_1 \in \rho_P$ and there is an edge between C_1 and C_2, then $C_2 \in \rho_P$.

Note that (ii) may also hold if C_1 and C_2 are in the same family, as $\overset{*}{\mapsto}$ is reflexive. Moreover, based on the structure of the families it can be easily showed, that if $C \in \rho_P$ and $\mathcal{F}_C$ is internal, then $C' \in \rho_P$ for any elementary component C' of $\mathcal{F}_C$.

For the main result of the section we introduce some technical notations and prove a lemma. For these we need the following simple observation.

Claim 3.7 *Let P be a canonical class of some elementary component C. An internal vertex of P is accessible from external vertex v in state M iff all vertices of P are M-accessible from v .*

Proof. Let us assume that α is a positive external M-alternating path from v to w and let u be an arbitrary vertex of P different from w. We claim that there exists an internal M-alternating path β between u and some vertex of α such that β is positive on the end of vertex u. If C is external, then according to [2, Proposition 2.3] there exists a positive external M_C-alternating path γ with endpoint u. Observe that $E(\alpha) \cap E(\gamma) \neq \emptyset$, because otherwise $\alpha' = \alpha + (w, u) + \gamma$ would form an alternating crossing indicating that $u \not\sim w$ by $S(M, \alpha')$. Therefore an appropriate subpath of γ is suitable for β. Now assume that C is internal. Then let w' denote the vertex incident with w by the edge covered by M. Clearly, $u \not\sim_C w'$, thus, based on Proposition 2.9, there exists a positive internal M_C-alternating path between u and w', from which the existence of β is straightforward again.

Now starting from u let u_α denote the first vertex along β for which $u_\alpha \in V(\alpha)$. $\alpha_1 = \alpha[w, u_\alpha] + \beta[u_\alpha, u]$ cannot form a positive internal alternating path, as it would contradict $u \sim w$. Therefore $\alpha[v, u_\alpha] + \beta[u_\alpha, u]$ gives a positive external M-alternating path, as desired. □

By Claim 3.7 it is justified to say that a *canonical class is accessible* from an external vertex in a given state.

For any internal elementary component C' of graph G:

$$\mathcal{R}_G(C') = \{P \mid P \text{ is a canonical class of some external elementary component and } C' \in \rho_P\}$$

For any external vertex v of a (possibly augmented) external elementary component C and state M of C in graph G:

$$\mathcal{P}_C(M,v) = \{P \mid P \text{ is a canonical class of } C, \text{ which is } M\text{-accessible from } v \text{ in the graph } C\ \}.$$

and

$$\mathcal{C}_G(M,v) = \{C' \mid C' \text{ is an internal elementary component such that } \mathcal{R}_G(C') \cap \mathcal{P}_C(M,v) \neq \emptyset.\}$$

Note that if G is understood then the subscript G is omitted from the above notations. Furthermore, if C is an augmented external elementary component, then it is indicated with a superscript 'h' , i.e. using $\mathcal{C}^h(M,v)$.

Lemma 3.8 *Let P' be a non-principal canonical class of some internal elementary component C' and v be an external vertex of an elementary component C. Then an edge e incident with a vertex of P' is viable from v in state M iff $C' \in \mathcal{C}^h(M_C,v)$.*

Proof. During the proof the notation C^h will be used for the augmented external elementary component constructed from C. Furthermore, for any external alternating trail α starting from C, w_α will denote the last vertex of α for which $w_\alpha \in V(C)$.

'Only if' Let α be a positive external M-alternating trail starting from v and terminating at vertex w, where w is an endpoint of e. Moreover, let P_α denote the canonical class containing w_α. Then substituting the C-loops for hidden edges in α, we obtain that $P_\alpha \in \mathcal{P}_{C^h}(M_C,v)$. Now using Corollary 2.21, it is easy to see that $C_s \in \rho_{P_\alpha}$ for each internal elementary component C_s reached by $\alpha[w_\alpha, w]$. Hence $P_\alpha \in \mathcal{R}(C') \cap \mathcal{P}_{C^h}(M_C,v)$, which gives the result.

'If' Suppose that $C' \in \rho_P$ for some canonical class $P \in \mathcal{P}_{C^h}(M_C,v)$. Then based on the definition of ρ_P there exist families $\mathcal{F}_1, \dots, \mathcal{F}_m$ containing members of ρ_P such that $\mathcal{F}_1 = \mathcal{F}_C$, $\mathcal{F}_m = \mathcal{F}_{C'}$ and for each $1 \leq s \leq m-1$ $\mathcal{F}_s \mapsto \mathcal{F}_{s+1}$ with some edges connecting elements of $\rho_P \cap \mathcal{F}_s$ and $\rho_P \cap \mathcal{F}_{s+1}$. Let α be an external M-alternating trail terminating at w, where w is an endpoint of e. Note that such an α exists, because [1, Corollary 3.3] states that an edge is impervious in one state iff it is impervious in all states. The proof will apply an induction on m.

Basis step. Applying Theorem 2.15 iteratively, we obtain that each two-way elementary component C_1 of $\mathcal{F}_1$ has a unique ultimate foremother – in notation $m(C_1)$– as a class of C. Then, making use of Proposition 2.16, it is clear that for any external M-alternating trail β reaching C_1, w_β is contained in $m(C_1)$.

It is clear, by Proposition 2.18, that $\rho' = \rho_P \cap \mathcal{F}_1$ can be built up iteratively according to Definition 3.6 (i) – (ii). We will show by a structural induction based on the building procedure of ρ', that for any elementary component $C_1 \in \rho'$, $m(C_1) = P$ holds. First suppose that C_1 is added to ρ' in a step of type (i). As $P \in \mathcal{P}_{C^h}(M_C,v)$, we obtain with the help of Proposition 2.13, that in this case $w_\gamma \in P$ holds, which implies $m(C_1) = P$ by the previous paragraph. Continuing the procedure with (ii) such that edge e connects C_1 with an elementary component C_2 already in ρ', let us consider an external alternating trail γ terminating at e. According to the hypothesis for C_2, w_γ must belong to P. Thus applying the observation of the previous paragraph again, we obtain that $m(C_1) = P$, as desired.

Summarizing the foregoings we conclude that $w_\alpha \in P$. Now choosing a positive M_C-alternating path α_1 between v and w_α and applying Proposition 2.13 for $\alpha_1 + \alpha[w_\alpha, w]$ we obtain a suitable alternating trail.

Induction step. Let u denote the first vertex of α which is also in $C_m = r(\mathcal{F}_m)$. Moreover, let u' denote a vertex of C_m which is connected by an edge to a vertex w' of some elementary component of $\mathcal{F}_{m-1} \cap \rho_P$. According to the induction hypothesis there exists an appropriate M-alternating trail β running from v and terminating at (u', w'). Based on Corollary 2.21 the following facts hold: the internal endpoint of β is u' such that β is a negative path, $\alpha[u, w]$ avoids $\mathcal{F}_1, \ldots, \mathcal{F}_{m-1}$ and $\beta - (w', u')$ does not "touch" $\mathcal{F}_m$. Now consider a positive internal M_C alternating path γ, which starts from u' and terminates in some vertex u_1 of $V(\alpha) \cap V(C_m)$ such that $u' \not\sim_C u_1$. By Corollary 2.21, $u \sim u'$, therefore if u_1' denotes the vertex where γ hits α first time, then we can conclude that $\beta + \gamma[u', u_1'] + \alpha[u_1', w]$ provides the desired alternating trail. □

Theorem 3.9 *Let $C_1, \ldots, C_l$ be the augmented external elementary components, $C_{l+1}, \ldots, C_k$ be the internal elementary components of G, and $\mathcal{A}(C_i^*) = (S(C_i), (X_i \times X_i), \delta_i)$ $(i = 1, \ldots, k)$, with $X_i = \{x_i\}$, if $i > l$. Then:*

$$\mathcal{A}(G) \cong \mathcal{A}^*(G), \text{ where}$$

$$\mathcal{A}^*(G) = \textstyle\prod_{i=1}^k \mathcal{A}(C_i^*)[Y, \phi] \text{ is an } \alpha_0^\varepsilon\text{-product such that}$$

(a) $Y = (Ext(G) \times Ext(G))$

(b) $\phi = (\phi_1, \ldots, \phi_k)$ *is defined in the following way:*

For each $1 \le i \le k$, $M_1 \in S(C_1), \ldots, M_k \in S(C_k)$ *and* $(y_1, y_2) \in Y$

(b/1) if $1 \le i \le l$ *and* $(y_1, y_2) \in X_i \times X_i$, *then*

$$\phi_i(M_1, \ldots, M_k, (y_1, y_2)) = (y_1, y_2)$$

(b/2) if $l + 1 \le i \le k$, $(y_1, y_2) \in X_j \times X_j$ *for some* $1 \le j \le l$, $C_i \in \mathcal{C}^h(M_j, y_1)$, *and either* $y_1 = y_2$, *or* $y_1 \ne y_2$ *with* $\delta_j(M_j, (y_1, y_2)) \ne \{M_j\}$, *then*

$$\phi_i(M_1, \ldots, M_k, (y_1, y_2)) = (x_i, x_i)$$

(b/3) Otherwise:

$$\phi_i(M_1, \ldots, M_k, (y_1, y_2)) = \varepsilon.$$

Proof. Let δ and δ^* denote the transition function of $\mathcal{A}(G)$ and that of $\mathcal{A}^*(G)$, respectively. Moreover, let $(y_1, y_2) \in Y$ and $M \in S(G)$ be arbitrary, such that $y_1 \in V(C_r)$, $y_2 \in V(C_s)$ for some $r, s \le l$. Since the mapping

$$\psi(M) = (M_{C_1}, \ldots, M_{C_k}) \tag{1}$$

is clearly a bijection between $S(G)$ and $S(C_1) \times \ldots S(C_k)$, we only have to prove that

$$\{\psi(M') \mid M' \in \delta(M, (y_1, y_2))\} = \delta^*(\psi(M), (y_1, y_2)) \tag{2}$$

For each $1 \le i \le k$ let z_i denote $\phi_i(M_{C_1}, \ldots, M_{C_k}, (y_1, y_2))$. Consider first the right side of (2). Then based on (1) and Definition 2.3, we have

First we provide a characterization of non-trivial self-transitions by alternating trails. For this result we need the following definition.

Definition 4.2 Let M be a state of G and $v \in Ext(G)$. An *M- alternating double v-racket* α is a pair of M-alternating v-rackets (α^1, α^2) with branching handles, i.e. with neither of α_1^h and α_2^h being a prefix of the other. The maximal common external subpath – denoted by α_h – of α_h^1 and of α_h^2 is called the *handle* of α, whereas the last vertex of α_h is referred to be as the *branching vertex* of α.

Note that the handle of a double v-racket is a positive external alternating path.

Theorem 4.3 *There exists a non-trivial self-transition of external vertex v with respect to state M of G, iff G contains either an M-alternating v-loop or an M-alternating double v-racket.*

Proof. During the proof if we refer to an alternating cycle α as a part of a decomposed form of a soliton walk β, then we mean that α as a subwalk of β is traversed in an appropriate way.

For an M-alternating v-loop α it is easy to check that $\alpha_h + \alpha_c + \alpha_c + \alpha_h^{-1}$ is a non-trivial self-transition of v. Therefore, we can suppose for the rest of the proof that G does not contain an M-alternating v-loop .

'Only if' Let $\alpha = v, e_1, v_1, \ldots, e_n, v_n$ be a non-trivial self-transition of v with respect to M, and let i be the smallest index for which there exists an index $j > i$ such that $v_j = v_i$, $n_\alpha(j) = 1$ and each edge of $\alpha[v, v_i]$ is traversed exactly once by $\alpha[v, v_j]$. In other words, v_i is the closest vertex to v where α returns to itself. Now, based on Proposition 2.6, there exists an M-alternating trail β such that β terminates at e_j and $E(\beta) \subseteq E(\alpha[v, v_j])$. By assumption, β is an alternating v-racket with $\beta_h = \alpha[v, v_i]$. Observe that $\alpha[v, v_j] + \alpha[v, v_i]^{-1}$ is a soliton walk from v to itself. Therefore, it is obvious that the edges traversed by $\alpha[v_i, v_j]$ an odd number of times will constitute an M-alternating network Γ consisting of alternating cycles. By the above facts we obtain that $e_{j+1} = e_i$. The edge e_j must be traversed by $\alpha[v_j, v_n]$, consequently there is a first edge e_m with $m > j$ which is not on $\alpha[v, v_i]$. Then, let e_r denote the edge for which $e_r = e_{m-1}$ with $r \leq i$. It is easy to see, that because of the choice of v_i, any vertex v_l with $l < i$ is incident with exactly two edges of $\alpha[v, v_j]$. Therefore $n_\alpha(m) = 1$ and we can select the first edge e_k of $\alpha[v_{m-1}, v_n]$ for which $n_\alpha(k)$ is even. Again, by the choice of v_i, we conclude that $\alpha[v, v_{r-1}]$ and $\alpha[v_{m-1}, v_k]$ are edge-disjoint. Furthermore, observe that $e_r \notin M$, therefore $\alpha' = \alpha[v, v_{r-1}] + \alpha[v_{m-1}, v_{k-1}]$ is a partial soliton walk with respect to M. As we have seen, there exists an M-alternating cycle γ' of Γ containing e_k. Making use of the former observations for Γ and for α' we obtain that γ' and α' are edge-disjoint. Now applying Proposition 2.6 for α', an M-alternating v-racket γ can be constructed such that $\delta = (\beta, \gamma)$ is a double v-racket with $\gamma_c = \gamma'$ and $\delta_h = \alpha[v, v_{r-1}]$.

'If' Let $\alpha = (\alpha^1, \alpha^2)$ be a double v-racket, and let w denote the branching vertex of α. Moreover, let us introduce the notation $\alpha_s^i = \alpha_h^i - \alpha_h$ and $\alpha_w^i = \alpha_s^i + \alpha_c^i + (\alpha_s^i)^{-1}$ for $i = 1, 2$. If $\alpha^2 - \alpha_h$ is edge-disjoint from α^1 we obtain that $\alpha_h + \alpha_w^1 + \alpha_w^2 + \alpha_w^1 + \alpha_w^2 + \alpha_h^{-1}$ is a soliton walk with the desired properties.

Otherwise let u be the first vertex of α_w^2 which is also on α^1 and extend $\alpha^2[w,u]$ to an M-alternating path α_u by continuing its way appropriately on α_h^2 until it reaches α_c^2. Note that $\alpha_u = \alpha^2[w,u]$ holds if $u \in V(\alpha_c^2)$. Also note that the construction described above is feasible, as G has no M-alternating v-loops. Then $\alpha_h + \alpha_w^1 + \alpha_u + \alpha_c^1 + \alpha_u^{-1} + \alpha_h^{-1}$ will result in the requested soliton walk. □

We now turn to the characterization of v-loops.

Proposition 4.4 *Let v be an external vertex of graph G and $M \in S(G)$. Then G contains an M-alternating v-loop iff there exists an internal edge (u,w) such that both u and w are accessible from v in M.*

Proof. It is sufficient to prove the 'If' part. Let α and β be positive external M-alternating paths from v to internal vertices u and w, respectively, such that $(u,w) \in E(G)$ and $\mid E(\alpha) \cup E(\beta) \mid$ is minimal. Then let w_β denote the last vertex of β with $w_\beta = u_\alpha$ for some $u_\alpha \in V(\alpha)$. We claim that $\alpha[v,u_\alpha]$ is positive. Indeed, otherwise both endpoints of the last edge of $\alpha[v,u_\alpha]$ would be accessible from v in M by the appropriate subpaths of α and β, which would be a contradiction in the choice of u and w. Therefore an M-alternating v-loop can be formed from the edges of the set $E(\alpha) \cup E(\beta[w_\beta,w]) \cup \{(u,w)\}$, as desired. □

To state the following important consequence of Proposition 4.4, let us call two states M_1, M_2 *compatible* if M_1 and M_2 cover the same external edges.

Corollary 4.5 *Let M_1 and M_2 be compatible states of G and $v \in Ext(G)$. Then G contains an M_1-alternating v-loop iff G contains an M_2-alternating v-loop.*

Proof. The role of M_1 and M_2 is symmetric, so we need to prove one direction only. To this end let (v_1,v_2) be an edge of the cycle of an M_1-alternating v-loop α and let α_1 and α_2 denote the appropriate positive external M_1-alternating subpaths of α running to vertices v_1 and v_2, respectively. According to Theorem 2.1, there exists a mediator alternating network Γ between M_1 and M_2 containing only alternating cycles. Clearly, we can suppose without loss of generality that Γ consists of one M_1-alternating cycle β. We claim that for α_i, $i = 1,2$, either α_i is accessible from v in M_2 or an M_2-alternating v-loop can be formed from the edges of $E(\alpha_i) \cup E(\beta)$. If the latter case holds for at least one of α_1 and α_2, then we are ready. Otherwise the Corollary can be obtained by Proposition 4.4.

Our claim is obviously enough to be proved for α_1 with the assumption that $E(\alpha_1) \cap E(\beta) \neq \emptyset$. Let w and w' denote the first and the last vertex of α_1 which are also in $V(\beta)$. If w and w' are in odd distance on β, then the requested M_2-alternating path is obtained by combining $\alpha[v,w]$, the positive M_2-alternating subpath of β between w and w', and $\alpha[w',v_1]$. Otherwise it is easy to see that there must exist a subpath α' having its endpoints x and y, but no other vertices, in $V(\beta)$ such that both x and y are in an odd distance from w on β . This allows an M_2-alternating v-loop to be constructed from $\alpha[v,w]$, α' and an appropriate M_2-alternating subpath of β. Hence the proof is complete. □

For a further analysis of v-loops we introduce the graph C_v^M, where C is an external elementary component of G containing external vertex v and $M \in S(G)$. The

graph C_v^M is the subgraph of C spanned by the edges that are M_C-viable from v in the subgraph C. Moreover, let $\mathcal{C}_v^M$ denote the set $\mathcal{C}(M_C, v) \cup \{C_v^M\}$. Finally, G_v^M will denote the graph which consists of $\cup\mathcal{C}_v^M$ plus the edges connecting different elements of $\mathcal{C}_v^M$.

Note that generally G_v^M is not equal to the graph $G[V(\cup\mathcal{C}_v^M)]$. Moreover, we might have the impression that G_v^M contains all the edges M-viable from v. However, by Lemma 3.8 and by Proposition 2.13, it is easy to see that the above fact is true iff $\mathcal{C}(M_C, v) = \mathcal{C}^h(M_C, v)$ and any edge of C M_C-viable from v in the augmentation of C is also M_C-viable from v in C.

Proposition 4.6 *G_v^M is a soliton subgraph with respect to M. Furthermore, the set of elementary components of G_v^M is $\mathcal{C}_v^M$.*

Proof. The first sentence is evidently tru if C is a trivial component. Furthermore, if C is non-trivial, then any maximal M-alternating trail starting from v is entirely contained in C_v^M, which implies that G_v^M is indeed a soliton subgraph with respect to M. To verify the second sentence, observe that if an edge of a soliton subgraph G' of G is forbidden in G, then it is also forbidden in G'. For this reason, all we have to prove is that C_v^M is elementary. To this end we will make use of the following two claims.

Claim A *If an edge of C_v^M is part of an even M-alternating cycle α of G, then α is enterily contained in C_v^M.*

Proof. Straightforward.

Claim B *Any edge of C_v^M traversed by an M-alternating crossing in G is in the unique external elementary component of C_v^M.*

Proof. It is clear that C_v^M has a unique external elementary component. Let $\alpha = v_0, e_1, \ldots, e_n, v_n$ be a fixed M-alternating crossing and e_i be an arbitrary edge of α which is also in C_v^M. Furthermore let β be an external M-alternating trail starting from v and terminating at e_i such that $k = \mid E(\beta)\backslash E(\alpha) \mid$ is minimal. We will prove the claim by induction on k. The basis step $k = 0$ is trivial. For the induction step, consider the last edge e_β of β not on α and let w denote the endpoint of e_β contained in $V(\alpha)$. We can assume without loss of generality that $w = v_j$ with $j < i$. Then clearly $e_{j+1} \in M$. If $\beta[v, w]$ does not overlap with $\alpha[v_i, v_n]$, then the crossing $\beta[v, w] + \alpha[v_j, v_n]$ does the job. Otherwise, let v_k be the first vertex of $\alpha[v_i, v_n]$ incident with an edge e of $E(\beta)\backslash E(\alpha)$ and let u denote the vertex of β with $u = v_k$. Observe that, starting from v, β must go through e before reaching u. Indeed, if not, then – since $e_{k+1} \in M$ – $\beta[v, u] + \alpha[v_k, v_{i-1}]^{-1}$ would contradict the choice of β. Therefore $\alpha' = \beta[u, w] + \alpha[v_j, v_k]$ will form an even M-alternating cycle, which shows by Claim A that e_i and e_{k+1} are in the same elementary component of C_v^M. Finally, by applying the induction hypothesis for e_{k+1} we obtain Claim B. □

Continuing the proof of Proposition 4.6, let us suppose by way of contradiction that C_v^M has an internal elementary component C'. Then, there must exist an allowed edge e of C having exactly one endpoint in C'. Let f denote the edge of C' incident

with e such that $f \in M$. Clearly $e \notin M$, consequently, by Corollary 2.2, a complete M-alternating trail α must go through e. Applying Claim A and Claim B for f and α, we obtain a contradiction, which makes the proof complete. □

Corollary 4.7 *Each edge of G_v^M is viable from v in $M_{G_v^M}$.*

Proof. Based on Proposition 4.6, we have $\mathcal{C}_{G_v^M}(M_{C_v^M}, v) = \mathcal{C}_G(M_C, v)$, i.e. $\mathcal{C}_{G_v^M}(M_{C_v^M}, v)$ contains all elementary components of G_v^M which are different from C_v^M. Then the claim is obtained with the help of Lemma 3.8. □

Proposition 4.8 *For any external vertex v of G, there exists an alternating v-loop with respect to state M iff G_v^M is non-bipartite.*

Proof.

'Only if' Let C be the elementary component containig v, and α be an M-alternating v-loop. If each edge of α is also contained in C_v^M, then we are ready. Otherwise, starting from v, let w be the first vertex of α such that an appropriate subpath α' of α forms a C-loop with one of its endpoints being w. Then, it is easy to see with the help of Corollary 2.21, that each edge of $\alpha[v, w]$ is contained in G_v^M. Therefore α' is also a C-loop in G_v^M, consequently, because of Claim 3.7, both endpoints of α' are $M_{G_v^M}$-accessible from v. Finally, applying Proposition 4.4 for the endpoints of the last edge of $\alpha[v, w]$, we obtain that G_v^M has a v-loop, indicating that it is non-bipartite.

'If' Let us suppose by way of contradiction that G_v^M does not contain M-alternating v-loops. Then let G' denote a maximal bipartite soliton subgraph of G_v^M with respect to $M_{G_v^M}$ such that $v \in V(G')$ and each edge of G' is viable from v in $M_{G'}$. Note that such a subgraph G' exists under our assumption, because any maximal external alternating trail starting from v as a v-racket or a crossing from v has the required properties. Based on Corollary 4.7, there exists a maximal external $M_{G_v^M}$-alternating trail β from v to some vertex v' traversing an edge not in G'. Let e denote the first edge of β not in $E(G')$. Moreover, let w be the endpoint of e belonging to $V(G')$ with A being the bipartition class of G' containing w. Observe that $E(\beta[w, v']) \cap E(G') \neq \emptyset$ and starting from w, the first overlap will occur at a vertex u in A. Indeed, checking any other possible cases, because of $G' + \beta[w, v']$, we would obtain a contradiction with the choice of G'. Furthermore, every edge is viable from v in $M_{G'}$, consequently there exists an $M_{G'}$-alternating trail γ from v to u. Observe that γ is also positive, as the parity of the length of γ and that of $\beta[v, w]$ must be equal because of the bipartition of G'. Finally, applying Proposition 4.4 for any edge of $\beta[w, u]$, we obtain a contradiction. Hence the proof is complete. □

Considering double v-rackets too, we can describe non-trivial self-transitions via the elementary structure of soliton automata. We also obtain that, similarly to Theorem 3.9, the problem can be reduced to elementary automata. For this final result we introduce the following concept.

Definition 4.9 Let $\{C_1, \ldots, C_n\}$ be the set of the elementary components of G with C_1 being external. G is a *component-chain graph* if it can be decomposed in

the *chain-form* $G = C_1 + (w_1, v_2) + C_2 + (w_2, v_3) + \ldots + (w_{n-1}, v_n) + C_n$ such that for each $2 \leq i \leq n-1$, $(w_{i-1}, v_i), (w_i, v_{i+1}) \in E(G)$ with the vertices v_i and w_i belonging to different canonical classes of C_i.

We shall be interested in situations when G_v^M is a component-chain graph for some graph G with external vertex v and $M \in S(G)$. In that case we augment Definition 4.9 by taking $v_1 = v$. We will call a (external or internal) positive M_{C_i}-alternating path *M^v-transit* if it connects v_i and w_i. Component C_i is said to be *M^v-transit* if $i \neq n$ and either C_i has two different M^v-transit paths or there exists an even M_{C_i}-alternating cycle disjoint from the unique M^v-transit path of C_i. Finally, C_i is called an *M^v-terminal* if $i = n$ and C_i' has an M_{C_i}-alternating double w-racket, where either $C_i' = C_i$ with $w = v$ or $C_i' = C_i + (w_i, v_{i-1})$ with $w = v_{i-1}$ depending on whether $n = 1$ or not.

Theorem 4.10 *Let $\mathcal{A}(G)$ be a soliton automaton, M be a state of $\mathcal{A}(G)$ and C be an elementary component of G containing external vertex v. Then, there exists a non-trivial self-transition of v with respect to M, iff one of the following conditions holds.*

(i) G_v^M is not a bipartite component-chain graph.

(ii) G_v^M is a bipartite component-chain graph having an M^v-transit or an M^v-terminal elementary component.

Proof.

'Only if' Based on Theorem 4.3 and Proposition 4.6, it is enough to prove that if G contains an M-alternating double v-racket $\alpha = (\alpha^1, \alpha^2)$ such that G_v^M is a bipartite component-chain graph, then *(ii)* holds. To this end, first we claim that in this case α is entirely contained in G_v^M. Indeed, if, on the contrary, e denotes the first edge of α^1 (or α^2) which is not in $E(G_v^M)$, then, based on Corollary 2.21 and the definition of G_v^M e must connect two elementary components belonging to $\{C\} \cup \mathcal{C}(M_C, v)$. Then, clearly, one of the endpoints of e, denoted by w, will be contained in $V(C)$. However, it is a contradiction in the choice of e, because an appropriate subpath of $\alpha[v, w]$ will be a C-loop between w_1 and w, which implies that w is also in the unique canonical class of $\mathcal{P}_C(M_C, v)$, consequently e should be contained in G_v^M.

Therefore let us consider the chain form $G_v^M = C_1 + (w_1, v_2) + \ldots + (w_{n-1}, v_n) + C_n$ with $C_1 = C_v^M$ and $v_1 = v$. Let C_i and C_j, $1 \leq i, j \leq n$, denote the elementary components containing α_c^1 and α_c^2, respectively. Furthermore, let α_l^k, with $k = 1, 2$ and $l = i, j$, denote the subtrail of α^k running entirely in C_l, whereas the notation (A_i, B_i) will be used for the bipartition of C_i with $w_i \in B_i$. We may suppose without loss of generality that $i \leq j$. Now consider the elementary component C_k containing the branching vertex of α. If $k < i$, then it is easy to see that C_k is M^v-transit. Therefore we may suppose for the rest of the proof that $i = k$. Then we distinguish two cases.

Case (a) $i < j$. Then α_i^2 is an M^v-transit path. Therefore, we are ready, if α_i^2 is disjoint from α_c^1. Otherwise, let u' denote the first vertex of α_i^1 incident with an edge of $E(\alpha_i^1) \backslash E(\alpha_i^2)$. Then $u' \neq w_i$, because it is easy to check that α_i^2 is not a subpath of α_i^1. Thus continuing α_i^1 from u', there will be a first vertex u'' of the

appropriate subtrail of α_i^1 which is also in $V(\alpha_i^2)$. Now it can be easily observed that $u' \in B_i$ and $u'' \in A_i$, therefore the edges of $E(\alpha_i^2) \cup E(\alpha_i^1[u', u''])$ form two M^v-transit paths, as desired.

Case (*b*) $i = j$. If $i = n$, then C_i is clearly M^v-terminal, thus we are ready. For other alternatives we will prove that C_i is M^v-transit. To this end let β_i be an M^v-transit path of C_i. Furthermore, starting from v_i let u_i denote the last vertex of β_i which is also in $V(\alpha_i^1) \cup V(\alpha_i^2)$. The role of α_i^1 and α_i^2 is symmetric, thus we can assume that $u_i = u$ for some vertex u of α_i^1. Obviously, $\alpha' = \alpha_i^1[v_i, u] + \beta_i[u_i, w_i]$ is an M^v-transit path, because u_i must belong to B_i. Now following the same argument for α' and α_i^2 which was applied in the proof of Case (*a*), we obtain the claim.

'If' By Theorem 4.3 and Proposition 4.8, it is sufficient to prove that a bipartite G_v^M contains an $M_{G_v^M}$-alternating double v-racket, if (*i*) or (*ii*) holds. In this case observe that each family of G_v^M is singleton. Indeed, if family $\mathcal{F}$ is not a singleton, then there must exist an M_1-alternating C'-loop β connecting vertices $v_1, v_2 \in V(C')$, where $M_1 \in S(G_v^M)$ and $C' = r(\mathcal{F})$. It was proved in [1] that any two vertices of an elementary graph is contained in a common complete alternating trail. Consequently, there exists for some $M' \in S(C')$ a complete M'-alternating trail γ traversing both v_1 and v_2. The length of $\gamma[v_1, v_2]$ is clearly even, thus $\beta + \gamma[v_1, v_2]$ indicates that G_v^M is non-bipartite, which contradicts our assumption.

Therefore, if (*i*) holds, then there must be elementary components C_1, C_2, C_3 of G_v^M such that $\mathcal{F}_{C_1} \mapsto \mathcal{F}_{C_i}$ for $i = 2, 3$ by two different edges $e_2 \neq e_3$. Then, as we have seen in the proof of Theorem 3.5, for $i = 2, 3$, the endpoint of e_i in C_i is connected to some vertex of any even M_{C_i}-alternating cycle by an internal positive M_{C_i}-alternating path. Based on Corollary 4.8, both e_2 and e_3 are viable by alternating paths entering C_2 and C_3 through e_2 and e_3, respectively. Summerizing the above facts we can easily obtain the claim, if (*i*) holds.

Finally, making use of Corollary 4.7, we can build an M-alternating double v-racket by an obvious way in a graph with the conditions of (*ii*). Therefore the proof is complete. □

Finally, observe that C_v^M is trivially determined for constant automata, thus Definition 4.9 has a simplified form. Therefore the use of Theorem 4.10 is much easier in this special case.

5 Conclusion

We have worked out a decomposition of soliton automata into elementary automata. As the internal component automata are full and the appropriate α_0^ε-product is effectively computable, future research will concentrate on elementary automata only. Moreover, with the help of our results, the class of constant soliton automata is fully characterized. Considering practical issues, non-trivial self-transitions have an important role. We have also reduced this problem to elementary components, namely we have proved that to find self-transitions we only need to search for a

double v-racket or a pair of disjoint alternating paths in a bipartite elementary graph.

References

[1] M. Bartha, E. Gombás, On graphs with perfect internal matchings, *Acta Cybernetica* **12** (1995), 111-124.

[2] M. Bartha, M. Krész, Structuring the elementary components of graphs having a perfect internal matching, *submitted.*

[3] F. L. Carter (ed.), *Molecular Electronic Devices*, Marcel Dekker, Inc., New York, 1982.

[4] J. Dassow, H. Jürgensen, Soliton automata, *J. Comput. System Sci.* **40** (1990), 154-181.

[5] J. Dassow, H. Jürgensen, Soliton automata with a single exterior node, *Theoretical Computer Science* **84** (1991), 281-292.

[6] J. Dassow, H. Jürgensen, Soliton automata with at most one cycle, *J. Comput. System Sci.* **46** (1993), 155-197.

[7] F. Gécseg, *Products of Automata*, Akademie-Verlag, Berlin, 1986.

[8] F. Gécseg, H. Jürgensen, Automata represented by products of soliton automata, *Theoretical Computer Science* **74** (1990), 163-181.

[9] L. Lovász, M. D. Plummer, *Matching Theory*, North-Holland, Amsterdam, 1986.

Acta Cybernetica 14 (2000) 653-664.

Regulated Pushdown Automata

Alexander Meduna* Dusan Kolar *

Abstract

The present paper suggests a new investigation area of the formal language theory—*regulated automata*. Specifically, it investigates pushdown automata that regulate the use of their rules by control languages. It proves that this regulation has no effect on the power of pushdown automata if the control languages are regular. However, the pushdown automata regulated by linear control languages characterize the family of recursively enumerable languages. All these results are established in terms of (A) acceptance by final state, (B) acceptance by empty pushdown, and (C) acceptance by final state and empty pushdown. In its conclusion, this paper formulates several open problems.

Key Words: pushdown automata; regulated accepting; control languages

1 Introduction

Over the past three or four decades, grammars that regulate the use of their rules by various control mechanisms have played an important role in the language theory. Indeed, literally hundreds studies were written about these grammars (see [1], Chapter 5 in the second volume of [4], and Chapter V in [5] for an overview of these studies). Besides grammars, however, the language theory uses automata as fundamental language models, and this very elementary fact gives rise to the idea of regulated automata, which are introduced and discussed in the present paper.

More specifically, this paper introduces pushdown automata that regulate the use of their rules by control languages. First, it demonstrates that this regulation has no effect on the power of pushdown automata if the control languages are regular. Based on this result, it points out that pushdown automata regulated by analogy with the control mechanisms used in most common regulated grammars, such as matrix grammars, are of little interest because their resulting power coincides with the power of ordinary pushdown automata. Then, however, the present paper proves that the pushdown automata increase their power remarkably if they are regulated by linear languages; indeed, they characterize the family of recursively enumerable languages.

*Department of Computer Science and Engineering, Technical University of Brno, Bozetechova 2, Brno 61266, Czech Republic

All results given in this paper are established in terms of (A) acceptance by final state, (B) acceptance by empty pushdown, and (C) acceptance by final state and empty pushdown. In its conclusion, this paper discusses some open problem areas concerning regulated automata.

2 Preliminaries

We assume that the reader is familiar with the language theory (see [3]). Set $\mathcal{N} = \{1, 2, \ldots\}$ and $\mathcal{I} = \{0, 1, 2, \ldots\}$.

Let V be an alphabet. V^* represents the free monoid generated by V under the operation of concatenation. The unit of V^* is denoted by ε. Set $V^+ = V^* - \{\varepsilon\}$; algebraically, V^+ is thus the free semigroup generated by V under the operation of concatenation.

For $w \in V^*$, $|w|$ and $reversal(w)$ denote the length of w and the reversal of w, respectively. Set $prefix(w) = \{x \mid x \text{ is a prefix of } w\}$, $suffix(w) = \{x \mid x \text{ is a suffix of } w\}$, and $alph(w) = \{a \mid a \in V, \text{and } a \text{ appears in } w\}$.

For $w \in V^+$ and $i \in \{1, \ldots, |w|\}$, $sym(w, i)$ denotes the ith symbol of w; for instance, $sym(abcd, 3) = c$.

A *linear grammar* is a quadruple, $G = (N, T, P, S)$, where N and T are alphabets such that $N \cap T = \emptyset$, $S \in N$, and P is a finite set of productions of the form $A \to x$, where $A \in N$ and $x \in T^*(N \cup \{\varepsilon\})T^*$. If $A \to x \in P$ and $u, v \in T^*$, then $uAv \Rightarrow uxv$ $[A \to x]$ or, simply, $uAv \Rightarrow uxv$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of G, $L(G)$, is defined as $L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$. A language, L, is *linear* if and only if $L = L(G)$, where G is a linear grammar.

Let $G = (N, T, P, S)$ be a linear grammar. G represents a *regular grammar* if for every $A \to x \in P$, $x \in T(N \cup \{\varepsilon\})$. A language, L, is *regular* if and only if $L = L(G)$, where G is a regular grammar.

A *queue grammar* (see [2]) is a sixtuple, $Q = (V, T, W, F, S, P)$, where V and W are alphabets satisfying $V \cap W = \emptyset$, $T \subseteq V$, $F \subseteq W$, $S \in (V - T)(W - F)$, and $P \subseteq (V \times (W - F)) \times (V^* \times W)$ is a finite relation such that for every $a \in V$, there exists an element $(a, b, x, c) \in P$. If $u, v \in V^*W$ such that $u = arb$, $v = rzc$, $a \in V$, $r, z \in V^*$, $b, c \in W$ and $(a, b, z, c) \in P$, then $u \Rightarrow v$ $[(a, b, z, c)]$ in G or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$. Based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of Q, $L(Q)$, is defined as $L(Q) = \{w \in T^* \mid S \Rightarrow^* wf \text{ where } f \in F\}$.

Next, this paper slightly modifies the notion of a queue grammar.

A *left-extended queue grammar* is a sixtuple, $Q = (V, T, W, F, S, P)$, where V, T, W, F, S, P have the same meaning as in a queue grammar; in addition, assume that $\# \notin V \cup W$. If $u, v \in V^*\{\#\}V^*W$ so $u = w\#arb$, $v = wa\#rzc$, $a \in V$, $r, z, w \in V^*$, $b, c \in W$, and $(a, b, z, c) \in P$, then $u \Rightarrow v$ $[(a, b, z, c)]$ in G or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$. Based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of $Q, L(Q)$, is defined as $L(Q) = \{v \in T^* \mid \#S \Rightarrow^* w\#vf \text{ for some } w \in V^* \text{ and } f \in F\}$.

Let REG, LIN, and RE denote the families of regular, linear, and recursively enumerable languages, respectively.

3 Definitions

Consider a pushdown automaton, M, and a control language, Ξ, over M's rules. Informally, with Ξ, M accepts a word, x, if and only if Ξ contains a control word according to which M makes a sequence of moves so it reaches a final configuration after reading x.

Formally, a *pushdown automaton* is a 7-tuple, $M = (Q, \Sigma, \Omega, R, s, S, F)$, where Q is a *finite set of states*, Σ is an *input alphabet*, Ω is a *pushdown alphabet*, R is a *finite set of rules* of the form $Apa \rightarrow wq$, where $A \in \Omega$, $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, and $w \in \Omega^*$, $s \in Q$ is the *start state*, $S \in \Omega$ is the *start symbol*, $F \subseteq Q$ is a *set of final states*. In addition, this paper requires that Q, Σ, Ω are pairwise disjoint.

Let Ψ be an alphabet of *rule labels* such that $card(\Psi) = card(R)$, and ψ be a bijection from R to Ψ. For simplicity, to express that ψ maps a rule, $Apa \rightarrow wq \in R$, to ρ, where $\rho \in \Psi$, this paper writes $\rho.Apa \rightarrow wq \in R$; in other words, $\rho.Apa \rightarrow wq$ means $\psi(Apa \rightarrow wq) = \rho$. A *configuration* of M, χ, is any word from $\Omega^* Q \Sigma^*$. For every $x \in \Omega^*$, $y \in \Sigma^*$, and $\rho.Apa \rightarrow wq \in R$, M makes a move from configuration $xApay$ to configuration $xwqy$ according to ρ, written as $xApay \Rightarrow xwqy\ [\rho]$. Let χ be any configuration of M. M makes *zero moves from* χ *to* χ *according to* ε, symbolically written as $\chi \Rightarrow^0 \chi\ [\varepsilon]$. Let there exist a sequence of configurations $\chi_0, \chi_1, \ldots, \chi_n$ for some $n \geq 1$ such that $\chi_{i-1} \Rightarrow \chi_i\ [\rho_i]$, where $\rho_i \in \Psi$, for $i = 1, \ldots, n$, then M makes n *moves from* χ_0 *to* χ_n *according to* $\rho_1 \ldots \rho_n$, symbolically written as $\chi_0 \Rightarrow^n \chi_n\ [\rho_1 \ldots \rho_n]$.

Let Ξ be a *control language* over Ψ; that is, $\Xi \subseteq \Psi^*$. With Ξ, M defines the following three types of accepted languages:

$L(M, \Xi, 1)$—*the language accepted by final state*

$L(M, \Xi, 2)$—*the language accepted by empty pushdown*

$L(M, \Xi, 3)$—*the language accepted by final state and empty pushdown*

defined as follows. Let $\chi \in \Omega^* Q \Sigma^*$. If $\chi \in \Omega^* F$, $\chi \in Q$, $\chi \in F$, then χ is a *1-final configuration*, *2-final configuration*, *3-final configuration*, respectively. For $i = 1, 2, 3$, define $L(M, \Xi, i)$ as $L(M, \Xi, i) = \{w \mid w \in \Sigma^*$, and $Ssw \Rightarrow^* \chi\ [\sigma]$ in M for an i–final configuration, χ, and $\sigma \in \Xi\}$.

For any family of languages, X, set $RPD(X, i) = \{L \mid L = L(M, \Xi, i)$, where M is a pushdown automaton and $\Xi \in X\}$, where $i = 1, 2, 3$. Specifically, $RPD(REG, i)$ and $RPD(LIN, i)$ are central to this paper.

4 Results

This section demonstrates that $CF = RPD(REG, 1) = RPD(REG, 2) = RPD(REG, 3)$ and $RE = RPD(LIN, 1) = RPD(LIN, 2) = RPD(LIN, 3)$.

Some of the following proofs involve several grammars and automata. To avoid any confusion, these proofs sometimes specify a regular grammar, G, as $G = (V[G], P[G], S[G], T[G])$ because this specification clearly expresses that $V[G]$, $P[G]$, $S[G]$, and $T[G]$ represent G's components. Other grammars and automata are specified analogously whenever any confusion may exist.

Regular Control Languages

Next, this section proves that if the control languages are regular, then the regulation of pushdown automata has no effect on their power. The proof of the following lemma presents a transformation that converts any regular grammar, G, and any pushdown automaton, K, to an ordinary pushdown automaton, M, such that $L(M) = L(K, L(G), 1)$.

Lemma 1
For every regular grammar, G, and every pushdown automaton, K, there exists a pushdown automaton, M, such that $L(M) = L(K, L(G), 1)$.

Proof: Let $G = (N[G], T[G], P[G], S[G])$ be any regular grammar, and let $K = (Q[K], \Sigma[K], \Omega[K], R[K], s[K], S[K], F[K])$ be any pushdown automaton. Next, we construct a pushdown automaton, M, that simultaneously simulates G and K so that $L(M) = L(K, L(G), 1)$.

Let f be a new symbol. Define the pushdown automaton $M = (Q[M], \Sigma[M], \Omega[M], R[M], s[M], S[M], F[M])$ as $Q[M] = \{\langle qB\rangle \mid q \in Q[K], B \in N[G] \cup \{f\}\}$, $\Sigma[M] = \Sigma[K]$, $\Omega[M] = \Omega[K]$, $s[M] = \langle s[K]S[G]\rangle$, $S[M] = S[K]$, $F[M] = \{\langle qf\rangle \mid q \in F[K]\}$, and $R[M] = \{C\langle qA\rangle b \rightarrow x\langle pB\rangle \mid a.Cqb \rightarrow xp \in R[K], A \rightarrow aB \in P[G]\} \cup \{C\langle qA\rangle b \rightarrow x\langle pf\rangle \mid a.Cqb \rightarrow xp \in R[K], A \rightarrow a \in P[G]\}$.

Observe that a move in M according to $C\langle qA\rangle b \rightarrow x\langle pB\rangle \in R[M]$ simulates a move in K according $a.Cqb \rightarrow xp \in R[K]$, where a is generated in G by using $A \rightarrow aB \in P[G]$. Based on this observation, it is rather easy to see that M accepts an input word, w, if and only if K reads w and enters a final state after using a complete word of $L(G)$; therefore, $L(M) = L(K, L(G), 1)$. A rigorous proof that $L(M) = L(K, L(G), 1)$ is left to the reader. □

Theorem 2
For $i \in \{1, 2, 3\}$, $CF = RPD(REG, i)$.

Proof: To prove $CF = RPD(REG, 1)$, notice that $RPD(REG, 1) \subseteq CF$ follows from Lemma 1. Clearly, $CF \subseteq RPD(REG, 1)$, so $RPD(REG, 1) = CF$.

By analogy with the demonstration of $RPD(REG, 1) = CF$, prove that $CF = RPD(REG, 2)$ and $CF = RPD(REG, 3)$. □

Let us point out that most fundamental regulated grammars use control mechanisms that can be expressed in terms of regular control languages (c.f. Theorem V.6.1 on page 175 in [5]). However, pushdown automata introduced by analogy

with these grammars are of little or no interest because they are as powerful as ordinary pushdown automata (see Theorem 2 above).

Linear Control Languages

The rest of this section demonstrates that the pushdown automata regulated by linear control languages are more powerful than ordinary pushdown automata. In fact, it proves that $RE = RPD(LIN, 1) = RPD(LIN, 2) = RPD(LIN, 3)$.

Lemma 3
For every left-extended queue grammar, K, there exists a left-extended queue grammar $Q = (V, T, W, F, s, P)$ satisfying $L(K) = L(Q)$, ! is a distinguished member of $(W - F)$, $V = U \cup Z \cup T$ such that U, Z, T are pairwise disjoint, and Q derives every $z \in L(Q)$ in this way

$$\begin{array}{lll} \#S & \Rightarrow^+ & x\#b_1b_2\ldots b_n! \\ & \Rightarrow & xb_1\#b_2\ldots b_ny_1p_2 \\ & \Rightarrow & xb_1b_2\#b_3\ldots b_ny_1y_2p_3 \\ & \vdots & \\ & \Rightarrow & xb_1b_2\ldots b_{n-1}\#b_ny_1y_2\cdots y_{n-1}p_n \\ & \Rightarrow & xb_1b_2\ldots b_{n-1}b_n\#y_1y_2\cdots y_np_{n+1} \end{array}$$

where $n \in N$, $x \in U^*$, $b_i \in Z$ for $i = 1, \ldots, n$, $y_i \in T^*$ for $i = 1, \ldots, n$, $z = y_1y_2\cdots y_n$, $p_i \in W - \{!\}$ for $i = 1, \ldots, n-1$, $p_n \in F$, and in this derivation $x\#b_1b_2\ldots b_n!$ is the only word containing !.

Proof: Let K be any left-extended queue grammar. Convert K to a left-extended queue grammar, $H = (V[H], T[H], W[H], F[H], S[H], P[H])$, such that $L(K) = L(H)$ and H generates every $x \in L(H)$ by making two or more derivation steps (this conversion is trivial and left to the reader).

Define the bijection α from W to W', where $W' = \{q' \mid q \in W\}$, as $\alpha(q) = \{q'\}$ for every $q \in W$. Analogously, define the bijection β from W to W'', where $W'' = \{q'' \mid q \in W\}$, as $\beta(q) = \{q''\}$ for every $q \in W$. Without any loss of generality, assume that $\{1, 2\} \cap (V \cup W) = \emptyset$. Set $\Xi = \{\langle a, q, u1v, p\rangle \mid (a, q, uv, p) \in P[H]$ for some $a \in V, q \in W - F, v \in T^*, u \in V^*$, and $p \in W\}$ and $\Gamma = \{\langle a, q, z2w, p\rangle \mid (a, q, zw, p) \in P[H]$ for some $a \in V, q \in W - F, w \in T^*, z \in V^*$, and $p \in W\}$. Define the relation χ from $V[H]$ to $\Xi\Gamma$ so for every $a \in V$, $\chi(a) = \{\langle a, q, y1x, p\rangle\langle a, q, y2x, p\rangle \mid \langle a, q, y1x, p\rangle \in \Xi, \langle a, q, y2x, p\rangle \in \Gamma, q \in W - F, x \in T^*, y \in V^*, p \in W\}$. Define the bijection δ from $V[H]$ to V', where $V' = \{a' \mid a \in V\}$, as $\delta(a) = \{a'\}$. In the standard manner, extend δ so it is defined from $(V[H])^*$ to $(V')^*$. Finally, define the bijection ϕ from $V[H]$ to V'', where $V'' = \{a'' \mid a \in V\}$, as $\phi(a) = \{a''\}$. In the standard manner, extend ϕ so it is defined from $(V[H])^*$ to $(V'')^*$.

Define the left-extended queue grammar

$$Q = (V[Q], T[Q], W[Q], F[Q], S[Q], P[Q])$$

so that $V[Q] = V[H] \cup \delta(V[H]) \cup \phi(V[H]) \cup \Xi \cup \Gamma$, $T[Q] = T[H]$, $W[Q] = W[H] \cup \alpha(W[H]) \cup \beta(W[H]) \cup \{!\}$, $F[Q] = \beta(F[H])$, $S[Q] = \delta(S[H])$, and $P[V]$ is constructed in this way

1. if $(a, q, x, p) \in P[H]$ where $a \in V$, $q \in W - F$, $x \in V^*$, and $p \in W$, then add $(\delta(a), q, \delta(x), p)$ and $(\delta(a), \alpha(q), \delta(x), \alpha(p))$ to $P[Q]$;

2. if $(a, q, xAy, p) \in P[H]$, where $a \in V$, $q \in W - F$, $x, y \in V^*$, $A \in V$, and $p \in W$, then add $(\delta(a), q, \delta(x)\chi(A)\phi(y), \alpha(p))$ to $P[Q]$;

3. if $(a, q, yx, p) \in P[H]$, where $a \in V$, $q \in W - F$, $y \in V^*$, $x \in T^*$, and $p \in W$, then add $(\langle a, q, y1x, p\rangle, \alpha(q), \phi(y), !)$ and $(\langle a, q, y2x, p\rangle, !, x, \beta(p))$ to $P[Q]$;

4. if $(a, q, y, p) \in P[H]$, where $a \in V$, $q \in W - F$, $y \in T^*$, and $p \in W$, then add $(\phi(a), \beta(q), y, \beta(p))$ to $P[Q]$.

Set $U = \delta(V[H]) \cup \Xi$ and $Z = \phi(V[H]) \cup \Gamma$. Notice that Q satisfies properties 2 and 3 of Lemma 3. To demonstrate that the other two properties hold as well, observe that H generates every $z \in L(H)$ in this way

$$
\begin{aligned}
\#S[H] &\Rightarrow^+ x\#b_1b_2\dots b_ip_1 \\
&\Rightarrow xb_1\#b_2\dots b_ib_{i+1}\dots b_ny_1p_2 \\
&\Rightarrow xb_1b_2\#b_3\dots b_ib_{i+1}\dots b_ny_1y_2p_3 \\
&\vdots \\
&\Rightarrow xb_1b_2\dots b_{i-1}\#b_ib_{i+1}\dots b_ny_1y_2\dots y_{i-1}p_i \\
&\Rightarrow xb_1b_2\dots b_i\#b_{i+1}\dots b_ny_1y_2\dots y_{i-1}y_ip_{i+1} \\
&\vdots \\
&\Rightarrow xb_1b_2\dots b_{n-1}\#b_ny_1y_2\dots y_{n-1}p_n \\
&\Rightarrow xb_1b_2\dots b_{n-1}b_n\#y_1y_2\dots y_np_{n+1}
\end{aligned}
$$

where $n \in \mathcal{N}$, $x \in V^+$, $b_i \in V$ for $i = 1, \dots, n$, $y_i \in T^*$ for $i = 1, \dots, n$, $z = y_1y_2\dots y_n$, $p_i \in W$ for $i = 1, \in, n$, $p_{n+1} \in F$. Q simulates this generation of z as follows

$$
\begin{aligned}
\#S[Q] &\Rightarrow^+ \delta(x)\#\chi(b_1)\phi(b_2\dots b_i)\alpha(p_1) \\
&\Rightarrow \delta(x)\langle b_1, p_1, b_{i+1}\dots b_n1y_1, p_2\rangle\#\langle b_1, p_1, b_{i+1}\dots b_n2y_1, p2\rangle \\
&\qquad \phi(b_2\dots b_ib_{i+1}\dots b_n)! \\
&\Rightarrow \delta(x)\chi(b_1)\#\phi(b_2\dots b_n)y_1p_2 \\
&\Rightarrow \delta(x)\chi(b_1)\phi(b_2)\#\phi(b_3\dots b_n)y_1y_2p_3 \\
&\vdots \\
&\Rightarrow \delta(x)\chi(b_1)\phi(b_2\dots b_{n-1})\#\phi(b_n)y_1y_2\dots y_{n-1}p_n \\
&\Rightarrow \delta(x)\chi(b_1)\phi(b_2\dots b_n)\#y_1y_2\dots y_np_{n+1}
\end{aligned}
$$

Q makes the first $|x| - 1$ steps of $\#S[Q] \Rightarrow^+ \delta(x)\#\chi(b_1)\phi(b_2\dots b_i)\alpha(p_1)$ according to productions introduced in 1; in addition, during this derivation, Q makes one step by using a production introduced in 2. By using productions introduced in 3,

Q makes the two steps

$$\begin{array}{ll} \delta(x)\#\chi(b_1)\phi(b_2\ldots b_i)\alpha(p_0) & \Rightarrow \\ \delta(x)\langle b_1,p_1,b_{i+1}\ldots b_n1y_1,p_2\rangle\#\langle b_1,p_1,b_{i+1}\ldots b_n2y_1,p_2\rangle\phi(b_2\ldots b_ib_{i+1}\ldots b_n)! & \Rightarrow \\ \delta(x)\chi(b_1)\#\phi(b_2\ldots b_n)y_1p_2 & \end{array}$$

with

$$\chi(b_1) = \langle b_1,p_0,b_{i+1}\ldots b_n1y_1,p_1\rangle\langle b_1,p_0,b_{i+1}\ldots b_n2y_1,p2\rangle.$$

Q makes the rest of the derivation by using productions introduced in 4.

Based on the previous observation, it easy to see that Q satisfies all the four properties stated in Lemma 3, whose rigorous proof is left to the reader. □

Lemma 4
Let Q be a left-extended queue grammar that satisfies the properties of Lemma 3. Then, there exist a linear grammar, G, and a pushdown automaton, M, such that $L(Q) = L(M, L(G), 3)$.

Proof: Let $Q = (V[Q], T[Q], W[Q], F[Q], s[Q], P[Q])$ be a left-extended queue grammar satisfying the properties of Lemma 3. Without any loss of generality, assume that $\{@, £, ¶\} \cap (V \cup W) = \emptyset$. Define the coding, ζ, from $(V[Q])^*$ to $\{\langle £as\rangle \mid a \in V[Q]\}^*$ as $\zeta(a) = \{\langle £as\rangle\}$ (s is used as the start state of the pushdown automaton, M, defined later in this proof).

Construct the linear grammar $G = (N[G], T[G], P[G], S[G])$ in the following way. Initially, set

$$N[G] = \{S[G], \langle!\rangle, \langle!,1\rangle\} \cup \{\langle f\rangle \mid f \in F[Q]\}$$

$$T[G] = \zeta(V[Q]) \cup \{\langle £\S s\rangle, \langle £@\rangle\} \cup \{\langle £\S f\rangle \mid f \in F[Q]\}$$

$$P[G] = \{S[G] \to \langle £\S s\rangle\langle f\rangle \mid f \in F[Q]\} \cup \{\langle!\rangle \to \langle!, 1\rangle\langle £@\rangle\}$$

Increase $N[G]$, $T[G]$, and $P[G]$ by performing 1 through 3, following next.

1. for every $(a,p,x,q) \in P[Q]$ where $p,q \in W[Q]$, $a \in Z$, $x \in T^*$,

$$\begin{array}{ll} N[G] = N[G] & \cup \{\langle apxqk\rangle \mid k = 0,\ldots,|x|\} \cup \{\langle p\rangle, \langle q\rangle\} \\ T[G] = T[G] & \cup \{\langle £\mathrm{sym}(y,k)\rangle \mid k = 1,\ldots,|y|\} \cup \{\langle £apxq\rangle\} \\ P[G] = P[G] & \cup \{\langle q\rangle \to \langle apxq|x|\rangle\langle £apxq\rangle, \langle apxq0\rangle \to \langle p\rangle\} \\ & \cup \{\langle apxqk\rangle \to \langle apxq(k-1)\rangle\langle £\mathrm{sym}(x,k)\rangle \mid k = 1,\ldots,|x|\}; \end{array}$$

2. for every $(a,p,x,q) \in P[Q]$ with $p,q \in W[Q]$, $a \in U$, $x \in (V[Q])^*$,

$$\begin{array}{ll} N[G] = N[G] & \cup \{\langle p,1\rangle, \langle q,1\rangle\} \\ P[G] = P[G] & \cup \{\langle q,1\rangle \to \mathrm{reversal}(\zeta(x))\langle p,1\rangle\zeta(a)\}; \end{array}$$

3. for every $(a,p,x,q) \in P[Q]$ with $ap = S[Q]$, $p,q \in W[Q]$, $x \in (V[Q])^*$,

$$N[G] = N[G] \cup \{\langle q, 1\rangle\}$$
$$P[G] = P[G] \cup \{\langle q, 1\rangle \to \text{reversal}(x)\langle \pounds\$s\rangle\}.$$

The construction of G is completed. Set $\Psi = T[G]$. Ψ represents the alphabet of rule labels corresponding to the rules of the pushdown automaton $M = (Q[M], \Sigma[M], \Omega[M], R[M], s[M], S[M], \{\rceil\})$, which is constructed next.

Initially, set $Q[M] = \{s[M], \langle\P!\rangle, \lfloor, \rceil\}$ (throughout the rest of this proof, $s[M]$ is abbreviated to s), $\Sigma[M] = T[Q]$, $\Omega[M] = \{S[M], \S\} \cup V[Q]$, $R[M] = \{\langle\pounds\S s\rangle.S[M]s \to \S s\} \cup \{\langle\pounds\S f\rangle.\S\langle\P f\rangle \to \rceil \mid f \in F[M]\}$. Increase $Q[M]$ and $R[M]$ by performing A through D, following next.

A. $R[M] = R[M] \cup \{\langle\pounds bs\rangle.as \to abs \mid a \in \Omega[M] - \{S[M]\}, b \in \Omega[M] - \{\$\}\}$;

B. $R[M] = R[M] \cup \{\langle\pounds\$s\rangle.as \to a\lfloor \mid a \in V[Q]\} \cup \{\langle\pounds a\rangle.a\lfloor \to \lfloor \mid a \in V[Q]\}$;

C. $R[M] = R[M] \cup \{\langle\pounds@\rangle.a\lfloor \to a\langle\P!\rangle \mid a \in Z\}$;

D. for every $(a, p, x, q) \in P[Q]$, where $p, q \in W[Q]$, $a \in Z$, $x \in (T[Q])^*$,

$$Q[M] = Q[M] \cup \{\langle\P p\rangle\} \cup \{\langle\P qu\rangle \mid u \in \text{prefix}(x)\}$$
$$R[M] = R[M] \cup \{\langle\pounds b\rangle.a\langle\P qy\rangle b \to a\langle\P qyb\rangle \mid b \in T[Q], y \in (T[Q])^*,$$
$$yb \in \text{prefix(x)}\} \cup \{\langle\pounds apxq\rangle.a\langle\P qx\rangle \to \langle\P p\rangle\}.$$

The construction of M is completed.

Notice that several components of G and M have this form: $\langle x\rangle$. Intuitively, if x begins with $\pounds$, then $\langle x\rangle \in T[G]$. If x begins with $\P$, then $\langle x\rangle \in Q[M]$. Finally, if x begins with a symbol different from $\pounds$ or $\P$, then $\langle x\rangle \in N[G]$.

First, we only sketch the reason why $L(Q)$ contains $L(M, L(G), 3)$. Accordinng to a word from $L(G)$, M accepts every word w as

$$\begin{array}{lll}
\S w_1 \ldots w_{m-1}w_m & \Rightarrow^+ & \S b_m \ldots b_1 a_n \ldots a_1 s w_1 \ldots w_{m-1}w_m \\
 & \Rightarrow & \S b_m \ldots b_1 a_n \ldots a_1 \lfloor w_1 \ldots w_{m-1}w_m \\
 & \Rightarrow^n & \S b_m \ldots b_1 \lfloor w_1 \ldots w_{m-1}w_m \\
 & \Rightarrow & \S b_m \ldots b_1 \langle\P q_1\rangle w_1 \ldots w_{m-1}w_m \\
 & \Rightarrow^{|w_1|} & \S b_m \ldots b_1 \langle\P q_1 w_1\rangle w_2 \ldots w_{m-1}w_m \\
 & \Rightarrow & \S b_m \ldots b_2 \langle\P q_2\rangle w_2 \ldots w_{m-1}w_m \\
 & \Rightarrow^{|w_2|} & \S b_m \ldots b_2 \langle\P q_2 w_2\rangle w_3 \ldots w_{m-1}w_m \\
 & \Rightarrow & \S b_m \ldots b_3 \langle\P q_3\rangle w_3 \ldots w_{m-1}w_m \\
 & \vdots & \\
 & \Rightarrow & \S b_m \langle\P q_m\rangle w_m \\
 & \Rightarrow^{|w_m|} & \S b_m \langle\P q_m w_m\rangle \\
 & \Rightarrow & \S\langle\P q_{m+1}\rangle \\
 & \Rightarrow & \rceil
\end{array}$$

where $w = w_1 \ldots w_{m-1}w_m$, $a_1 \ldots a_n b_1 \ldots b_m = x_1 \ldots x_{n+1}$, and $R[Q]$ contains $(a_0, p_0, x_1, p_1), (a_1, p_1, x_2, p_2), \ldots, (a_n, p_n, x_{n+1}, q_1), (b_1, q_1, w_1, q_2), (b_2, q_2, w_2, q_3),$

$\ldots, (b_m, q_m, w_m, q_{m+1})$. According to these members of $R[Q]$, Q makes

$$
\begin{array}{lll}
\#a_0p_0 & \Rightarrow & a_0\#y_0x_1p_1 \qquad [(a_0, p_0, x_1, p_1)] \\
 & \Rightarrow & a_0a_1\#y_1x_2p_2 \qquad [(a_1, p_1, x_2, p_2)] \\
 & \Rightarrow & a_0a_1a_2\#y_2x_3p_3 \qquad [(a_2, p_2, x_3, p_3)] \\
 & \vdots & \\
 & \Rightarrow & a_0a_1a_2\ldots a_{n-1}\#y_{n-1}x_np_n \qquad [(a_{n-1}, p_{n-1}, x_n, p_n)] \\
 & \Rightarrow & a_0a_1a_2\ldots a_n\#y_nx_{n+1}q_1 \qquad [(a_n, p_n, x_{n+1}, q_1)] \\
 & \Rightarrow & a_0\ldots a_nb_1\#b_2\ldots b_mw_1q_2 \qquad [(b_1, q_1, w_1, q_2)] \\
 & \Rightarrow & a_0\ldots a_nb_1b_2\#b_3\ldots b_mw_1w_2q_3 \qquad [(b_2, q_2, w_2, q_3)] \\
 & \vdots & \\
 & \Rightarrow & a_0\ldots a_nb_1\ldots b_{m-1}\#b_mw_1w_2\ldots w_{m-1}q_m \qquad [(b_{m-1}, q_{m-1}, w_{m-1}, q_m)] \\
 & \Rightarrow & a_0\ldots a_nb_1\ldots b_m\#w_1w_2\ldots w_mq_{m+1} \qquad [(b_m, q_m, w_m, q_{m+1})]
\end{array}
$$

Therefore, $L(M, L(G), 3) \subseteq L(Q)$.

More formally, to demonstrate that $L(Q)$ contains $L(M, L(G), 3)$, consider any $h \in L(G)$. G generates h as

$$
\begin{array}{ll}
S[G] \Rightarrow & \langle \pounds\S s\rangle\langle q_{m+1}\rangle \\
\Rightarrow^{|w_m|+1} & \langle \pounds\S s\rangle\langle q_m\rangle t_m\langle \pounds\, b_m\, q_m\, w_m\, q_{m+1}\rangle \\
\Rightarrow^{|w_{m-1}|+1} & \langle \pounds\S s\rangle\langle q_{m-1}\rangle t_{m-1}\langle \pounds\, b_{m-1}\, q_{m-1}\, w_{m-1}\, q_m\rangle t_m\langle \pounds\, b_m\, q_m\, w_m\, q_{m+1}\rangle \\
\vdots & \\
\Rightarrow^{|w_1|+1} & \langle \pounds\S s\rangle\langle q_1\rangle o \\
\Rightarrow^{|w_1|+1} & \langle \pounds\S s\rangle\langle q_1, 1\rangle\langle \pounds@\rangle o \\
 & [\langle q_1\rangle \rightarrow \langle q_1, 1\rangle\langle \pounds@\rangle] \\
 & \\
\Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_{n+1}))\langle p_n, 1\rangle\langle \pounds\, a_n\rangle\langle \pounds@\rangle o \\
 & [\langle q_1, 1\rangle \rightarrow \text{reversal}(\zeta(x_{n+1}))\langle p_n, 1\rangle\langle \pounds\, a_n\rangle\langle \pounds@\rangle] \\
 & \\
\Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_nx_{n+1}))\langle p_{n-1}, 1\rangle\langle \pounds\, a_{n-1}\rangle\langle \pounds\, a_n\rangle\langle \pounds@\rangle o \\
 & [\langle p_n, 1\rangle \rightarrow \text{reversal}(\zeta(x_n))\langle p_{n-1}, 1\rangle\langle \pounds\, a_{n-1}\rangle] \\
 & \\
\vdots & \\
 & \\
\Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_2\ldots x_nx_{n+1}))\langle p1, 1\rangle\langle \pounds\, a_1\rangle\langle \pounds\, a_2\rangle\ldots\langle \pounds\, a_n\rangle\langle \pounds@\rangle o \\
 & [\langle p_2, 1\rangle \rightarrow \text{reversal}(\zeta(x_2))\langle p_1, 1\rangle\langle \pounds\, a_1\rangle] \\
 & \\
\Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_1\ldots x_nx_{n+1}))\langle \pounds\$s\rangle\langle \pounds\, a_1\rangle\langle \pounds\, a_2\rangle\ldots\langle \pounds\, a_n\rangle\langle \pounds@\rangle o \\
 & [\langle p_1, 1\rangle \rightarrow \text{reversal}(\zeta(x_1))\langle \pounds\$s\rangle]
\end{array}
$$

where $n, m \in \mathcal{N}$; $a_i \in U$ for $i = 1, \ldots, n$; $b_k \in Z$ for $k = 1, \ldots, m$; $x_l \in V^*$ for $l = 1, \ldots, n+1$; $p_i \in W$ for $i = 1, \ldots, n$; $q_l \in W$ for $l = 1, \ldots, m+1$ with $q_1 = !$ and $q_{m+1} \in F$; $t_k = \langle \pounds\text{sym}(w_k, 1)\rangle\ldots\langle \pounds\text{sym}(w_k, |w_k| - 1)\rangle\langle \pounds\text{sym}(w_k, |w_k|)\rangle$ for $k = 1, \ldots, m$; $o =$

$t_1\langle \pounds b_1 q_1 w_1 q_2\rangle \ldots \langle \pounds\S s\rangle\langle q_{m-1}\rangle t_{m-1}\langle \pounds b_{m-1} q_{m-1} w_{m-1} q_m\rangle t_m\langle \pounds b_m q_m w_m q_{m+1}\rangle$;
$h = \langle \pounds\S s\rangle\zeta(\text{reversal}(x_1 \ldots x_n x_{n+1}))\langle \pounds\$\rangle\langle \pounds a_1\rangle\langle \pounds a_2\rangle \ldots \langle \pounds a_n\rangle\langle \pounds @\rangle o$.

In greater detail, G makes $S[G] \Rightarrow \langle \pounds\S s\rangle\langle q_{m+1}\rangle$ according to $S[G] \to \langle \pounds\S s\rangle\langle q_{m+1}\rangle$. Furthermore, G makes

$$\begin{array}{ll} & \langle \pounds\S s\rangle\langle q_{m+1}\rangle \\ \Rightarrow^{|w_m|+1} & \langle \pounds\S s\rangle\langle q_m\rangle t_m\langle \pounds b_m q_m w_m q_{m+1}\rangle \\ \Rightarrow^{|w_{m-1}|+1} & \langle \pounds\S s\rangle\langle q_{m-1}\rangle t_{m-1}\langle \pounds b_{m-1} q_{m-1} w_{m-1} q_m\rangle t_m\langle \pounds b_m q_m w_m q_{m+1}\rangle \\ \vdots & \\ \Rightarrow^{|w_1|+1} & \langle \pounds\S s\rangle\langle q_1\rangle o \end{array}$$

according to productions introduced in step 1. Then, G makes

$$\langle \pounds\S s\rangle\langle q_1\rangle o \Rightarrow \langle \pounds\S s\rangle\langle q_1, 1\rangle\langle \pounds @\rangle o$$

according to $\langle !\rangle \to \langle !, 1\rangle\langle \pounds @\rangle$ (recall that $q_1 = !$). After this step, G makes

$$\begin{array}{ll} & \langle \pounds\S s\rangle\langle q_1, 1\rangle\langle \pounds @\rangle o \\ \Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_{n+1}))\langle p_n, 1\rangle\langle \pounds a_n\rangle\langle \pounds @\rangle o \\ \Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_n x_{n+1}))\langle p_{n-1}, 1\rangle\langle \pounds a_{n-1}\rangle\langle \pounds a_n\rangle\langle \pounds @\rangle o \\ \vdots & \\ \Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_2 \ldots x_n x_{n+1}))\langle p_1, 1\rangle\langle \pounds a_1\rangle\langle \pounds a_2\rangle \ldots \langle \pounds a_n\rangle\langle \pounds @\rangle o \end{array}$$

according to productions introduced in step 2. Finally, according to $\langle p_1, 1\rangle \to \text{reversal}(\zeta(x_1))\langle \pounds\$\rangle$, which is introduced in step 3, G makes

$$\begin{array}{ll} & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_2 \ldots x_n x_{n+1}))\langle p_1, 1\rangle\langle \pounds a_1\rangle\langle \pounds a_2\rangle \ldots \langle \pounds a_n\rangle\langle \pounds @\rangle o \\ \Rightarrow & \langle \pounds\S s\rangle\zeta(\text{reversal}(x_1 \ldots x_n x_{n+1}))\langle \pounds\$\rangle\langle \pounds a_1\rangle\langle \pounds a_2\rangle \ldots \langle \pounds a_n\rangle\langle \pounds @\rangle o \end{array}$$

If $a_1 \ldots a_n b_1 \ldots b_m$ differs from $x_1 \ldots x_{n+1}$, then M does not accept according to h. Assume that $a_1 \ldots a_n b_1 \ldots b_m = x_1 \ldots x_{n+1}$. At this point, according to h, M makes this sequence of moves

$$\begin{array}{lll} \S w_1 \ldots w_{m-1} w_m & \Rightarrow^+ & \S b_m \ldots b_1 a_n \ldots a_1 s w_1 \ldots w_{m-1} w_m \\ & \Rightarrow & \S b_m \ldots b_1 a_n \ldots a_1 \lfloor w_1 \ldots w_{m-1} w_m \\ & \Rightarrow^n & \S b_m \ldots b_1 \lfloor w_1 \ldots w_{m-1} w_m \\ & \Rightarrow & \S b_m \ldots b_1 \langle \P q_1\rangle w_1 \ldots w_{m-1} w_m \\ & \Rightarrow^{|w_1|} & \S b_m \ldots b_1 \langle \P q_1 w_1\rangle w_2 \ldots w_{m-1} w_m \\ & \Rightarrow & \S b_m \ldots b_2 \langle \P q_2\rangle w_2 \ldots w_{m-1} w_m \\ & \Rightarrow^{|w_2|} & \S b_m \ldots b_2 \langle \P q_2 w_2\rangle w_3 \ldots w_{m-1} w_m \\ & \Rightarrow & \S b_m \ldots b_3 \langle \P q_3\rangle w_3 \ldots w_{m-1} w_m \\ & \vdots & \\ & \Rightarrow & \S b_m \langle \P q_m\rangle w_m \\ & \Rightarrow^{|w_m|} & \S b_m \langle \P q_m w_m\rangle \\ & \Rightarrow & \S \langle \P q_{m+1}\rangle \\ & \Rightarrow & \rfloor \end{array}$$

In other words, according to h, M accepts $w_1 \dots w_{m-1} w_m$. Return to the generation of h in G. By the construction of $P[G]$, this generation implies that $R[Q]$ contains $(a_0, p_0, x_1, p_1), (a_1, p_1, x_2, p_2), \dots, (a_{j-1}, p_{j-1}, x_j, p_j), \dots, (a_n, p_n, x_{n+1}, q_1), (b_1, q_1, w_1, q_2), (b_2, q_2, w_2, q_3), \dots, (b_m, q_m, w_m, q_{m+1})$.

Thus, in Q,

$$
\begin{array}{lll}
\#a_0p_0 & \Rightarrow & a_0\#y_0x_1p_1 \qquad [(a_0, p_0, x_1, p_1)] \\
 & \Rightarrow & a_0a_1\#y_1x_2p_2 \qquad [(a_1, p_1, x_2, p_2)] \\
 & \Rightarrow & a_0a_1a_2\#y_2x_3p_3 \qquad [(a_2, p_2, x_3, p_3)] \\
 & \vdots & \\
 & \Rightarrow & a_0a_1a_2 \dots a_{n-1}\#y_{n-1}x_np_n \qquad [(a_{n-1}, p_{n-1}, x_n, p_n)] \\
 & \Rightarrow & a_0a_1a_2 \dots a_n\#y_nx_{n+1}q_1 \qquad [(a_n, p_n, x_{n+1}, q_1)] \\
 & \Rightarrow & a_0 \dots a_nb_1\#b_2 \dots b_mw_1q_2 \qquad [(b_1, q_1, w_1, q_2)] \\
 & \Rightarrow & a_0 \dots a_nb_1b_2\#b_3 \dots b_mw_1w_2q_3 \qquad [(b_2, q_2, w_2, q_3)] \\
 & \vdots & \\
 & \Rightarrow & a_0 \dots a_nb_1 \dots b_{m-1}\#b_mw_1w_2 \dots w_{m-1}q_m \qquad [(b_{m-1}, q_{m-1}, w_{m-1}, q_m)] \\
 & \Rightarrow & a_0 \dots a_nb_1 \dots b_m\#w_1w_2 \dots w_mq_{m+1} \qquad [(b_m, q_m, w_m, q_{m+1})]
\end{array}
$$

Therefore, $w_1w_2 \dots w_m \in L(Q)$. Consequently, $L(M, L(G), 3) \subseteq L(Q)$.

A proof that that $L(Q) \subseteq L(M, L(G), 3)$ is left to the reader. As $L(Q) \subseteq L(M, L(G), 3)$ and $L(M, L(G), 3) \subseteq L(Q)$, $L(Q) = L(M, L(G), 3)$. Therefore, Lemma 4 holds. □

Theorem 5
For $i \in \{1, 2, 3\}$, $RE = RPD(LIN, i)$.

Proof: Obviously, $RPD(LIN, 3) \subseteq RE$. To prove $RE \subseteq RPD(LIN, 3)$, consider any recursively enumerable language, $L \in RE$. By Theorem 2.1 in [2], $L(Q) = L$, for a queue grammar. Clearly, there exists a left-extended queue grammar, Q', so $L(Q) = L(Q')$. Furthermore, by Lemmas 3 and 4, $L(Q') = L(M, L(G), 3)$, for a linear grammar, G, and a pushdown automaton, M. Thus, $L = L(M, L(G), 3)$. Hence, $RE \subseteq RPD(LIN, 3)$. As $RPD(LIN, 3) \subseteq RE$ and $RE \subseteq RPD(LIN, 3)$, $RE = RPD(LIN, 3)$.

By analogy with the demonstration of $RE = RPD(LIN, 3)$, prove $RE = RPD(LIN, i)$ for $i = 1, 2$. □

5 Future Investigation

As already pointed out, this paper has discussed regulated automata as a new investigation field of the formal language theory. Therefore, it has defined all notions and established all results in terms of this new field. However, this approach does not rule out a relation of the achieved results to the classical formal language theory. Specifically, Theorem 5 can be viewed as a new characterization of RE and

compared with other well-known characterizations of this family (see pages 180 through 184 in the first volume of [4] for an overview of these characterizations).

Several research topics remain to be explored:

A. For $i = 1, \ldots, 3$, consider $RPD(X, i)$, where X is a language family satisfying $REG \subset X \subset LIN$; for instance, set X equal to the family of minimal linear languages. Compare RE with $RPD(X, i)$.

B. Investigate special cases of regulated pushdown automata, such as their deterministic versions.

C. By analogy with regulated pushdown automata, introduce and study some other types of regulated automata.

D. Investigate the descriptional complexity of regulated pushdown automata.

References

[1] Dassow, J. and Paun, G.: *Regulated Rewriting in Formal Language Theory*. Springer, New York, 1989.

[2] Kleijn, H. C. M. and Rozenberg, G.: On the Generative Power of Regular Pattern Grammars, *Acta Informatica*, Vol. 20, pp. 391–411, 1983.

[3] Meduna, A.: *Automata and Languages: Theory and Applications*. Springer, London, 2000.

[4] Rozenberg, G. and Salomaa, A. (eds.): *Handbook of Formal Languages; Volumes 1 through 3*. Springer, Berlin/Heidelberg, 1997.

[5] Salomaa, A.: *Formal Languages*. Academic Press, New York, 1973.

Received January, 2000

CONTENTS

ISSN 0324—721 X

Felelős szerkesztő és kiadó: Csirik János
A kézirat a nyomdába érkezett: 2000. november