TARGET BASED ACCEPTING NETWORKS OF EVOLUTIONARY PROCESSORS WITH REGULAR FILTERS

Bianca Truthe
Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik
Universitätsplatz 2, D-39016 Magdeburg, Germany
Email: `bianca.truthe@ovgu.de`

Abstract
In this paper, a new definition of accepting networks – called target based accepting networks – is given. In a target based accepting network of evolutionary processors, each node is equipped with a regular language – the target set. As soon as a node contains a word of its target set, the input word is accepted by the network. In this way, no further output nodes are necessary. It is shown that conventional accepting networks with regular filters and target based accepting networks with regular filters have the same computational power. However, the number of processors needed for accepting a language can be reduced when using target based networks.

1. Introduction

Motivated by some models of massively parallel computer architectures, networks of language processors have been introduced in [6] by E. Csuhaj-Varjú and A. Salomaa. Such a network can be considered as a graph where the nodes are sets of productions and at any moment of time a language is associated with a node.

Inspired by biological processes, J. Castellanos, C. Martín-Vide, V. Mitrana and J. Sempere introduced in [4] a special type of networks of language processors which are called networks with evolutionary processors because the allowed productions model the point mutation known from biology. The sets of productions have to be substitutions of one letter by another letter or insertions of letters or deletion of letters; the nodes are then called substitution, insertion, or deletion node, respectively. Results on networks of evolutionary processors can be found e. g. in [4], [5], [3], [1].

The generative capacity of networks with regular filters and evolutionary processors where only two types of nodes are allowed were investigated in [8] and [2].

Accepting networks of evolutionary processors with regular filters were introduced by J. Dassow and V. Mitrana in [7]. Further results on such networks were published in [9].

In the present paper, we give a new definition of accepting networks (called target based accepting networks). It is shown that, for every conventional accepting network, there exists a target based network accepting the same language and the converse that, for every target based accepting network, there exists a conventional network accepting the same language. So both kinds of accepting networks have the same computational power. However, the number of processors needed for accepting a language can be reduced when using target based networks.

We show that every context-sensitive language can be accepted by a target based accepting network of evolutionary processors with regular filters and with exactly one processor (a substitution processor). Any recursively enumerable language can be accepted by a target based network with exactly one insertion processor and exactly one substitution or deletion processor.

2. Definitions

We assume that the reader is familiar with the basic concepts of formal language theory (see e. g. [10]). We here only recall some notations used in the paper.

By V^* we denote the set of all words (strings) over an alphabet V (including the empty word λ). The length of a word w is denoted by $|w|$. When we add new letters of an alphabet U to a given alphabet V, then we assume that $V \cap U = \emptyset$.

A phrase structure grammar is specified as a quadruple $G = (N, T, P, S)$ where N is a set of non-terminals, T is a set of terminals, P is a finite set of productions which are written as $\alpha \to \beta$ with $\alpha \in (N \cup T)^* \setminus T^*$ and $\beta \in (N \cup T)^*$, and $S \in N$ is the axiom.

We call a production $\alpha \to \beta$ a substitution if $|\alpha| = |\beta| = 1$ and a deletion if $|\alpha| = 1$ and $\beta = \lambda$. Insertion is the counterpart of deletion. We write $\lambda \to a$, where a is a letter. The application of an insertion $\lambda \to a$ derives from a word w any word $w_1 a w_2$ with $w = w_1 w_2$ for some (possibly empty) words w_1 and w_2.

We now introduce the basic concept of this paper, the target based accepting networks of evolutionary processors.

Definition 2.1.

1. *A target based accepting network of evolutionary processors of size n is a tuple*

$$\mathcal{T}^{(n)} = (U, V, N_1, N_2, \dots, N_n, E, j)$$

where

- *U and V are finite alphabets (the input and network alphabet, resp.), $U \subseteq V$,*
- *for $1 \le i \le n$, $N_i = (M_i, I_i, O_i, B_i)$ where*
  - *M_i is a set of evolution rules of a certain type, $M_i \subseteq \{a \to b \mid a, b \in V\}$ or $M_i \subseteq \{a \to \lambda \mid a \in V\}$ or $M_i \subseteq \{\lambda \to b \mid b \in V\}$,*

  - *I_i, O_i, and B_i are regular sets over V,*
- *E is a subset of $\{1, 2, \ldots, n\} \times \{1, 2, \ldots, n\}$, and*
- *j is a natural number such that $1 \leq j \leq n$.*

2. *A configuration C of $\mathcal{T}^{(n)}$ is an n-tuple $C = (C(1), C(2), \ldots, C(n))$ where $C(i)$ is a subset of V^* for $1 \leq i \leq n$.*

3. *Let $C = (C(1), C(2), \ldots, C(n))$ and $C' = (C'(1), C'(2), \ldots, C'(n))$ be two configurations of $\mathcal{T}^{(n)}$. We say that C derives C' in one*

   – *evolutionary step (written as $C \Longrightarrow C'$) if, for $1 \leq i \leq n$, $C'(i)$ consists of all words $w \in C(i)$ to which no rule of M_i is applicable and of all words w for which there are a word $v \in C(i)$ and a rule $p \in M_i$ such that $v \Longrightarrow_p w$ holds,*

   – *communication step (written as $C \vdash C'$) if, for $1 \leq i \leq n$,*

$$C'(i) = (C(i) \setminus O_i) \cup \bigcup_{(k,i) \in E} C(k) \cap O_k \cap I_i.$$

   The computation of a network $\mathcal{T}^{(n)}$ on an input word $w \in V^*$ is a sequence of configurations $C_t^w = (C_t^w(1), C_t^w(2), \ldots, C_t^w(n))$, $t \geq 0$, such that

   – *$C_0^w = (C_0^w(1), C_0^w(2), \ldots, C_0^w(n))$ where $C_0^w(j) = \{w\}$ and $C_0^w(i) = \emptyset$ for $1 \leq i \leq n$ and $i \neq j$,*

   – *for any $t \geq 0$, C_{2t}^w derives C_{2t+1}^w in one evolutionary step: $C_{2t}^w \Longrightarrow C_{2t+1}^w$,*

   – *for any $t \geq 0$, C_{2t+1}^w derives C_{2t+2}^w in one communication step: $C_{2t+1}^w \vdash C_{2t+2}^w$.*

4. *Let O be the set of all nodes with a non-empty target set. The languages $L_{\mathrm{w}}(\mathcal{T}^{(n)})$ weakly accepted by $\mathcal{T}^{(n)}$ and $L_{\mathrm{s}}(\mathcal{T}^{(n)})$ strongly accepted by $\mathcal{T}^{(n)}$ are defined as*

$$L_{\mathrm{w}}(\mathcal{T}^{(n)}) = \{\, w \in U^* \mid \exists t \geq 0\, \exists o \in O : C_t^w(o) \cap B_o \neq \emptyset \,\},$$
$$L_{\mathrm{s}}(\mathcal{T}^{(n)}) = \{\, w \in U^* \mid \exists t \geq 0\, \forall o \in O : C_t^w(o) \cap B_o \neq \emptyset \,\},$$

   where $C_t^w = (C_t^w(1), C_t^w(2), \ldots, C_t^w(n))$, $t \geq 0$ is the computation of $\mathcal{T}^{(n)}$ on w.

Intuitively, a network with evolutionary processors is a graph consisting of some, say n, nodes $N_1, N_2, \ldots, N_n$ (called processors) and the set of edges given by E such that there is a directed edge from N_k to N_i if and only if $(k, i) \in E$. The node N_j is called the input node; every node N_o with $B_o \neq \emptyset$ is called an output node. Any processor N_i consists of a set of evolution rules M_i, an input filter I_i, an output filter O_i, and a target (base) language B_i. We say that N_i is a substitution node or a deletion node or an insertion node if $M_i \subseteq \{a \to b \mid a, b \in V\}$ or $M_i \subseteq \{a \to \lambda \mid a \in V\}$ or $M_i \subseteq \{\lambda \to b \mid b \in V\}$, respectively. The input filter I_i and the output filter O_i control the words which are allowed to enter and to leave the node, respectively. With any node N_i and any time moment $t \geq 0$ we associate a set $C_t(i)$ of words (the words contained in the node at time t). Initially, the input node N_j contains an input word w; all other nodes do not contain words. In an evolutionary step, we derive from $C_t(i)$ all words by applying rules from the set M_i. In a communication step, any processor N_i sends out all words $C_t(i) \cap O_i$ (which pass the output filter) to all processors to which a directed edge exists (only the words from $C_t(i) \setminus O_i$ remain in the set associated with N_i) and, moreover, it receives from any processor N_k such that there is an edge from N_k to N_i all words sent by N_k and passing the input filter I_i of N_i, i. e., the processor N_i gets in addition all words of $(C_t(k) \cap O_k) \cap I_i$. We start with an evolutionary step and then communication steps and evolutionary steps are alternately performed. The language accepted consists of all words w such that if w is given as an input word in the node N_j then, at some moment t, $t \geq 0$, one output node (for weak acceptance) contains a word of its target set or all output nodes (for strong acceptance) contain a word of their target sets.

For comparing target based networks with conventional ones, we give their definition, too.

Definition 2.2.

1. *An accepting network of evolutionary processors of size n is a tuple*

$$\mathcal{N}^{(n)} = (U, V, N_1, N_2, \ldots, N_n, E, j, O)$$

where

- *U, V, E, and j are defined as in Definition 2.1,*
- *for $1 \leq i \leq n$, $N_i = (M_i, I_i, O_i)$ where M_i, I_i, and O_i are defined as in Definition 2.1,*
- *$O \subseteq \{ 1, \ldots, n \}$.*

2. *The configuration and computation of a network are defined as in Definition 2.1.*

3. *The languages $L_{\mathrm{w}}(\mathcal{N}^{(n)})$ weakly accepted by $\mathcal{N}^{(n)}$ and $L_{\mathrm{s}}(\mathcal{N}^{(n)})$ strongly accepted by $\mathcal{N}^{(n)}$ are defined as*

$$L_{\mathrm{w}}(\mathcal{N}^{(n)}) = \{ w \in U^* \mid \exists t \geq 0 \, \exists o \in O : C_t^w(o) \neq \emptyset \} ,$$
$$L_{\mathrm{s}}(\mathcal{N}^{(n)}) = \{ w \in U^* \mid \exists t \geq 0 \, \forall o \in O : C_t^w(o) \neq \emptyset \} ,$$

where $C_t^w = (C_t^w(1), C_t^w(2), \ldots, C_t^w(n))$, $t \geq 0$ is the computation of $\mathcal{N}^{(n)}$ on w.

The main difference of this definition to the previous one is that each processor N_i in a target based network is equipped additionally with a target (base) language B_i for $1 \leq i \leq n$ whereas a conventional network has designated output nodes. A target based network accepts an input word if it can be transformed into a word of the base language of one or all output nodes. A conventional network accepts an input word if at some moment one or all output nodes contain a word.

3. Computational Power

We now show that target based and conventional accepting networks have the same computational power. However, the number of processors needed for accepting a language can be reduced when using target based networks.

Theorem 3.1. *Every network given according to Definition 2.2 can be transformed into a target based network that accepts the same language.*

Proof. Let $n \geq 1$ be a natural number and $\mathcal{N}^{(n)} = (U, V, N_1, N_2, \ldots, N_n, E, j, O)$ be a conventional accepting network of evolutionary processors with $N_i = (M_i, I_i, O_i)$ for $1 \leq i \leq n$. If one sets $B_i = \emptyset$ for the nodes N_i that are not output nodes ($1 \leq i \leq n$ and $i \notin O$) as well as $M_o = \emptyset$ and $B_o = V^*$ for the output nodes ($o \in O$), then the target based network $\mathcal{T}^{(n)} = (U, V, N'_1, N'_2, \ldots, N'_n, E, j)$ with $N'_i = (M_i, I_i, O_i, B_i)$ accepts the same language as $\mathcal{N}^{(n)}$, because $C_t^w(o) \cap V^* \neq \emptyset$ holds if and only if $C_t^w(o) \neq \emptyset$. Hence, $L_{\mathrm{w}}(\mathcal{N}^{(n)}) = L_{\mathrm{w}}(\mathcal{T}^{(n)})$ and $L_{\mathrm{s}}(\mathcal{N}^{(n)}) = L_{\mathrm{s}}(\mathcal{T}^{(n)})$. $\square$

Due to the construction given in the previous proof, a target based network needs at most as many processors for accepting a language as a conventional network.

If a conventional network accepts a word then it is accepted in an even numbered moment (immediately in the beginning of the process or after a communication step). A target based network can accept a word also in an odd numbered moment (if in some node N_i, a word $w \notin B_i$ is derived during an evolutionary step to a word $w' \in B_i$). To treat this difference, we call a target based network $\mathcal{T}$ *acceptance uniform* if all nodes $N_o = (M_o, I_o, O_o, B_o)$ with $B_o \neq \emptyset$ satisfy $M_o = \emptyset$. Since a word in an acceptance indicating node cannot be modified, an input word can always be accepted in an even numbered moment.

The target based network constructed in the proof of Theorem 3.1 is also acceptance uniform. We first show that, for every target based network $\mathcal{T}$, there is an acceptance uniform target based network $\mathcal{A}$ that accepts the same language and then we prove that, for every acceptance uniform target based network $\mathcal{A}$, there is a conventional network $\mathcal{N}$ that accepts the same language.

Lemma 3.2. *Every target based network can be transformed into an acceptance uniform target based network that accepts the same language.*

Proof. Let $n \geq 1$ be a natural number and $\mathcal{T}^{(n)} = (U, V, N_1, N_2, \ldots, N_n, E, j)$ be a target based accepting network of evolutionary processors $N_i = (M_i, I_i, O_i, B_i)$. Let $O = \{\, o \mid B_o \neq \emptyset \,\}$ be the set of all indices of output nodes.

We construct a target based network $\mathcal{A}^{(7n)} = (U, V, N'_1, N'_2, \ldots, N'_{7n}, E', j)$ as follows. For every node N_i of $\mathcal{T}^{(n)}$ ($1 \leq i \leq n$), we construct seven nodes that are connected according to the following graph:

$$\begin{aligned}
N'_i &= (\emptyset, V^*, V^*, \emptyset),\\
N'_{i+n} &= (M_i, V^*, V^*, \emptyset),\\
N'_{i+2n} &= (\emptyset, V^* \setminus O_i, V^*, \emptyset),\\
N'_{i+3n} &= (\emptyset, V^*, V^*, \emptyset),\\
N'_{i+4n} &= (\emptyset, O_i, V^*, \emptyset),\\
N'_{i+5n} &= (\emptyset, I_i, V^*, \emptyset),\\
N'_{i+6n} &= (\emptyset, B_i, V^*, B_i).
\end{aligned}$$

$$\begin{tikzcd}
i+3n \arrow[d] & i+2n \arrow[l] & & & \\
i \arrow[r] \arrow[dr] & i+n \arrow[u] \arrow[r] \arrow[d] & i+4n \arrow[dr, "\text{if } (i,k) \in E"] & & \\
 & i+6n & & k+5n \arrow[r] & k
\end{tikzcd}$$

The network $\mathcal{A}^{(7n)}$ is acceptance uniform.

In $\mathcal{T}^{(n)}$, a loss-less evolution-communication cycle has the form

$$w \in C_{2t}(i) \implies w' \in C_{2t+1}(i) \vdash w' \in C_{2(t+1)}(k)$$

if $w \Longrightarrow_{M_i} w'$ as well as $w' \in O_i \cap I_k$ and $(i,k) \in E$ or $w' \notin O_i$ and $k = i$.

This corresponds in $\mathcal{A}^{(7n)}$ to the following cycle (the configuration is denoted by K here):

$$\begin{array}{rcll}
w \in K_{8t}(i) & \implies & w \in K_{8t+1}(i) & \text{(no evolution in } N'_i\text{)}\\
 & \vdash & w \in K_{8t+2}(i+n) & \\
 & & \text{and } w \in K_{8t+2}(i+6n) \text{ if } w \in B_i & \\
w \in K_{8t+2}(i+n) & \implies & w' \in K_{8t+3}(i+n) \text{ with } w \Longrightarrow_{M_i} w' & \\
 & \vdash & w' \in K_{8t+4}(i+2n) \text{ if } w' \notin O_i & \\
 & & w' \in K_{8t+4}(i+4n) \text{ if } w' \in O_i & \\
 & & \text{and } w' \in K_{8t+4}(i+6n) \text{ if } w' \in B_i & \\
w' \in K_{8t+4}(i+2n) & \implies & w' \in K_{8t+5}(i+2n) & \text{(no evolution in } N'_{i+2n}\text{)}\\
 & \vdash & w' \in K_{8t+6}(i+3n) & \\
 & \implies & w' \in K_{8t+7}(i+3n) & \text{(no evolution in } N'_{i+3n}\text{)}\\
 & \vdash & w' \in K_{8(t+1)}(i) & \\
w' \in K_{8t+4}(i+4n) & \implies & w' \in K_{8t+5}(i+4n) & \text{(no evolution in } N'_{i+4n}\text{)}\\
 & \vdash & w' \in K_{8t+6}(k+5n) \text{ if } (i,k) \in E \text{ and } w' \in I_k & \\
 & \implies & w' \in K_{8t+7}(k+5n) & \text{(no evolution in } N'_{k+5n}\text{)}\\
 & \vdash & w' \in K_{8(t+1)}(k). &
\end{array}$$

Now let $z \in C_t^w(o) \cap B_o$ for $o \in O$ be a word that indicates acceptance of the input word in $\mathcal{T}^{(n)}$. There are two cases.

Case 1. The time t is even. Let $t' = \frac{t}{2}$. According to the derivation cycle above, we have $z \in C_{2t'}(o)$ therefore $z \in K_{8t'}(o)$, $z \in K_{8t'+1}(o)$ and $z \in K_{8t'+2}(o+6n)$ because $z \in B_o$. Since $t' = \frac{t}{2}$, we have $z \in K_{4t+2}(o+6n)$.

Case 2. The time t is odd. In the previous step, an evolutionary rule of M_o was applied. Hence, there exists $z' \in C_{t-1}(o)$ with $z' \Longrightarrow_{M_o} z$. Let $t' = \frac{t-1}{2}$. In $\mathcal{A}^{(7n)}$, we have $z' \in K_{8t'}(o)$, following the cycle to $z' \in K_{8t'+2}(o+n)$, $z \in K_{8t'+3}(o+n)$, and $z \in K_{8t'+4}(o+6n)$. Because $t' = \frac{t-1}{2}$, we have $z \in K_{4t}(o+6n)$.

In both cases, we see: if for an input word w there is a time t such that $C_t^w(o) \cap B_o$ is not empty for one/all output nodes then there is also a time t' such that $K_{t'}(o+6n) \cap B_o$ is not empty.

To show the convers, let $z \in K_t(o+6n) \cap B_o$. If t is odd, then z was in the same node already a step before ($z \in K_{t-1}(o+6n) \cap B_o$) because there are no evolutionary rules in N'_{o+6n}. So, we can assume t to be even. In a moment $8t'$ or $8t'+6$, the node N'_{o+6n} does not contain a word (it can receive a word only by communication before time $8t'+2$ and $8t'+4$ and looses everything in the next communication step). Hence, we have the following cases (we write $t \equiv r \pmod 8$ if and only if $t = 8m + r$ for an integer m).

Case 1. $t \equiv 2 \pmod 8$: The word z was sent by N'_o. Hence, $z \in K_{t-1}(o)$ and also $z \in K_{t-2}(o)$ (no evolution in N'_o). Since $t-2 \equiv 0 \pmod 8$ let $t' = \frac{t-2}{8}$. We have $z \in K_{8t'}(o)$ and therefore $z \in C_{2t'}(o)$ and $z \in C_{\frac{t-2}{4}}(o)$.

Case 2. $t \equiv 4 \pmod 8$: The word z was sent by N'_{o+n}. Hence, $z \in K_{t-1}(o+n)$. There is a word z' with $z' \in K_{t-2}(o+n)$ and $z' \Longrightarrow_{M_o} z$. If we follow the cycle backwards, we obtain $z' \in K_{t-3}(o)$, $z' \in K_{t-4}(o)$ (no evolution in N'_o). Since $t-4 \equiv 0 \pmod 8$ let $t' = \frac{t-4}{8}$. We have $z' \in K_{8t'}(o)$ and therefore $z' \in C_{2t'}(o)$. Because $z' \Longrightarrow_{M_o} z$, it holds $z \in C_{2t'+1}(o)$ and $z \in C_{\frac{t}{4}}(o)$.

In both cases, we have: if for an input word w there is a time t such that the set $K_t^w(o+6n) \cap B_o$ is not empty for one/all output nodes then there is also a time t' such that $C_{t'}(o) \cap B_o$ is not empty.

This yields $L_{\mathrm{w}}(\mathcal{T}^{(n)}) = L_{\mathrm{w}}(\mathcal{A}^{(7n)})$ and $L_{\mathrm{s}}(\mathcal{T}^{(n)}) = L_{\mathrm{s}}(\mathcal{A}^{(7n)})$. □

Lemma 3.3. *Every acceptance uniform target based network can be transformed into a conventional network that accepts the same language.*

Proof. Let $n \geq 0$ and $m \geq 1$ be natural numbers and

$$\mathcal{T}^{(n+m)} = (U, V, N_1, N_2, \ldots, N_{n+m}, E, j)$$

be an acceptance uniform target based network of evolutionary processors with nodes $N_i = (M_i, I_i, O_i, B_i)$ for $1 \leq i \leq n+m$ where

$$\begin{aligned} B_i &= \emptyset \quad \text{for } 1 \leq i \leq n \text{ and} \\ B_i &\neq \emptyset \quad \text{for } n+1 \leq i \leq n+m. \end{aligned}$$

Let $O = \{ o \mid n+1 \leq o \leq n+m \}$ be the set of all indices of output nodes. We construct a conventional network

$$\mathcal{N}^{(n+m+k)} = (U, V, N'_1, N'_2, \ldots, N'_{n+m+k}, E', e, O')$$

with

$$k = \begin{cases} m, & \text{if } 1 \leq j \leq n \text{ (the input node is not an output node)}, \\ 2m+1, & \text{if } n+1 \leq j \leq n+m \text{ (the input node is an output node)} \end{cases}$$

as follows. For $i = 1, \ldots, n+m$, we set $N'_i = (M_i, I_i, O_i)$. Whenever $(p,q) \in E$ is an edge in $\mathcal{T}^{(n+m)}$, we connect the nodes N'_p and N'_q.

Case 1. The input node is not an output node ($1 \leq j \leq n$). Then it will be the new input node, too: $e = j$. For every output node N_o, we create a new processor $N'_{o+m} = (\emptyset, I_o \cap B_o, O_o)$. These nodes are the output nodes of the conventional network:

$$O' = \{ o+m \mid n+1 \leq o \leq n+m \} = \{ o' \mid n+1+m \leq o' \leq n+2m \}.$$

Whenever $(p,o) \in E$ is an edge in $\mathcal{T}^{(n+m)}$ with $1 \leq p \leq n+m$ and $o \in O$, we also connect the nodes N'_p and N'_{o+m}:

$$E_1 = \{ (p, o+m) \mid (p,o) \in E,\ 1 \leq p \leq n+m,\ n+1 \leq o \leq n+m \}.$$

Furthermore, we set $E_2 = \emptyset$ and $E_3 = \emptyset$.

Case 2. The input node is also an output node ($n+1 \leq j \leq n+m$). Then we set $j' = j+2m$ and we create a new input node $N'_e = (\emptyset, \emptyset, U^*)$ with $e = n+3m+1$. For every output node N_o, we create two new processors $N'_{o+m} = (\emptyset, I_o \cap B_o, V^*)$ and $N'_{o+2m} = (\emptyset, B_o, O_o)$. The output nodes of the conventional network are the nodes N'_{o+2m}:

$$O' = \{ o+2m \mid n+1 \leq o \leq n+m \} = \{ o' \mid n+1+2m \leq o' \leq n+3m \}.$$

Whenever $(p,o) \in E$ is an edge in $\mathcal{T}^{(n+m)}$ with $1 \leq p \leq n+m$ and $o \in O$, we also connect the nodes N'_p and N'_{o+m}:

$$E_1 = \{ (p, o+m) \mid (p,o) \in E,\ 1 \leq p \leq n+m,\ n+1 \leq o \leq n+m \}.$$

These intermediate nodes are connected to the output nodes:

$$E_2 = \{ (o+m, o+2m) \mid n+1 \leq o \leq n+m \}.$$

Additionally, we connect the new input node N'_e to $N'_{j'}$ and to any node N'_i with $1 \leq i \leq n$ and $(j,i) \in E$ and to any node N'_o and N'_{o+m} with $n+1 \leq o \leq n+m$ and $(j,o) \in E$:

$$E_3 = \{ (e,i) \mid (j,i) \in E,\ 1 \leq i \leq n+m \} \cup \{ (e, o+m) \mid (j,o) \in E,\ o \in O \} \cup \{ (e, j') \}.$$

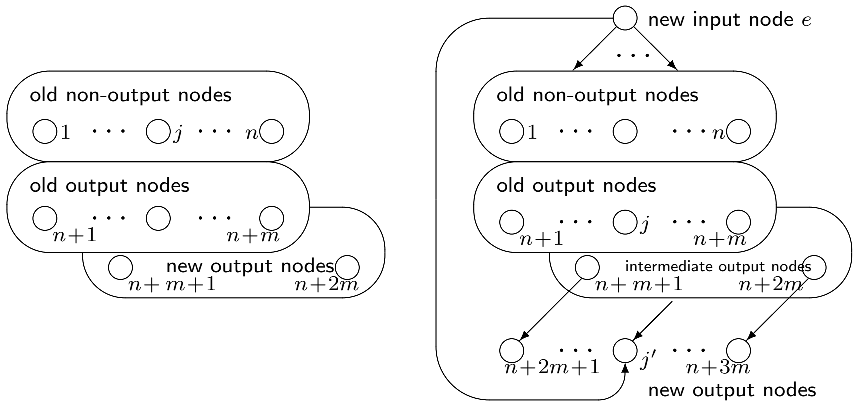

Figure 1: Network constructed if the input node is not an output node (left) or it is (right)

The edge set of the new network is $E' = E \cup E_1 \cup E_2 \cup E_3$. Figure 1 illustrates the construction described above.

We now prove that the network $\mathcal{N}^{(n+m+k)}$ accepts the same language as $\mathcal{T}^{(n+m)}$. We first consider the case $1 \leq j \leq n$ that the input node is not among the output nodes. Then no word can be accepted immediately (in time $t = 0$).

If, for an input word $w \in U^*$, an output node N_o and a time $t > 0$, the condition $z \in C_t^w(o) \cap B_o$ holds for a word z then t is even and we have one of the following possibilities:

Case 1. The node N_o has received the word z from some node N_l in the last step. Hence, $z \in C_{t-1}^w(l) \cap O_l \cap I_o$ and $(l, o) \in E$. Then also $z \in C_{t-1}^w(l) \cap O_l \cap I_o \cap B_o$ and $z \in C_t^w(o+m)$.

Case 2. The word z was already in the node N_o. Then $z \in C_{t-1}^w(o)$ and $z \notin O_o$. Since $C_0^w(o) = \emptyset$, the word z arrived in some time from a node N_l in N_o. Let t' be an even natural number with $t' < t$. Then $z \in C_{t'-1}^w(l) \cap O_l \cap I_o$ and $(l, o) \in E$. Then also $z \in C_{t'-1}^w(l) \cap O_l \cap I_o \cap B_o$ and $z \in C_{t'}^w(o+m)$. Because $z \notin O_o$, we have $z \in C_t^w(o+m)$.

According to this case distinction, we have that if $C_t^w(o) \cap B_o \neq \emptyset$ for one/all output nodes then $C_t^w(o+m) \neq \emptyset$ holds, too. Hence, if a word $w \in U^*$ is accepted by $\mathcal{T}^{(n+m)}$ then it is also accepted by $\mathcal{N}^{(n+m+k)}$.

If, for an input word $w \in U^*$, a node $o' \in O'$ and a time t, the condition $C_t^w(o') \neq \emptyset$ holds then $o' = o + m$ for an index o with $n + 1 \leq o \leq n + m$, t is an even number and $t > 0$ (in the beginning, we have $C_0^w(o') = \emptyset$ and the non-emptiness cannot arise by evolution) and we have one of the following two situations (some word was sent in some time from a node N_l'):

Case 1. Let $z \in C_{t-1}^w(l) \cap O_l \cap I_o \cap B_o$ (a word z was sent by N_l'). In this case, we also have $z \in C_t^w(o) \cap B_o$.

Case 2. Let $z \in C_{t-1}^w(o+m)$ (z was already contained in N_{o+m}'). Then $z \notin O_o$. Since $C_0^w(o+m) = \emptyset$, the word z arrived in some time from a node N_l' in N_{o+m}'. Let t' be an even natural number with $t' < t$. Then $z \in C^w_{t'-1}(l) \cap O_l \cap I_o \cap B_o$ and $(l,o) \in E$. Then also $z \in C^w_{t'}(o) \cap B_o$. Because $z \notin O_o$, we have $z \in C^w_t(o) \cap B_o$.

Together, we have the result: if the condition $C^w_t(o') \neq \emptyset$ holds for one/all output nodes then $C^w_t(o) \cap B_o \neq \emptyset$ holds, too. Hence, if a word $w \in U^*$ is accepted by $\mathcal{N}^{(n+m+k)}$ then it is also accepted by $\mathcal{T}^{(n+m)}$.

We now consider the case $n+1 \leq j \leq n+m$ that the input node is also an output node. Then the input word could be accepted immediately (in time $t=0$) by $\mathcal{T}^{(n+m)}$.

If an input word w belongs to the set B_j then it is accepted by $\mathcal{T}^{(n+m)}$. In the network $\mathcal{N}^{(n+m+k)}$, the node N'_e does not modify the word, so w is sent out. It passes the input filter B_j of the new output node $N'_{j'}$ and hence, it is also accepted by the conventional network $\mathcal{N}^{(n+m+k)}$. On the other hand, w is only accepted immediately (after one communication step) by $\mathcal{N}^{(n+m+k)}$ if it belongs to B_j.

If the input word w does not belong to B_j then the word is sent unchanged from N_j to any node N_k in $\mathcal{T}^{(n+m)}$ with $(j,k) \in E$ or from N'_e to any node N'_k in $\mathcal{N}^{(n+m+k)}$ with $(j,k) \in E$. It does not pass the input filter of the output node $N'_{j'}$. From here on, we have the same situation as described above for the case $1 \leq j \leq n$.

If a word arrives in some node N'_{o+m} then it will be sent during the next communication step to the output node N'_{o+2m}. Hence, we have $C^w_t(o) \cap B_o \neq \emptyset$ for one/all output nodes if and only if $C^w_{t+2}(o') \neq \emptyset$ holds, too.

Hence, a word $w \in U^*$ is accepted by the target based network $\mathcal{T}^{(n+m)}$ if and only if it is accepted by the conventional network $\mathcal{N}^{(n+m+k)}$.

All cases together yield

$$\begin{aligned} L_{\mathrm{w}}(\mathcal{N}^{(n+m+k)}) &= \{\, w \in U^* \mid \exists t \geq 0\, \exists o' \in O' : C^w_t(o') \neq \emptyset \,\} \\ &= \{\, w \in U^* \mid \exists t \geq 0\, \exists o \in O : C^w_t(o) \cap B_o \neq \emptyset \,\} \\ &= L_{\mathrm{w}}(\mathcal{T}^{(n+m)}), \end{aligned}$$

and also

$$\begin{aligned} L_{\mathrm{s}}(\mathcal{N}^{(n+m+k)}) &= \{\, w \in U^* \mid \exists t \geq 0\, \forall o' \in O' : C^w_t(o') \neq \emptyset \,\} \\ &= \{\, w \in U^* \mid \exists t \geq 0\, \forall o \in O : C^w_t(o) \cap B_o \neq \emptyset \,\} \\ &= L_{\mathrm{s}}(\mathcal{T}^{(n+m)}), \end{aligned}$$

because for each $o \in O$ the node $o' \in O'$ is uniquely determined. □

The next theorem summarizes the previous results.

Theorem 3.4. *Every target based network can be transformed into a conventional network that accepts the same language.*

Every language accepted by a conventional network is also accepted by a target based network and vice versa. The number of processors a target based network needs for accepting a language is not higher than the number of processors that a conventional network needs for the same language.

In the following sections, we consider accepting networks with two types of nodes as presented in [9] and show that in all cases we can omit the output nodes.

4. Networks with Processors of Two Types only

Deletion and substitution nodes do not increase the length of the words. Such nodes are also called non-increasing. In a network with only non-increasing nodes, the length of every word in every node at any step in the computation is bounded by the length of the input word. In [7], it was shown that every network with substitution and deletion nodes accepts a context-sensitive language. In [9], it was shown that every context-sensitive language can be accepted by a network with one substitution node and one output node without rules.

Theorem 4.1. *For any context-sensitive language L, there is a target based network $\mathcal{S}$ with exactly one substitution node that weakly and strongly accepts the language L.*

Proof. Let L be a context-sensitive language and $G = (N, T, P, S)$ be a grammar in Kuroda normal form with $L(G) = L$.

Let Let $R_1, R_2, \ldots, R_8$ be the following sets:

$$\begin{aligned}
R_1 &= \{\, x \to x_{p,0},\ x_{p,0} \to A \mid A \to x \in P,\ A \in N,\ x \in N \cup T \,\},\\
R_2 &= \{\, C \to C_{p,1} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ A, B, C, D \in N \,\},\\
R_3 &= \{\, D \to D_{p,2} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ A, B, C, D \in N \,\},\\
R_4 &= \{\, C_{p,1} \to C_{p,3} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ A, B, C, D \in N \,\},\\
R_5 &= \{\, D_{p,2} \to D_{p,4} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ A, B, C, D \in N \,\},\\
R_6 &= \{\, C_{p,3} \to A \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ A, B, C, D \in N \,\},\\
R_7 &= \{\, D_{p,4} \to B \mid p = AB \to CD \in P,\ A, B, C, D \in N \,\},\\
R_8 &= \{\, D_{p,4} \to \sqcup \mid p = A \to CD \in P,\ A, B, C, D \in N \,\}.
\end{aligned}$$

We construct a target based network of evolutionary processors

$$\mathcal{S} = (T, V, (M, \emptyset, O, B), \{\,(1,1)\,\}, 1)$$

with

$$\begin{aligned}
V &= N \cup T \cup \{\textvisiblespace\} \cup \bigcup_{p=A\to x} \{\, x_{p,0} \,\} \cup \bigcup_{\substack{p=A\to CD\\ p=AB\to CD}} \{\, C_{p,1}, D_{p,2}, C_{p,3}, D_{p,4} \,\},\\
M &= R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5 \cup R_6 \cup R_7 \cup R_8,\\
B &= \begin{cases} \{\textvisiblespace\}^*\{S\}\{\textvisiblespace\}^* \cup \{\lambda\} & \text{if } \lambda \in L,\\ \{\textvisiblespace\}^*\{S\}\{\textvisiblespace\}^* & \text{otherwise,} \end{cases}
\end{aligned}$$

and

$$O = V^* \setminus ((N \cup T \cup \{\textvisiblespace\})^*\bar{O}(N \cup T \cup \{\textvisiblespace\})^*),$$

where

$$\begin{aligned}
\bar{O} = &\{\, x_{p,0} \mid p = A \to x \in P,\ A \in N,\ x \in N \cup T \,\}\\
&\cup \{\, C_{p,1} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ A, B, C, D \in N \,\}\\
&\cup \{\, C_{p,1}\varepsilon D_{p,2} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ \varepsilon \in \{\textvisiblespace\}^* \,\}\\
&\cup \{\, C_{p,3}\varepsilon D_{p,2} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ \varepsilon \in \{\textvisiblespace\}^* \,\}\\
&\cup \{\, C_{p,3}\varepsilon D_{p,4} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ \varepsilon \in \{\textvisiblespace\}^* \,\}\\
&\cup \{\, A\varepsilon D_{p,4} \mid p = A \to CD \in P \text{ or } p = AB \to CD \in P,\ \varepsilon \in \{\textvisiblespace\}^* \,\}\\
&\cup \{\, \lambda \,\}.
\end{aligned}$$

The network $\mathcal{S}$ has only one node. Therefore, there is no difference between weak and strong acceptance, and we write $L(\mathcal{S})$ for the language accepted by the network $\mathcal{S}$.

The empty word λ is accepted by $\mathcal{S}$ if and only if it is generated by G. By a case distinction on the types of rules, one can show, that a non-empty word is accepted if and only if it can be reduced to the axiom S by a reverse simulation of the generation process. For the details, we refer to the section on non-deleting generating networks in [2].

The trick of this network is that the substitution processor can simulate the deletion by marking symbols as deleted (the symbols are replaced by the special symbol $\textvisiblespace$). When simulating a derivation $w \Longrightarrow v$ in G, we have to take into account that the corresponding word in the substitution node may contain gaps in form of several occurrences of the special symbol $\textvisiblespace$.

An input word $w \in T^+$ can be reduced to a word $s \in \{\textvisiblespace\}^*\{S\}\{\textvisiblespace\}^* \subseteq B$ if and only if w is generated by the grammar G. If the input word is λ, then it cannot be modified in the first node. It is accepted if and only if it belongs to $L(G)$.

Hence, we have proved $L(G) = L_{\mathrm{w}}(\mathcal{S}) = L_{\mathrm{s}}(\mathcal{S}) = L$. □

This number of processors is optimal. The main difference between context-sensitive and non-context-sensitive grammars is that, in arbitrary phrase structure grammars, erasing rules (λ-

rules) are allowed. In order to simulate a λ-rule in reverse direction, we introduce an insertion node.

Theorem 4.2. *For any recursively enumerable language L, there is a target based accepting network $\mathcal{N}$ of evolutionary processors with exactly one substitution node and one insertion node that weakly and strongly accepts the language L.*

Proof. Let L be a recursively enumerable language and $G = (N, T, P, S)$ be a grammar in Kuroda normal form with $L(G) = L$.

The idea of the proof is to extend the network $\mathcal{S}$ constructed in the proof of Theorem 4.1 by an inserting processor who is responsible for the reverse simulation of λ-rules.

For a formal definition, we have to implement that the insertion node gets the opportunity to do something (the substitution processor must indicate that a word can pass to the insertion node).

We construct a network of evolutionary processors

$$\mathcal{N} = (T, V^{\mathcal{N}}, (M_1^{\mathcal{N}}, I_1^{\mathcal{N}}, O_1^{\mathcal{N}}, B^{\mathcal{N}}), (M_2^{\mathcal{N}}, I_2^{\mathcal{N}}, O_2^{\mathcal{N}}, \emptyset), \{\, (1,2), (2,1) \,\}, 1)$$

with

$$\begin{aligned}
V^{\mathcal{N}} &= V \cup \{\, x' \mid x \in N \cup T \,\},\\
M_1^{\mathcal{N}} &= M \cup \{\, x \to x' \mid x \in N \cup T \,\} \cup \{\, x' \to x \mid x \in N \cup T \,\},\\
I_1^{\mathcal{N}} &= I_2^{\mathcal{N}} = O_2^{\mathcal{N}} = (N \cup T \cup \{␣\})^* \,\{\, x' \mid x \in N \cup T \,\}\, (N \cup T \cup \{␣\})^*,\\
O_1^{\mathcal{N}} &= O \cup I_1^{\mathcal{N}},\\
B^{\mathcal{N}} &= B,\\
M_2^{\mathcal{N}} &= \{\, \lambda \to A \mid A \to \lambda \in P \,\},
\end{aligned}$$

where V, M, O, and B are defined as in the proof of Theorem 4.1.

Between two simulation phases, the substitution node can mark a symbol such that the word can leave the node and enter the insertion node. This processor inserts a non-terminal that belongs to a λ-rule of the grammar G and returns the word to the substitution node. This processor then has to unmark the primed symbol. If marking or unmarking is not performed in the correct moment, the word will be lost. The network $\mathcal{N}$ accepts the language that is generated by the grammar G. □

In [2], we have shown that every recursively enumerable language can be generated by a network of one inserting processor and one deleting processor. Similar to the proof of this statement, we proved in [11] that any recursively enumerable language can be accepted by a network with exactly one deletion node, one insertion node, and one output node.

Theorem 4.3. *For any recursively enumerable language L, there is a target based accepting network $\mathcal{N}$ of evolutionary processors with exactly one deletion node and one insertion node that weakly and strongly accepts the language L.*

Proof. Let L be a recursively enumerable language and $G = (N, T, P, S)$ be a grammar in Kuroda normal form with $L(G) = L$.

We define the sets of partial prefixes and partial suffixes of a word u by

$$PPref(u) = \{x \mid u = xy,\ |y| \geq 1\}, \quad \text{and} \quad PSuf(u) = \{y \mid u = xy,\ |x| \geq 1\},$$

respectively.

Let $V = N \cup T$ and $V_{\llcorner\!\lrcorner} = V \cup \{\llcorner\!\lrcorner\}$. We define a homomorphism $h : V^* \to V_{\llcorner\!\lrcorner}^*$ by $h(a) = a$ for $a \in T$ and $h(A) = A_{\llcorner\!\lrcorner}$ for $A \in N$ and set $W = \{\, h(w) \mid w \in V^* \,\}$. We construct the following network $\mathcal{N} = (T, X, (M_1, I_1, O_1, \emptyset), (M_2, I_2, O_2, \{S_{\llcorner\!\lrcorner}\}), E, 1)$ of evolutionary processors with

$$\begin{aligned}
X &= V_{\llcorner\!\lrcorner} \cup \bigcup_{p \in P} \{p_1, p_2, p_3, p_4\},\\
M_1 &= \{\, \lambda \to \llcorner\!\lrcorner \,\} \cup \{\, \lambda \to p_i \mid p \in P, 1 \leq i \leq 4 \,\} \cup \{\, \lambda \to A \mid A \in N \,\},\\
I_1 &= W \setminus \{\, S_{\llcorner\!\lrcorner} \,\},\\
O_1 &= X^* \setminus (W R_{1,1} W),\\
M_2 &= \{\, p_i \to \lambda \mid p \in P, 1 \leq i \leq 4 \,\} \cup \{\, x \to \lambda \mid x \in V_{\llcorner\!\lrcorner} \,\},\\
I_2 &= W R_{1,2} W,\\
O_2 &= X^* \setminus (W R_{2,2} W),\\
E &= \{(1,2), (2,1)\}
\end{aligned}$$

where

$$\begin{aligned}
R_{1,1} &= \bigcup_{p = u \to v \in P} (\{p_1 h(v), p_1 h(v) p_2, p_1 p_3 h(v) p_2, p_1 p_3 h(v) p_2 p_4\}\\
&\qquad\qquad \cup \{p_1 p_3\} PSuf(h(u)) \{h(v) p_2 p_4\})\\
R_{1,2} &= \{\, p_1 p_3 h(uv) p_2 p_4 \mid p = u \to v \in P \,\},\\
R_{2,2} &= \bigcup_{p = u \to v \in P} (\{p_1 p_3 h(u)\} PPref(h(v)) \{p_2 p_4\} \cup \{p_3 h(u) p_2 p_4, p_3 h(u) p_4, h(u) p_4\}).
\end{aligned}$$

The reverse simulation of the application of a rule $p = a_1 \ldots a_s \to b_1 \ldots b_t$ to a sentential form $\alpha a_1 \ldots a_s \beta$ with $x = h(\alpha)$ and $y = h(\beta)$ has the following form.

In the insertion node, we have

$$\begin{aligned}
&xh(b_1)\dots h(b_t)y \Longrightarrow xp_1h(b_1)\dots h(b_t)y \Longrightarrow xp_1h(b_1)\dots h(b_t)p_2y \Longrightarrow xp_1p_3h(b_1)\dots h(b_t)p_2y\\
&\Longrightarrow xp_1p_3h(b_1)\dots h(b_t)p_2p_4y \Longrightarrow xp_1p_3{\sqcup}h(b_1)\dots h(b_t)p_2p_4y \Longrightarrow xp_1p_3a_s{\sqcup}h(b_1)\dots h(b_t)p_2p_4y\\
&\Longrightarrow^* xp_1p_3a_2{\sqcup}\dots a_s{\sqcup}(b_1)\dots h(b_t)p_2p_4y \Longrightarrow xp_1p_3{\sqcup}a_2{\sqcup}\dots a_s{\sqcup}h(b_1)\dots h(b_t)p_2p_4y\\
&\Longrightarrow xp_1p_3a_1{\sqcup}\dots a_s{\sqcup}h(b_1)\dots h(b_t)p_2p_4y.
\end{aligned}$$

This word leaves the insertion node and enters the deletion node. There, the evolution continues to

$$\begin{aligned}
&xp_1p_3a_1{\sqcup}\dots a_s{\sqcup}h(b_1)\dots h(b_t)p_2p_4y \Longrightarrow^{|h(b_t)|} xp_1p_3a_1{\sqcup}\dots a_s{\sqcup}h(b_1)\dots h(b_{t-1})p_2p_4y\\
&\Longrightarrow^* xp_1p_3a_1{\sqcup}\dots a_s{\sqcup}h(b_1)p_2p_4y \Longrightarrow^{|h(b_1)|} xp_1p_3a_1{\sqcup}\dots a_s{\sqcup}p_2p_4y\\
&\Longrightarrow xp_3a_1{\sqcup}\dots a_s{\sqcup}p_2p_4y \Longrightarrow xp_3a_1{\sqcup}\dots a_s{\sqcup}p_4y \Longrightarrow xa_1{\sqcup}\dots a_s{\sqcup}p_4y\\
&\Longrightarrow xa_1{\sqcup}\dots a_s{\sqcup}y.
\end{aligned}$$

If this word is $S_{\sqcup}$, the input word is accepted. Otherwise, the word enters the insertion node again for the next simulation phase. Only those words remain in the network that are obtained in the sequence described above; all other words get lost. □

In all three cases, the number of nodes can be reduced if we use target based networks for accepting a language instead of conventional networks.

References

[1] A. Alhazov, C. Martín-Vide and Yu. Rogozhin, On the number of nodes in universal networks of evolutionary processors. *Acta Inf.* **43** (2006), 331–339.

[2] A. Alhazov, J. Dassow, C. Martín-Vide, Yu. Rogozhin and B. Truthe, On Networks of Evolutionary Processors with Nodes of Two Types. *Fundamenta Informaticae* **91** (2009) 1, 1–15.

[3] J. Castellanos, P. Leupold and V. Mitrana, On the size complexity of hybrid networks of evolutionary processors. *Theor. Comput. Sci.* **330** (2005), 205–220.

[4] J. Castellanos, C. Martín-Vide, V. Mitrana and J. Sempere, Solving NP-complete problems with networks of evolutionary processors. In: *Proc. IWANN*, Lecture Notes in Computer Science 2084, Springer-Verlag, Berlin, 2001, 621–628.

[5] J. Castellanos, C. Martín-Vide, V. Mitrana and J. Sempere, Networks of evolutionary processors. *Acta Informatica* **38** (2003), 517–529.

[6] E. Csuhaj-Varjú and A. Salomaa, Networks of parallel language processors. In: Gh. Păun and A. Salomaa (eds.), *New Trends in Formal Language Theory.* Lecture Notes in Computer Science 1218, Springer-Verlag, Berlin, 1997, 299–318.

[7] J. Dassow and V. Mitrana, Accepting Networks of Non-Increasing Evolutionary Processors. In: I. Petre and G. Rozenberg (eds.), *Proceedings of NCGT 2008. Workshop on Natural Computing and Graph Transformations. September 8, 2008, Leicester, UK.* University of Leicester, 2008, 29–41.

[8] J. Dassow and B. Truthe, On the Power of Networks of Evolutionary Processors. In: J. Durand-Lose and M. Margenstern (eds.), *Machines, Computations and Universality, MCU 2007, Orléans, France, September 10–13, 2007, Proceedings.* Lecture Notes in Computer Science 4664, Springer, 2007, 158–169.

[9] V. Mitrana and B. Truthe, On Accepting Networks of Evolutionary Processors with at Most Two Types of Nodes. In: A. H. Dediu, A. M. Ionescu, and C. Martín-Vide (eds.), *Language and Automata Theory and Applications, Third International Conference, LATA 2009, Tarragona, Spain, April 2009, Proceedings.* Lecture Notes in Computer Science 5457, Springer, 2009, 588–600.

[10] G. Rozenberg and A. Salomaa, *Handbook of Formal Languages.* Springer-Verlag, Berlin, 1997.

[11] B. Truthe, On small accepting networks of evolutionary processors with regular filters. In: J. Dassow and B. Truthe (eds.), *Colloquium on the Occasion of the 50th Birthday of Victor Mitrana. Proceedings.* Otto-von-Guericke-Universität Magdeburg, Germany, 2008, 37–52.