Networks of Evolutionary Picture Processors with Filtered Connections

Paolo Bottoni[1], Anna Labella[1], Florin Manea[2,*], Victor Mitrana[2,3,⋆], and Jose M. Sempere[3,⋆⋆]

[1] Department of Computer Science, "Sapienza" University of Rome
Via Salaria 113, 00198 Rome, Italy
{bottoni,labella}@di.uniroma1.it
[2] Faculty of Mathematics, University of Bucharest
Str. Academiei 14, 70109 Bucharest, Romania
{flmanea,mitrana}@fmi.unibuc.ro
[3] Department of Information Systems and Computation
Technical University of Valencia,
Camino de Vera s/n. 46022 Valencia, Spain
jsempere@dsic.upv.es

Abstract. In this paper we simplify the model of computation considered in [1], namely network of evolutionary picture processors, by moving the filters from the nodes to the edges. Each edge is now viewed as a two-way channel such that input and output filters, respectively, of the two nodes connected by the edge coincide. Thus, the possibility of controlling the computation in such networks seems to be diminished. In spite of this observation all the results concerning the computational power of networks of evolutionary picture processors reported in [1] are extended over these simplified networks.

1 Introduction

The origin of accepting networks of evolutionary processors (ANEP for short) is a basic architecture for parallel and distributed symbolic processing, related to the Connection Machine [8] as well as the Logic Flow paradigm [5], which consists of several very simple processors (called evolutionary processors), each of them being placed in a node of a virtual complete graph. By an evolutionary processor we mean an abstract processor which is able to perform very simple operations, namely point mutations in a DNA sequence (insertion, deletion or substitution of a pair of nucleotides). More generally, each node may be viewed as a cell having genetic information encoded in DNA sequences which may evolve by local evolutionary events, that is point mutations. Each node is specialized just for one of these evolutionary operations. Furthermore, the data in each node

* Work supported by the PN-II Programs 11052 (GlobalComp) and 11056 (CellSim). Victor Mitrana acknowledges support from Academy of Finland, project 132727.
** Work supported by the Spanish Ministerio de Educación y Ciencia under project TIN2007-60769.

C.S. Calude et al. (Eds.): UC 2009, LNCS 5715, pp. 70–84, 2009.

is organized in the form of multisets of strings (each string appears in an arbitrarily large number of copies), and all copies are processed in parallel such that all the possible events that can take place do actually take place once on each copy. Further, all the nodes send simultaneously their data and the receiving nodes handle also simultaneously all the arriving messages, according to some strategies. The reader interested in a more detailed discussion about the model is referred to [10,11] and the references thereof.

A picture (2-dimensional word) is a rectangular array of symbols over an alphabet. Picture languages defined by different mechanisms have been studied extensively in the literature. For a survey on picture languages the reader may consult [6] while an early survey on automata recognizing rectangular pictures languages is [9].

The investigation on ANEPs started in [10] has been carried over rectangular picture in [1] where accepting networks of evolutionary picture processors (ANEPP for short) have been considered. Each node of an ANEPP is either a row/column substitution node or a row/column deletion node. The action of each node on the data it contains is precisely defined. For instance, if a node is a row substitution node, then it can substitute a letter by another letter in either the topmost or the last or an arbitrary row. Moreover, if there are more occurrences of the letter that is to be substituted in the row on which the substitution rule acts, then each such occurrence is substituted in different copies of that picture. An implicit assumption is that arbitrarily many copies of every picture are available. A similar informal explanation concerns the column substitution and deletion nodes, respectively. Two ways of accepting pictures are considered in [1]: *weak acceptance*, when at least one output node is nonempty, and *strong acceptance*, when all output nodes are nonempty. Every language weakly accepted by a network can be strongly accepted by another network. One shows that ANEPPs can weakly accept the complement of any local language, as well as languages that are not recognizable. The problem of pattern matching in pictures is then considered in the framework of ANEPPs.

It is clear that filters associated with each node allow a strong control of the computation. Indeed, every node has an input and output filter; two nodes can exchange data if it passes the output filter of the sender *and* the input filter of the receiver. Moreover, if some data is sent out by some node and not able to enter any node, then it is lost. In this paper we simplify the ANEPP model considered in [1] by moving the filters from the nodes to the edges. A similar investigation has been done for ANEPs in [3,4], where it was shown that both devices equal the computational power of Turing machines. Each edge of a network of evolutionary picture processors with filtered connections (ANEPPFC for short) is viewed as a two-way channel such that the input and output filters, respectively, of the two nodes connected by the edge coincide. Clearly, the possibility of controlling the computation in such networks seems to be diminished. For instance, there is no possibility to lose data during the communication steps. In spite of this fact we can extend all the results reported in [1] to these new devices. Moreover, in all cases the ANEPPFCs have a smaller size (number of processors). This suggests

that moving the filters from the nodes to the edges does not decrease the computational power of the model. It is worth mentioning that a direct proof showing that ANEPs and ANEPs with filtered connections are computationally equivalent was proposed in [2]. However, that construction essentially need an operation that has not a corresponding one in ANEPPs or ANEPPFCs, therefore we do not know a proof for a direct simulation of one model by the other.

2 Basic Definitions

For basic terminology and notations concerning the theory of one-dimensional languages the reader is referred to [13]. The definitions and notations concerning two-dimensional languages are taken from [6].

The set of natural numbers from 1 to n is denoted by $[n]$. The cardinality of a finite set A is denoted by $card(A)$. Let V be an alphabet, V^* the set of one-dimensional strings over V and ε the empty string. A *picture* (or two-dimensional string) over the alphabet V is a two-dimensional array of elements from V. We denote the set of all pictures over the alphabet V by V_*^*, while the empty picture will be still denoted by ε. A two-dimensional language over V is a subset of V_*^*. The minimal alphabet containing all symbols appearing in a picture π is denoted by $alph(\pi)$. Let π be a picture in V_*^*; we denote the number of rows and the number of columns of π by $\underline{\overline{\pi}}$ and $|\pi|$, respectively. The pair $(\underline{\overline{\pi}}, |\pi|)$ is called *the size* of the picture π. The size of the empty picture ε is obviously $(0, 0)$. The set of all pictures over V of size (m, n), where $m, n \geq 1$, is denoted by V_m^n. The symbol placed at the intersection of the ith row with the jth column of the picture π, is denoted by $\pi(i, j)$. The row picture of size $(1, n)$ containing occurrences of the symbol a only is denoted by a_1^n. Similarly the column picture of size $(m, 1)$ containing occurrences of the symbol a only is denoted by a_m^1.

We recall informally the row and column concatenation operations between pictures. For a formal definition the reader is referred to [9] or [6]. The row concatenation of two pictures π of size (m, n) and ρ of size (m', n') is denoted by ® and is defined only if $n = n'$. The picture $\pi ® \rho$ is obtained by adding the picture ρ below the last row of π. Analogously one defines the column concatenation denoted by ©. We now recall from [1] the definition of four new operations, in some sense the inverse operations of the row and column concatenation. Let π and ρ be two pictures of size (m, n) and (m', n'), respectively. We define

- The column right-quotient of π with ρ: $\pi /_{\rightarrow} \rho = \theta$ iff $\pi = \theta © \rho$.
- The column left-quotient of π with ρ: $\pi /_{\leftarrow} \rho = \theta$ iff $\pi = \rho © \theta$.
- The row down-quotient of π with ρ to the right: $\pi /_{\downarrow} \rho = \theta$ iff $\pi = \theta ® \rho$.
- The column up-quotient of π with ρ: $\pi /_{\uparrow} \rho = \theta$ iff $\pi = \rho ® \theta$.

We now proceed with the definition of an evolutionary picture processor following [1]. We want to stress that the evolutionary processor described here is just a mathematical concept similar to that of an evolutionary algorithm, both being inspired from the Darwinian evolution. Let V be an alphabet; a rule of the form

$a \to b(X)$, with $a, b \in V \cup \{\varepsilon\}$ and $X \in \{-, |\}$ is called an *evolutionary rule*. For any rule $a \to b(X)$, X indicates which component of a picture (row if $X = -$ or column if $X = |$) the rule is applied to. We say that a rule $a \to b(X)$ is a *substitution rule* if both a and b are not ε, is a *deletion rule* if $a \neq \varepsilon$, $b = \varepsilon$, and is an *insertion rule* if $a = \varepsilon$, $b \neq \varepsilon$. In this paper we shall ignore insertion rules because we want to process every given picture in a space bounded by the size of that picture. We denote by $RSub_V = \{a \to b(-) \mid a, b \in V\}$ and $RDel_V = \{a \to \varepsilon(-) \mid a \in V\}$. The sets $CSub_V$ and $CDel_V$ are defined analogously. Given a rule σ as above and a picture $\pi \in V_m^n$, we define the following *actions* of σ on π:

• If $\sigma \equiv a \to b(|) \in CSub_V$, then

$$\sigma^{\leftarrow}(\pi) = \begin{cases} \{\pi' \in V_m^n : \exists i \in [m]\ (\pi(i,1) = a\ \&\ \pi'(i,1) = b), \pi'(k,1) = \pi(k,1), \\ k \in [m] \setminus \{i\}, \pi'(j,l) = \pi(j,l), (j,l) \in [m] \times ([n] \setminus \{1\})\} \\ \\ \{\pi\}, \text{ if the first column of } \pi \text{ does not contain any occurrence} \\ \text{of the letter } a. \end{cases}$$

$$\sigma^{\rightarrow}(\pi) = \begin{cases} \{\pi' \in V_m^n : \exists i \in [m]\ (\pi(i,n) = a\ \&\ \pi'(i,n) = b), \pi'(k,n) = \pi(k,n), \\ k \in [m] \setminus \{i\}, \pi'(j,l) = \pi(j,l), (j,l) \in [m] \times [n-1]\} \\ \\ \{\pi\}, \text{ if the last column of } \pi \text{ does not contain any occurrence} \\ \text{of the letter } a. \end{cases}$$

$$\sigma^{*}(\pi) = \begin{cases} \{\pi' \in V_m^n : \exists (i,j) \in [n] \times [m] \text{ such that } \pi(i,j) = a \text{ and} \\ \pi'(i,j) = b, \pi'(k,l) = \pi(k,l), \forall (k,l) \in ([n] \times [m]) \setminus \{(i,j)\}\} \\ \\ \{\pi\}, \text{ if no column of } \pi \text{ contains any occurrence of the letter } a. \end{cases}$$

Note that a rule as above is applied to all occurrences of the letter a either in the first or in the last or in any column of π, respectively, in different copies of the picture π. Analogously, we define:

• If $\sigma \equiv a \to b(-) \in RSub_V$, then

$$\sigma^{\uparrow}(\pi) = \begin{cases} \{\pi' \in V_m^n : \exists i \in [n](\pi(1,i) = a\ \&\ \pi'(1,i) = b), \pi'(1,k) = \pi(1,k), \\ \forall k \in [n] \setminus \{i\}, \pi'(j,l) = \pi(j,l), \forall (j,l) \in ([m] \setminus \{1\}) \times [n]\} \\ \\ \{\pi\}, \text{ if the first row of } \pi \text{ does not contain any occurrence} \\ \text{of the letter } a. \end{cases}$$

$$\sigma^{\downarrow}(\pi) = \begin{cases} \{\pi' \in V_m^n : \exists i \in [n](\pi(m,i) = a\ \&\ \pi'(m,i) = b), \pi'(m,k) = \pi(m,k), \\ \forall k \in [n] \setminus \{i\}, \pi'(j,l) = \pi(j,l), \forall (j,l) \in [m-1] \times [n]\} \\ \\ \{\pi\}, \text{ if the last row of } \pi \text{ does not contain any occurrence} \\ \text{of the letter } a. \end{cases}$$

$\sigma^{*}(\pi) = \rho^{*}(\pi)$, where $\rho \equiv a \to b(|) \in CSub_V$.

• If $\sigma \equiv a \to \varepsilon(|) \in CDel_V$, then

$$\sigma^{\leftarrow}(\pi) = \begin{cases} \pi/_{\leftarrow}\rho, \text{ where } \rho \text{ is the leftmost column of } \pi, \text{ if the leftmost column of } \pi \text{ does contain at least one occurrence of the letter } a \\ \pi, \text{ if the leftmost column of } \pi \text{ does not contain any occurrence of the letter } a. \end{cases}$$

$$\sigma^{\rightarrow}(\pi) = \begin{cases} \pi/_{\rightarrow}\rho, \text{ where } \rho \text{ is the rightmost column of } \pi, \text{ if the rightmost column of } \pi \text{ does contain at least one occurrence of the letter } a \\ \pi, \text{ if the rightmost column of } \pi \text{ does not contain any occurrence of the letter } a. \end{cases}$$

$$\sigma^{*}(\pi) = \begin{cases} \{\pi_1 ⓒ \pi_2 \mid \pi = \pi_1 ⓒ \rho ⓒ \pi_2, \text{ for some } \pi_1, \pi_2 \in V_{**} \text{ and } \rho \text{ is a column of } \pi_1 \text{ that contains an occurrence of the letter } a\} \\ \{\pi\}, \text{ if } \pi \text{ does not contain any occurrence of the letter } a. \end{cases}$$

In an analogous way we define:

• If $\sigma \equiv a \rightarrow \varepsilon(-) \in RDel_V$, then

$$\sigma^{\uparrow}(\pi) = \begin{cases} \pi/_{\uparrow}\rho, \text{ where } \rho \text{ is the first row of } \pi, \text{ if the first row of } \pi \text{ does contain at least one occurrence of the letter } a \\ \pi, \text{ if the first row of } \pi \text{ does not contain any occurrence of the letter } a. \end{cases}$$

$$\sigma^{\downarrow}(\pi) = \begin{cases} \pi/_{\downarrow}\rho, \text{ where } \rho \text{ is the last row of } \pi, \text{ if the last row of } \pi \text{ does contain at least one occurrence of the letter } a \\ \pi, \text{ if the last row of } \pi \text{ does not contain any occurrence of the letter } a. \end{cases}$$

$$\sigma^{*}(\pi) = \begin{cases} \{\pi_1 ® \pi_2 \mid \pi = \pi_1 ® \rho ® \pi_2, \text{ for some } \pi_1, \pi_2 \in V_{**} \text{ and } \rho \text{ is a row of } \pi_1 \text{ that contains an occurrence of the letter } a\} \\ \{\pi\}, \text{ if } \pi \text{ does not contain any occurrence of the letter } a. \end{cases}$$

For every rule σ, action $\alpha \in \{*, \leftarrow, \rightarrow, \uparrow, \downarrow\}$, and $L \subseteq V_{**}$, we define the *α-action of σ on L* by $\sigma^{\alpha}(L) = \bigcup_{\pi \in L} \sigma^{\alpha}(\pi)$. Given a finite set of rules M, we define the *α-action of M* on the picture π and the language L by $M^{\alpha}(\pi) = \bigcup_{\sigma \in M} \sigma^{\alpha}(\pi)$ and $M^{\alpha}(L) = \bigcup_{\pi \in L} M^{\alpha}(\pi)$, respectively. In what follows, we shall refer to the rewriting operations defined above as *evolutionary picture operations* since they may be viewed as the 2-dimensional linguistic formulations of local gene mutations. For two disjoint subsets $P \neq \emptyset$ and F of an alphabet V and a picture π over V, we define the following two predicates which will define later two types of filters:

$$\begin{aligned} rc_s(\pi; P, F) &\equiv P \subseteq alph(\pi) \wedge F \cap alph(\pi) = \emptyset \\ rc_w(\pi; P, F) &\equiv alph(\pi) \cap P \neq \emptyset \wedge F \cap alph(\pi) = \emptyset. \end{aligned}$$

The construction of these predicates is based on *context conditions* defined by the two sets P (*permitting contexts/symbols*) and F (*forbidding contexts/symbols*). Informally, both conditions require that no forbidding symbol is present in π; furthermore the first condition requires all permitting symbols to appear in π, while the second one requires that at least one permitting symbol appear in π. It is plain to see that the first condition is stronger than the second one.

For every picture language $L \subseteq V_*^*$ and $\beta \in \{s, w\}$, we define:

$$rc_\beta(L, P, F) = \{\pi \in L \mid rc_\beta(\pi; P, F) = \texttt{true}\}.$$

We now introduce the concept of an *accepting network of evolutionary picture processors with filtered connections* (ANEPPFC for short). An ANEPPFC is a 9-tuple $\Gamma = (V, U, G, \mathcal{R}, \mathcal{N}, \alpha, \beta, x_I, Out)$, where:

- V and U are the input and network alphabet, respectively, $V \subseteq U$.
- $G = (X_G, E_G)$ is an undirected graph without loops with the set of vertices X_G and the set of edges E_G. Each edge is given in the form of a binary set. G is called the *underlying graph* of the network.
- $\mathcal{R} : X_G \longrightarrow 2^{CSub_U} \cup 2^{RSub_U} \cup 2^{CDel_U} \cup 2^{RDel_U}$ is a mapping which associates with each node the set of evolutionary rules that can be applied in that node. Note that each node is associated only with one type of evolutionary rules, namely for every $x \in X_G$ either $\mathcal{R}(x) \subset CSub_U$ or $\mathcal{R}(x) \subset RSub_U$ or $\mathcal{R}(x) \subset CDel_U$ or $\mathcal{R}(x) \subset RDel_U$ holds.
- $\alpha : X_G \longrightarrow \{*, \leftarrow, \rightarrow, \uparrow, \downarrow\}$; $\alpha(x)$ gives the action mode of the rules of node x on the pictures existing in that node.
- $\mathcal{N} : E_G \longrightarrow 2^U \times 2^U$ is a mapping which associates with each edge $e \in E_G$ the disjoint sets $\mathcal{N}(e) = (P_e, F_e)$.
- $\beta : E_G \longrightarrow \{s, w\}$ defines the filter type of an edge.
- $x_I \in X_G$ is the *input* node and $Out \subset X_G$ is the set of *output* nodes of Γ.

We say that $card(X_G)$ is the size of Γ. A *configuration* of an ANEPPFC Γ as above is a mapping $C : X_G \longrightarrow 2_f^{U_*^*}$ which associates a finite set of pictures with every node of the graph. A configuration may be understood as the sets of pictures which are present in any node at a given moment. Given a picture $\pi \in V_*^*$, the initial configuration of Γ on π is defined by $C_0^{(\pi)}(x_I) = \{\pi\}$ and $C_0^{(\pi)}(x) = \emptyset$ for all $x \in X_G - \{x_I\}$.

A configuration can change via either an *evolutionary step* or a *communication step*. When changing via an evolutionary step, each component $C(x)$ of the configuration C is changed in accordance with the set of evolutionary rules M_x associated with the node x and the way of applying these rules $\alpha(x)$. Formally, we say that the configuration C' is obtained in *one evolutionary step* from the configuration C, written as $C \Longrightarrow C'$, iff

$$C'(x) = M_x^{\alpha(x)}(C(x)) \text{ for all } x \in X_G.$$

When changing by a communication step, each node processor $x \in X_G$ sends one copy of each picture it has to every node processor y connected to x, provided it can pass the filter of the edge between x and y. It keeps no copy of these

picture but receives all the pictures sent by any node processor z connected with x providing that they can pass the filter of the edge between x and z.

Formally, we say that the configuration C' is obtained in *one communication step* from configuration C, written as $C \vdash C'$, iff

$$C'(x) = (C(x) \setminus (\bigcup_{\{x,y\} \in E_G} rc_{\beta(\{x,y\})}(C(x), \mathcal{N}(\{x,y\}))))$$
$$\cup (\bigcup_{\{x,y\} \in E_G} rc_{\beta(\{x,y\})}(C(y), \mathcal{N}(\{x,y\})))$$

for all $x \in X_G$.

Let Γ be an ANEPPFC, the computation of Γ on an input picture $\pi \in V_*^*$ is a sequence of configurations $C_0^{(\pi)}, C_1^{(\pi)}, C_2^{(\pi)}, \ldots$, where $C_0^{(\pi)}$ is the initial configuration of Γ on π, $C_{2i}^{(\pi)} \Longrightarrow C_{2i+1}^{(\pi)}$ and $C_{2i+1}^{(\pi)} \vdash C_{2i+2}^{(\pi)}$, $\forall i \geq 0$. Note that configurations are changed by alternative steps. By the previous definitions, each configuration $C_i^{(\pi)}$ is uniquely determined by $C_{i-1}^{(\pi)}$. A computation is said to be *weak (strong) accepting*, if there exists a configuration in which the set of pictures existing in at least one output node (all output nodes) is non-empty. The *picture language weakly (strongly) accepted* by Γ is

$$L_{wa(sa)}(\Gamma) = \{\pi \in V_*^* \mid \text{ the computation of } \Gamma \text{ on } \pi \text{ is a weak (strong) accepting one}\}.$$

In network theory, some types of underlying graphs are common like *rings, stars, grids*, etc. Networks of evolutionary strings processors, seen as language generating or accepting devices, having underlying graphs of these special forms have been considered in several papers, see, e.g., [12] for an early survey. On the other hand, the ANEPPs considered in [1] are also complete. Starting from the observation that every ANEPPFC can be immediately transformed into an equivalent ANEPPFC with a complete underlying graph (the edges that are to be added are associated with filters which make them useless), for the sake of simplicity we discuss in what follows ANEPPFCs with underlying graphs having useful edges only. Note that this is not always possible for ANEPPs.

We denote by $\mathcal{L}_{wa}(ANEPPFC)$ and $\mathcal{L}_{sa}(ANEPPFC)$ the class of picture languages weakly and strongly accepted by ANEPPFCs, respectively.

3 Preliminary Results

The following two notions will be very useful in the sequel. If h is a one-to-one mapping from U to W and $\Gamma = (V, U, G, \mathcal{R}, \mathcal{N}, \alpha, \beta, x_I, Out)$, with $G = (X_G, E_G)$, is an ANEPPFC, then we denote by Γ_h the ANEPPFC $\Gamma_h = (h(V), h(U), G, h(\mathcal{R}), h(\mathcal{N}), \alpha, \beta, x_I, Out)$, where $h(\mathcal{R}(x)) = \{h(a) \to h(b)(X) \mid a \to b(X) \in \mathcal{R}(x)\}$, for any $x \in X_G$ and $h(\mathcal{N}(e)) = (h(P_e), h(F_e))$ for any $e \in E_G$.

We first establish a useful relationship between the classes $\mathcal{L}_{wa}(ANEPPFC)$ and $\mathcal{L}_{sa}(ANEPPFC)$. As it was expected, we have:

Theorem 1. $\mathcal{L}_{wa}(ANEPPFC) \subseteq \mathcal{L}_{sa}(ANEPPFC)$.

Proof. Actually, we prove a bit more general result, namely that for every ANEPPFC Γ there exists an ANEPPFC Γ' with one output node only and $L_{wa}(\Gamma) = L_{wa}(\Gamma') = L_{sa}(\Gamma')$. W.l.o.g. we assume that there is no edge connecting any two output nodes of Γ. We may further assume that each of these nodes is a substitution node containing all rules $a \to a(-)$ applied in the $*$ mode, for all symbols a in the working alphabet of Γ. In order to get Γ' it suffices to choose one of the output nodes of Γ, consider it the only output node of Γ', and connect it to each output node of Γ. Each such connection is filtered by a weak filter with the set of permitting symbols formed by all symbols of Γ and an empty set of forbidding symbols. □

We continue the series of preliminary results with two simple examples which lay the basis for further results.

Example 1. *Let L be the set of all pictures $\pi \in V_2^*$ with identical rows over the alphabet V. The language L can be formally described as*

$$L = \{\pi \in V_2^m \mid \pi(1,i) = \pi(2,i),\ i \in [m],\ m \geq 1\}.$$

L can be weakly accepted by the following ANEPPFC with $2 \cdot card(V) + 3$ nodes, namely x_I, x_a, x'_a, for every $a \in V$, x_{del}, one output node only, namely x_O, and the working alphabet $U = V \cup \{X_a, Y_a, X'_a, Y'_a \mid a \in V\}$. The nodes, different than x_O which has an empty set of rules, are defined as follows:

Node	$\mathcal{R}$	α
x_I	$\{a \to X_a(-), a \to X'_a(-) \mid a \in V\}$	$\uparrow$
$x_a, a \in V$	$\{a \to Y_a(-)\}$	$\downarrow$
$x'_a, a \in V$	$\{a \to Y'_a(-)\}$	$\downarrow$
x_{del}	$\{X_a \to \varepsilon(\vert) \mid a \in V\}$	$\leftarrow$

Further, the edges of the underlying graph and the filters associated with them are defined in the following way:

Edge	P	F	β
$\{x_I, x_a\}, a \in V$	$\{X_a\}$	$U \setminus (V \cup \{X_a\})$	s
$\{x_a, x_{del}\}, a \in V$	$\{X_a, Y_a\}$	$U \setminus (V \cup \{X_a, Y_a\})$	s
$\{x_{del}, x_I\}$	V	$U \setminus V$	w
$\{x_I, x'_a\}, a \in V$	$\{X'_a\}$	$U \setminus (V \cup \{X'_a\})$	s
$\{x'_a, x_O\}, a \in V$	$\{X'_a, Y'_a\}$	$U \setminus \{X'_a, Y'_a\}$	s

Let us follow a computation of this network on an input picture π. In x_I three situations are possible after the first evolutionary step: (i) an occurrence of some letter a on the first row of π is replaced by X_a, (ii) an occurrence of some letter a on the first row of π is replaced by X'_a, and (iii) π remains unchanged. If π is left unchanged, then it is sent to x_{del}, where it still remains unchanged, and it is sent back to x_I. We consider, the first non-trivial case, namely when an

occurrence of some letter a on the first row of π is replaced by X_a. All these pictures (remember that if the first row of π contains more than one occurrence of a, then each such occurrence is replaced by X_a in different copies of π) are sent to x_a. After the next evolutionary step two situations regarding each of these pictures are possible: (i) an occurrence of a on the last row of each picture, say ρ, is replaced by Y_a, or (ii) ρ remains unchanged. Note that in the second case, ρ does not contains any a on its last row. If ρ remains unchanged, then it is sent back to x_I and it either remains forever in x_I, provided that a letter $b \neq a$ is replaced by X_b, or it is sent back and forth between x_I and x_a. If an occurrence of a on the last row of ρ is replaced by Y_a, then all these pictures are sent to x_{del}. Note that each of these pictures contains exactly one occurrence of X_a. In x_{del} one tries to delete the leftmost column of all these pictures provided that this column contains X_a. If this process is not successful, then the pictures will continue to go forth and back between x_a and x_{del}. If the leftmost column of some picture is successfully deleted in x_{del}, then that picture can continue the computation in x_I provided that it contains only letters from V.

We now consider the case when an occurrence of some letter a on the first row of π is replaced by X'_a in x_I. In a similar way as that described above, all these pictures arrive in x'_a, where an occurrence of a on the last row of each picture is replaced by Y'_a and then only at most one picture can enter x_O, that is the picture $\begin{matrix} X'_a \\ Y'_a \end{matrix}$. By these explanations, it follows that every input picture with a different number of rows than two cannot be accepted. □

Clearly, the language of all pictures of size $(n, 2)$, $n \geq 1$, over a given alphabet V, where the two columns are identical can also be accepted by an ANEPPFC. The role of the next example is to show how two ANEPPFCs can be combined in order to form a new ANEPPFC. To this aim, we extend the network from Example 1 to accept the language of all pictures (of any size) having two identical rows.

Example 2. *Let L be the set of all pictures $\pi \in V_*^*$ with two identical rows over the alphabet V. The language L can be formally described as*
$L = \{\pi \in V_n^m \mid \exists i, j \in N, 1 \leq i \neq j \leq n \ (\pi(i,k) = \pi(j,k)),\ k \in [m], n, m \geq 1\}$.

In what follows we assume that the same alphabet V is used in Examples 1 and 2. First, we construct the ANEPPFC $\Gamma_1 = (V, U_1, G_1, N_1, \alpha_1, \beta_1, y_I, y_O)$, $G_1 = (X_{G_1}, E_{G_1})$, of size 3 with the working alphabet $U_1 = V \cup \{\bar{a} \mid a \in V\}$, and the nodes of $X_{G_1} = \{y_I, y_{del}, y_O\}$ defined by:

Node	$\mathcal{R}$	α
y_I	$\{b \to \varepsilon(-) \mid b \in V\}$	$*$
y_{del}	$\{b \to \varepsilon(-) \mid b \in V\}$	$*$
y_O	$\{a \to \bar{a}(-) \mid a \in V\}$	$*$

The edges of E_{G_1} together with the filters associated with them are defined by:

Edge	P	F	β_1
$\{y_I, y_{del}\}$	V	$\{\bar{a} \mid a \in V\}$	w
$\{y_I, y_O\}$	V	$\{\bar{a} \mid a \in V\}$	w
$\{y_{del}, y_O\}$	V	$\{\bar{a} \mid a \in V\}$	w

The informal idea of the role of this network is the following one. In the nodes y_I and y_{del} some nondeterministically chosen rows are repeatedly deleted from the pictures visiting these nodes several times. Note that a copy of any picture going out from y_{del} and y_I may enter y_O. As soon as a picture arrives in y_O and an occurrence of a symbol a from that picture is replaced by $\bar{a}$, the picture remains blocked in this node, until all its symbols are replaced with their barred copies.

We now consider the ANEPPFC $\Gamma = (V, U, G, N, \alpha, \beta, x_I, x_O)$ from Example 1 and the one-to-one mapping $h : U \longrightarrow \{\bar{a} \mid a \in V\} \cup (U \setminus V)$ defined by $h(a) = \bar{a}$, $a \in V$, and $h(b) = b$, $b \in U \setminus V$. Let Γ_2 be the ANEPPFC obtained from Γ_h by replacing $h(U)$ with $U_1 \cup U$ wherever $h(U)$ appears in the definition of parameters of Γ_h. We now connect y_O of Γ_1 with x_I of Γ_2 and impose that a picture cannot go out from y_O unless all its symbols were substituted by barred copies. Furthermore, besides all symbols in V, the set of forbidding symbols of the filter on the edge $\{y_O, x_I\}$ contains all symbols $X_{\bar{a}}$, $X'_{\bar{a}}$, $Y_{\bar{a}}$, $Y'_{\bar{a}}$ for $a \in V$. We claim that the new network weakly accepts L. Indeed, the subnetwork Γ_2 can start to work when it receives pictures having barred symbols only. By the above explanations, they must be pictures with only two rows that are barred copies of two rows randomly selected from the input picture. □

In what follows, instead of giving all the details of how two networks are merged, as in Example 2, we simply say that the pictures processed by the network Γ_1 are given as inputs to the network Γ_2 suitably modified.

4 Comparison with Other Devices

In this section we compare the classes $\mathcal{L}_{wa}(ANEPPFC)$ and $\mathcal{L}_{sa}(ANEPPFC)$ of picture languages weakly and strongly accepted by ANEPPFCs, respectively, with $\mathcal{L}(LOC)$ and $\mathcal{L}(REC)$ denoting the classes of local and recognizable picture languages, respectively, see [7].

Theorem 2. $\mathcal{L}_{wa}(ANEPPFC) \setminus \mathcal{L}(REC) \neq \emptyset$.

Proof. We first claim that the following language

$$L = \{\pi \in V_{2n}^m \mid n, m \geq 1, (\pi(n, i) = \pi(n+1, i)), \forall i \in [m]\}$$

is not recognizable, provided that $card(V) \geq 2$. The proof is identical to that for the same statement in [1]. As work [1] is not accessible yet we give it here again. Clearly, L consists of all pictures that can be written in the form $\pi_1 ® \pi_2$, where π_1, π_2 are pictures of the same size and the last row of π_1 is equal to the first row of π_2. Assume that L is recognizable and let $L = h(L')$, where h is a projection from some alphabet U to V and $L' \subseteq U_*^*$ is a local language. For two

positive integers n, m, let $L(n, m)$ be the subset of L formed by all pictures that can be written in the form $\pi_1 ® \pi_2$ with π_1, π_2 as above but satisfying also the following two conditions:

- both π_1 and π_2 are of size (n, m);
- neither π_1 nor π_2 contains two consecutive identical rows.

Therefore, there exists a subset $L'(n, m)$ of L' such that $L(n, m) = h(L'(n, m))$ for all n, m. Let m be fixed; as every set $L(n, m)$ is not empty for all values of n, it follows that all sets $L'(n, m)$ are nonempty as well.

Therefore, there are two pictures $\rho \in L'(n_1, m)$ and $\tau \in L'(n_2, m)$, with $n_1 \neq n_2$ such that the stripe rectangle of size $(2, m)$ consisting of the n_1-th and $(n_1 + 1)$-th rows in ρ equals the stripe rectangle of size $(2, m)$ consisting of the n_2-th and $(n_2 + 1)$-th rows in τ. Consequently, both pictures obtained from ρ and τ by interchanging their first halves with each other are in L'. However, the projection by h of any of these pictures is not in L, a contradiction.

We now prove that the language

$$L = \{\pi \in V_{2n}^m \mid n, m \geq 1, \pi(n, i) = \pi(n + 1, i), \forall i \in [m]\}$$

is in $\mathcal{L}_{wa}(ANEPPFC)$ for any alphabet V. We give only the description of the network processing the input pictures until they are sent to the input node of the network from Example 1 suitably modified. The five nodes of this network are defined as follows:

Node	$\mathcal{R}$	α
x_I	$\{a \to X(-), a \to a'(-) \mid a \in V\}$	$\uparrow$
x_1	$\{a \to Y(-) \mid a \in V\}$	$\downarrow$
x_2	$\{X \to \varepsilon(-)\}$	$\uparrow$
x_3	$\{Y \to \varepsilon(-)\}$	$\downarrow$
x_4	$\{a \to a'(-) \mid a \in V\}$	$*$

We now define the edges of this network with the filters associated with them:

Edge	P	F	β_1
$\{x_I, x_1\}$	$\{X\}$	$\{a' \mid a \in V\} \cup \{Y\}$	s
$\{x_I, x_4\}$	$\{a' \mid a \in V\}$	$\{X, Y\}$	w
$\{x_1, x_2\}$	$\{X, Y\}$	$\{a' \mid a \in V\}$	s
$\{x_2, x_3\}$	$\{Y\}$	$\{a' \mid a \in V\} \cup \{X\}$	s
$\{x_3, x_I\}$	V	$\{a' \mid a \in V\} \cup \{X, Y\}$	w

The working mode of this network is rather simple. In the input node the first row of the picture is marked either for deletion (if a symbol of the first row was replaced by X) or for the checking phase. If the first row was marked for deletion, the picture goes to the node x_1 where the last row is marked for deletion. When the picture contains X and Y it enters x_2. Note that when a picture enters x_2 it may contains more than one occurrence of X on its first row but only one Y on its last row. The first and the last row are deleted in the nodes x_2 and x_3, and the process resumes in the input node x_I.

Let us now see what happens with a picture marked for the checking phase in the input node. This picture enters nodes x_4. We connect this node with the input node of the network in Example 1 suitable modified and impose that a picture can enter the input node of this network only if all its symbols are primed copies. Note that in the process of changing all symbols into their primed copies the picture can enter x_I several time. Pictures entering x_I may either go back to x_3 for continuing the computational process or remain in x_I forever. □

We do not know whether the inclusion $\mathcal{L}(REC) \subset \mathcal{L}_{wa}(ANEPPFC)$ holds, however a large part of $\mathcal{L}(REC)$ is included in $\mathcal{L}_{wa}(ANEPPFC)$ as the next result states. We recall that the complement of any local language is recognizable [7].

Theorem 3. *The complement of every local language can be weakly accepted by an ANEPPFC.*

Proof. We start with an informal argument such that the formal proof can be understood easily. The argument starts with the observation that one can construct a network that weakly accepts only a fixed picture of size $(2, 2)$. Now, if L is a local language over the alphabet V defined by the set F of $(2, 2)$-tiles, then we consider the set F^c of all $(2, 2)$-tiles over V that do not belong to F. This network is made up of mainly two disjoint subnetworks: one subnetwork weakly accepts all pictures of size (n, m), with $n, m \geq 2$, in the complement of L, while the other subnetwork weakly accepts all pictures of size $(1, n)$ and $(n, 1)$ with $n \geq 1$. As the construction of the latter network is pretty simple, we discuss here the former one in detail.

The rough idea of the network weakly accepting all the pictures of size (n, m), with $n, m \geq 2$, in the complement of L is the following one. It consists of a subnetwork that cuts an arbitrary subpicture of the input picture. This subpicture is sent to every subnetwork from a set of completely disjoint networks each one accepting exactly one picture from F^c.

Formally, we assume that F^c has the tiles $t_1, t_2, \ldots t_n$ for some $n \geq 1$ and $t_i = \begin{matrix} a_i & b_i \\ c_i & d_i \end{matrix}$, where $a_i, b_i, c_i, d_i \in V$. The shape of this network is shown in Figure 1 while its nodes are defined as follows:

Node	$\mathcal{R}$	α
x_I	$\emptyset$	$*$
x_1	$\{a \to \varepsilon(-) \mid a \in V\}$	$\uparrow$
x_2	$\{a \to \varepsilon(-) \mid a \in V\}$	$\downarrow$
y_1	$\{a \to \varepsilon(\vert) \mid a \in V\}$	$\leftarrow$
y_2	$\{a \to \varepsilon(\vert) \mid a \in V\}$	$\rightarrow$
$z_i^1, 1 \leq i \leq n$	$\{a_i \to a_i^{11}(-)\}$	$\uparrow$
$z_i^2, 1 \leq i \leq n$	$\{b_i \to b_i^{12}(-)\}$	$\uparrow$
$z_i^3, 1 \leq i \leq n$	$\{c_i \to c_i^{21}(-)\}$	$\downarrow$
$z_i^4, 1 \leq i \leq n$	$\{d_i \to d_i^{22}(-)\}$	$\downarrow$
$z_i^5, 1 \leq i \leq n$	$\{a_i^{11} \to \varepsilon(\vert)\}$	$\leftarrow$
$z_i^{out}, 1 \leq i \leq n$	$\emptyset$	$*$

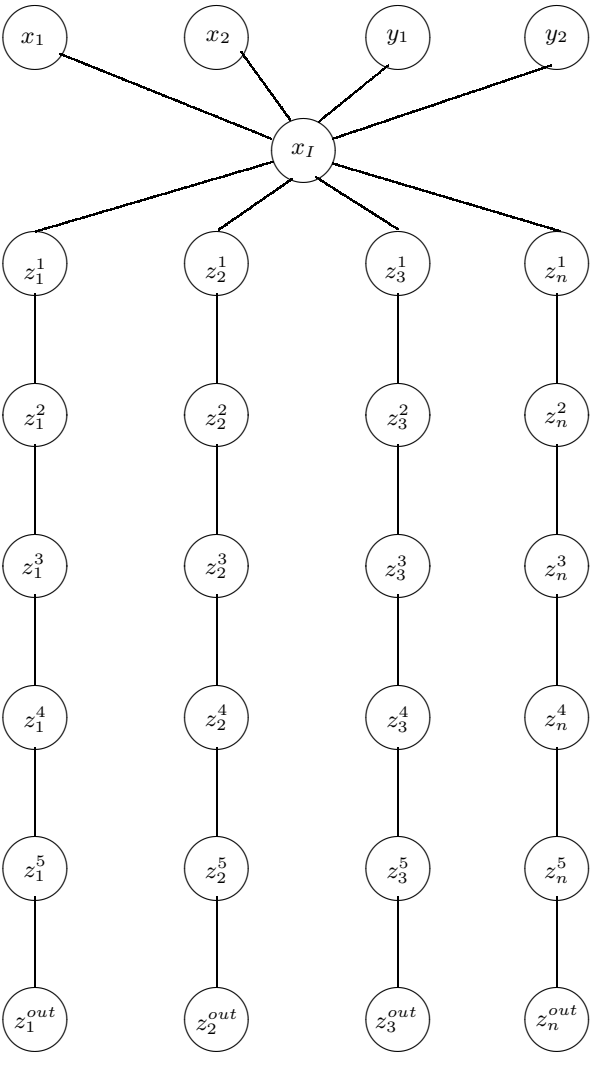

Fig. 1.

The role of the nodes defined above is as follows:

- Nodes x_1 and x_2 delete the first and the last row of a picture, respectively.
- Nodes y_1 and y_2 delete the leftmost and the rightmost row of a picture, respectively.
- Nodes z_i^1, z_i^2, z_i^3, z_i^4, z_i^5 and z_i^{out} check whether or not a picture equals the tile t_i, $1 \leq i \leq n$.

We now define the edges of this network with the filters associated with them:

Edge	P	F	β
$\{x_I, u\}, u \in \{x_1, x_2, y_1, y_2\} \cup \{z_i^1 \mid 1 \leq i \leq n\}$	V	$\{a_i^{11} \mid a_i \in V, 1 \leq i \leq n\}$	w
$\{z_i^1, z_i^2\}$	$\{a_i^{11}\}$	$\{b_i^{12}\}$	s
$\{z_i^2, z_i^3\}$	$\{a_i^{11}, b_i^{12}\}$	$\{c_i^{21}\}$	s
$\{z_i^3, z_i^4\}$	$\{a_i^{11}, b_i^{12}, c_i^{21}\}$	$\{d_i^{22}\}$	s
$\{z_i^4, z_i^5\}$	$\{a_i^{11}, b_i^{12}, c_i^{21}, d_i^{22}\}$	V	s
$\{z_i^5, z_i^{out}\}$	$\{b_i^{12}, d_i^{22}\}$	V	s

By the aforementioned explanations, it is rather easy to check that a computation of this network on a picture π leads to a non-empty node z_i^{out} if and only if π contains the tile t_i. □

We finish this work by pointing out a natural problem that regards the equality of the classes $\mathcal{L}_{wa}(ANEPPFC)$ and $\mathcal{L}_{sa}(ANEPPFC)$. Another attractive problem, in our view, concerns the relationships between these two classes and the classes $\mathcal{L}(LOC)$ and $\mathcal{L}(REC)$. Last but not least, a possible direct simulation of one model by another which is suggested by the results presented here will be in our focus of interest.

References

1. Bottoni, P., Labella, A., Mitrana, V., Sempere, J.: Networks of evolutionary picture processors (submitted)
2. Bottoni, P., Labella, A., Manea, F., Mitrana, V., Sempere, J.: Filter position in networks of evolutionary processors does not matter: a direct proof. In: The 15th International Meeting on DNA Computing and Molecular Programming (in press)
3. Drăgoi, C., Manea, F., Mitrana, V.: Accepting networks of evolutionary processors with filtered connections. Journal of Universal Computer Science 13, 1598–1614 (2007)
4. Drăgoi, C., Manea, F.: On the descriptional complexity of accepting networks of evolutionary processors with filtered connections. International Journal of Foundations of Computer Science 19, 1113–1132 (2008)
5. Errico, L., Jesshope, C.: Towards a new architecture for symbolic processing. In: Artificial Intelligence and Information-Control Systems of Robots 1994, pp. 31–40. World Scientific, Singapore (1994)
6. Giammarresi, D., Restivo, A.: Two-dimensional languages. In: [13], pp. 215–267
7. Giammarresi, D., Restivo, A.: Recognizable picture languages. Int. J. Pattern Recognition and Artificial Intelligence 6, 241–256 (1992)
8. Hillis, W.: The Connection Machine. MIT Press, Cambridge (1985)
9. Inoue, I., Takanami, I.: A survey of two-dimensional automata theory. In: Dassow, J., Kelemen, J. (eds.) IMYCS 1988. LNCS, vol. 381, pp. 72–91. Springer, Heidelberg (1989)
10. Margenstern, M., Mitrana, V., Jesús Pérez-Jímenez, M.: Accepting hybrid networks of evolutionary processors. In: Ferretti, C., Mauri, G., Zandron, C. (eds.) DNA 2004. LNCS, vol. 3384, pp. 235–246. Springer, Heidelberg (2005)

11. Manea, F., Martin-Vide, C., Mitrana, V.: On the size complexity of universal accepting hybrid networks of evolutionary processors. Mathematical Structures in Computer Science 17, 753–771 (2007)
12. Martín-Vide, C., Mitrana, V.: Networks of evolutionary processors: results and perspectives. In: Molecular Computational Models: Unconventional Approaches, pp. 78–114. Idea Group Publishing, Hershey (2005)
13. Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages. Springer, Berlin (1997)