Catalytic Petri Nets are Turing Complete*

Gabriel Ciobanu[1] and G. Michele Pinna[2]

[1] Institute of Computer Science, Romanian Academy, Iaşi, Romania
[2] Dipartimento di Matematica e Informatica, Università di Cagliari, Italy

Abstract. In this paper we introduce a class of Petri nets, called *catalytic Petri nets*, and a suitable firing strategy where transitions are fired only when they use tokens from specific places, called catalytic places. By establishing a one-to-one relationship with *catalytic membrane systems*, we can prove that the class of catalytic Petri nets with at least two catalytic places is Turing complete.

1 Introduction

Soon after their introduction in the early 60's, Petri nets have been acknowledged as a formalism for modeling distributed and concurrent computations and, from a formal language theoretic point of view, their *expressivity* has been investigated. On the one side, the classes of languages defined by Petri nets, i.e., sets of sequences of labeled or unlabeled transitions, have been studied from the beginning of the 70's (*e.g.* in [1] and references therein) and the research on this topic is still in progress. On another side the question of "how *expressive* Petri nets are" has been asked and the answer can be summarized as follows: Petri nets under the usual step firing strategy are not Turing complete (see [2] and the surveys [3, 4] for a comprehensive knowledge), while they are Turing complete under suitable assumptions on the firing strategy, either *maximality* (all the possible transitions are fired together) or *ordering* (a transition should fire as soon as it is enabled), as shown in [5] and [6].

In order to make Petri nets Turing complete, different approaches have been taken; notably the one which *extends* the kind of arcs considered. The classic extension is the one where *inhibitor* arcs are considered, *i.e.*, arcs where the absence of tokens in certain places is modeled ([7]), or *reset* arcs, *i.e.*, arcs with the characteristic of emptying the preset, regardless of the number of tokens present in the place ([8]). Other extensions of Petri Nets, not necessarily always Turing complete, allow the introduction of *non blocking* arcs or *transfer* arcs ([9] and [10]) or making the transitions *marking* dependent ([11]).

Summing up, to make Petri nets Turing complete either suitable extensions of the model have to be considered or quite *heavy* assumptions on the firing rule have to be made. In the former case many among the main features of Petri nets are retained, *e.g.*, the *locality* of firing or the *distributed* state, whereas in

* Work partially supported by the Regione Autonoma della Sardegna; grants L.R. 7/2007 CRP2-120 (Project TESLA) and CRP-17285 (Project TRICS)

the latter the characteristics that have made Petri nets an appealing and useful model for concurrent and distributed computations are lost.

In this paper, inspired by *membrane computing*, we investigate the expressivity of Petri nets looking at suitable firing strategy and at the *structure* of the nets itself. The connection between membrane computing and Petri nets has already been established (see chapter 15 of the Oxford Handbook of Membrane Computing [12] and references therein), where it is shown that to each kind of membrane systems it is possible to associate a suitable *labeled* Petri net, where the labeling of transitions is used to model the membrane structure.

We establish a relation between Petri nets and membrane systems showing first that to each Petri net a membrane systems can be associated and vice versa, and then that various firing strategy definable on Petri nets are matched by corresponding evolutions steps in membrane systems. In particular we focus on membrane systems with only one membrane and on *catalytic* membrane systems, where the rules may *use* a *catalyst*, *i.e.*, an object needed for applying the rule but that it is never been *consumed*[3]. Then we use this relationship to establish the Turing completeness of a suitable class of Petri nets (intuitively the one that is translated into a catalytic membrane system) under a suitable firing strategy which establish that a particular (proper) subset of the enabled transitions are fired, namely the subset where the transitions involving catalytic places are used. We use results of membrane computing to prove that the number of catalytic places needed for Turing completeness is quite limited, namely just two.

The merit of the results presented in this paper is that they show that expressiveness of Petri nets can be increased without introducing suitable arcs, without requiring transitions fired upon a complete snapshot of the system, and therefore without loosing the locality of firing that is one of the main features of Petri nets. In other word the state is still *distributed*. Furthermore the relation established between Petri nets and membrane systems suggests that also other classes of Petri nets with *minimal parallelism* (*i.e.*, where a minimal number of concurrent transitions are considered) could be Turing complete.

The paper is organized as follows: in the next section we will recall the basic notions on Petri nets and their firing strategies, in section 3 we briefly recall the definition of (catalytic) membrane system and establish some expressiveness results. In section 4 we recall how to relate membrane systems to Petri nets and introduce the vice versa as well, and in section 5 we show that catalytic Petri nets with two catalysts are Turing complete when considering suitable firing strategies (which are not the maximal one, or based on ordered firing of transitions).

2 Petri Nets and firing strategies

Notations With $\mathbb{N}$ we denote the set of *natural numbers* including zero, and with $\mathbb{N}^+$ the set of positive natural numbers.

[3] This notion is quite different from the one of *read* arc on Petri nets.

Multisets Given a set S, a *multiset* over S is a function $m : S \to \mathbb{N}$ and with ∂S we denote the set of multisets of S. The *multiplicity* of an element s in m is given by $m(s)$. A multiset m over S is *finite* iff the set $dom(m) = \{s \in S \,|\, m(s) \neq 0\}$ is finite. All the multisets we consider in this paper are finite. A multiset m such that $dom(m) = \emptyset$ is called *empty* and it is denoted by $\mathbf{0}$. The cardinality of a multiset is defined as $\#(m) = \sum_{s \in S} m(s)$. Given a multiset in ∂S and a subset $\hat{S}$ of S, with $m|_{\hat{S}}$ we indicate the multiset over $\hat{S}$ such that $m|_{\hat{S}}(s) = m(s)$. We write $m \subseteq m'$ if $m(s) \leq m'(s)$ for all $s \in S$, and $m \subset m'$ if $m \subseteq m'$ and $m \neq m'$. The operator $\oplus$ denotes *multiset union*: $m \oplus m'(s) = m(s) + m'(s)$. The operator $\ominus$ denotes *multiset difference*: $m \ominus m'(s) =$ if $m(s) > m'(s)$ then $m(s) - m'(s)$ else 0. The *scalar product* of a number j with a multiset m is $(j{\cdot}m)(s) = j{\cdot}(m(s))$. Sometimes a multiset $m \in \partial S$ is written as $\oplus_{s \in S} n_s{\cdot}s$, where $n_s = m(s)$; we omit the summands where n_s is equal 0. If $m \in \partial S$, we denote by $[\![m]\!]$ the multiset defined as $[\![m]\!](a) = 1$ if $m(a) > 0$, and $[\![m]\!](a) = 0$ otherwise; sometimes $[\![m]\!]$ is identified (used interchangeably) with the corresponding subset $\{a \in S \mid [\![m]\!](a) = 1\}$ of S (observe that this is different from $dom(m)$: this one is the set of elements on S such that $m(s) > 0$, whereas $[\![m]\!]$ is a particular multiset associated to m).

Petri Nets A Petri net is a tuple $N = ((S, T, F, m_0), \mathfrak{S})$ where S is a set of *places*, T is a set of *transitions*, $F : (S \times T) \cup (T \times S) \to \mathbb{N}$ is a *flow relation*, $m_0 \in \partial S$ is the initial marking and $\mathfrak{S} \subseteq S$ is a set of *final* places. Furthermore $S \cap T = \emptyset$. With ${}^\bullet x$ ($x^\bullet$, respectively) we indicate the multiset $F(\_, x)$ ($F(x, \_)$, respectively), and they are called the *preset* (*postset*, respectively) of x. We assume that for each transition t, $dom({}^\bullet t) \neq \emptyset$.

Given a net $N = ((S, T, F, m_0), \mathfrak{S})$, we say that N is a *state machine* iff $\forall t \in T.\ |dom({}^\bullet t)| = |dom(t^\bullet)| = 1$; and N is an *input state machine* iff $\forall t \in T.\ |dom({}^\bullet t)| = 1$ and ${}^\bullet t = [\![{}^\bullet t]\!]$. Let $S' \subseteq S$ be a subset of places, the *subnet* of N generated by S' is the net defined as follows: $N@S' = ((S', T@S', F@S', m_0@S'), \mathfrak{S} \cap S')$ where $T@S' = \{t \in T \mid \exists s \in S' \text{ such that either } F(s,t) > 0 \text{ or } F(t,s) > 0\}$, $F@S'$ is the restriction of F to S', and $m_0@S' = m_0|_{S'}$.

Firing strategies We discuss now the *dynamic* of Petri nets. A multiset over S is called a marking for the net. Let $m \in \partial S$ be a marking of a net. A multiset $U \in \partial T$ of transitions is *enabled* under m if for all $s \in S$ $\sum_{t \in T} U(t) \cdot F(s,t) \leq m(s)$, and it is written as $m\,[U\rangle_{st}$. If a finite multiset $U \in \partial T$ is *enabled* at a marking m (i.e., $m\,[U\rangle_{st}$), then U may *fire* reaching a new marking m' defined as $m'(s) = m(s) + \sum_{t \in T} U(t) \cdot (F(t,s) - F(s,t))$, for all $s \in S$. We write $m\,[U\rangle_{st}\,m'$, and call U a *step*. We observe that if $t \in dom(U)$ is such that either ${}^\bullet t = [\![{}^\bullet t]\!]$ or $t^\bullet = [\![t^\bullet]\!]$, then the firing of the transition t consumes one token from each place in $dom({}^\bullet t)$ or produces one token in each place in $dom(t^\bullet)$. When considering steps, we often omit the subscript st, thus we write simply $m\,[U\rangle$ and $m\,[U\rangle\,m'$.

A *step firing sequence* is defined as follows: m_0 is a step firing sequence, and if $m_0\,[U_1\rangle\,m_1 \ldots m_{n-1}\,[U_n\rangle\,m_n$ is a step firing sequence and $m_n\,[U_{n+1}\rangle\,m_{n+1}$, then also $m_0\,[U_1\rangle\,m_1 \ldots m_{n-1}\,[U_n\rangle\,m_n\,[U_{n+1}\rangle\,m_{n+1}$ is a step firing sequence. Given a net $N = ((S.T, F, m_0), \mathfrak{S})$, a marking m is *reachable* if there is a step firing

sequence leading to it, *i.e.*, $m_0\,[U_1\rangle\,m_1 \ldots m_{n-1}\,[U_n\rangle\,m_n$ with $m = m_n$. The set of reachable marking of the net N is denoted with $\mathfrak{M}(N)$. Given a Petri net N, it has been shown that the problem of deciding whether a given marking m is reachable from the initial one (*i.e.*, if $m \in \mathfrak{M}(N)$) is decidable (see [3] for a survey).

The *ordinary* firing strategy of Petri nets (just one enabled transition is fired at each marking, regardless of how many are *simultaneously* enabled) is an instance of the step firing strategy we have previously revised: in this case the multiset U is such that $U = [\![U]\!]$ and $\#(dom(U)) = 1$. We observe that, given a step $m\,[U\rangle\,m'$, it may be always *linearized* and m' can be reached from m with a possibly different step firing sequence.

Steps, and hence firing strategies, may have suitable characteristics. Here we concentrate on two of these, namely *maximality* and *place-awareness*. A step U enabled at a marking m is *maximal* iff each step U' such that $U \subset U'$ is such that $\neg\,(m\,[U'\rangle\,)$. A maximal step will be denoted with $m\,[U\rangle_{max}\,m'$, and a maximal step firing sequence is a step firing sequence where each step is maximal. The set of reachable marking of a net N with maximal step firing sequences is $\mathfrak{M}_{max}(N)$. In this case it holds that $\mathfrak{M}_{max}(N) \subseteq \mathfrak{M}(N)$ and the containment may be proper (indeed it often is, as reachability in the case of this firing strategy is undecidable [6]). Place-awareness concerns the requirement that suitable places are involved, if possible. More formally, consider a subset of places $\mathcal{S} \subseteq S$. A step U enabled at a marking m is $\mathcal{S}$-enabled iff for all $s \in \mathcal{S}$ either there exists a $t \in dom(s^\bullet)$ and $U(t) \neq 0$, or for all $t' \in dom(s^\bullet)$ it holds that $\neg m\,[t'\rangle$. We write $m\,[U\rangle_{\mathcal{S}}$ to indicate this, and the corresponding step called $\mathcal{S}$-step is denoted by $m\,[U\rangle_{\mathcal{S}}\,m'$. Firing sequences where each step is a $\mathcal{S}$-step are defined as usual, and the set of reachable markings under this firing strategy is $\mathfrak{M}_{\mathcal{S}}(N)$. To ease the notation we use a subscript to indicate which firing strategy is used: *step*, *maximal*, $\mathcal{S}$*-step*; when no subscript is used, we assume that it is the step firing strategy.

Given a net N and a marking m of $N = ((S, T, F, m_0), \mathfrak{S})$, we say that m is a *final* marking iff $m \in \mathfrak{M}(N)$ and for all $t \in T$ we have $\neg(m\,[t\rangle\,)$. We are interested in the reachable markings (under a firing strategy $fs \in \{step, max, \mathcal{S}\}$) that are also final: $\mathfrak{F}_{fs}(N) = \{m \in \mathfrak{M}_{fs}(N) \mid m \text{ is final}\}$. Among the final markings, we may be interested in considering either the numbers of tokens contained in certain places, or the marking in these places. Let $((S, T, F, m_0), \mathfrak{S})$ be a Petri net; we denote by $\mathfrak{F}^{\mathfrak{S}}_{fs}(N)$ and $\mathfrak{F}^{\#\mathfrak{S}}_{fs}(N)$ the sets $\{m|_{\mathfrak{S}} \mid m \in \mathfrak{F}_{fs}(N)\}$ and $\{\#(m|_{\mathfrak{S}}) \mid m \in \mathfrak{F}_{fs}(N)\}$, respectively.

3 Membrane systems with catalysts

Membrane systems (also called P systems) represent abstract computing models based on operations with multisets. They are inspired by the structure of (eukaryotic) cells, namely by the role of their membranes delimiting different compartments, and thus can help to understand information processes in the nature. The most comprehensive recent monograph is the Oxford Handbook of Membrane Computing [12]. Membrane computing was introduced in [13], vol-

ume [14] presents various applications, while the P Systems webpage presents recent developments.

Starting from the observation that there is an obvious parallelism in the cell biochemistry [15], and relying on the assumption that "if we wait enough, then all reactions which may take place will take place", a basic feature of the P systems is the maximally parallel way of using the rules: in each step, in each region of a system, we have to use a maximal multiset of rules. This maximal parallelism decreases the non-determinism of the systems evolution, and it provides enough power to get computational Turing completeness.

There are several types of membrane systems (defined by various ingredients inspired from cell biology); we use one of the basic types, namely catalytic P systems. A catalytic P system is formed by a hierarchical membrane structure with uniquely labeled membranes, the whole structure being embedded in a single skin membrane. Each membrane contains a multiset of abstract objects (members of a finite alphabet) which is divided into two parts: the set C of catalysts, and the set of non-catalytic objects. Each membrane has assigned an initial multiset of objects and a fixed set of evolution rules of two possible types: catalytic rules $ca \to cv$, and non-cooperative rules $a \to v$, where c is a catalyst object, a is a non-catalyst object, and v is a (possibly empty) multiset of objects. A catalytic P system is called *purely* catalytic if it contains only catalytic rules. The rules are (usually) applied in the maximally parallel mode: at each computational step and in each membrane, the selected multiset of applicable rules must be maximal, i.e., unused objects do not allow to apply an additional rule.

We formalize now the notion of *flat* P system and of *flat* catalytic P system. It is shown in [16] that the flattened version of a transition P system has the same computability power as a non flattened one; thus we can use the flattened one without losing generality. The idea behind the notion of flattening is the following one: the membrane structure of a P system, capturing the intuition that a rule uses objects in specific compartment and produces objects in the same compartment or in compartments that are *close* to this one (*i.e.*, either in the one surrounding it or compartments that are immediately internal to the specific one).

Definition 1. *A* (flat) P system *is the 4-tuple* $\Pi_f = (O, w^0, R, O')$ *where*

- O *is a finite set of* objects*, and* $O' \subseteq O$ *are the* final *objects,*
- $w^0 \in \partial V$ *is a finite multiset of objects, called the* initial *configuration, and*
- R *is a finite set of rules of the form* $r = u \to v$*, with* $u, v \in \partial O$ *and* $u \neq \mathbf{0}$*.*

A configuration *of a membrane system is any finite multiset of objects.*

A flat membrane system Π_f *is called* catalytic *iff there is a designated subset* $O_C \subset O$ *of* catalysts *and the rules have the following form: either* $r = a \to v$ *with* $a \in O \backslash O_C$ *and* $v \in \partial(O \backslash O_C)$ *or* $r = ca \to cv$ *with* $a \in O \backslash O_C$*,* $v \in \partial(O \backslash O_C)$ *and* $c \in O_C$*. If all the rules are of the form* $r = ca \to cv$ *we say that the catalytic P system is* purely *catalytic. We denote catalytic P systems with* $C\Pi$*, and purely catalytic ones with* $CP\Pi$*.*

According to [16], any property proved for flat membranes could be also proved for non-flat ones.

Given a rule $r = u \to v$, with $lhs(r)$ we denote the multiset u and with $rhs(r)$ the multiset v. The dynamic of a P system is formalized in the following way: given a multiset of rules $\mathcal{R}$ ($\mathcal{R} \in \partial R$) and a configuration w, $w \overset{\mathcal{R}}{\Rightarrow} w'$ iff $\oplus_{r\in R}\mathcal{R}(r) \cdot lhs(r) \subseteq w$ and $w' = w \ominus (\oplus_{r\in R}\mathcal{R}(r) \cdot lhs(r)) \oplus (\oplus_{r\in R}\mathcal{R}(r) \cdot rhs(r))$. The transition $w \overset{\mathcal{R}}{\Rightarrow} w'$ is called an evolution step. A configuration w is reachable from the initial one iff there exists a finite sequence of evolution steps $w_1 \overset{\mathcal{R}_1}{\Rightarrow} w_1 \cdots w_n \overset{\mathcal{R}_r}{\Rightarrow} w_{n+1}$ such that $w_1 = w^0$ and $w = w_{n+1}$. The basic assumption is that each step $w \overset{\mathcal{R}}{\Rightarrow} w'$ is *maximal*, namely for each $\mathcal{R}' \supseteq \mathcal{R}$ it holds that $\oplus_{r\in R}\mathcal{R}'(r) \cdot lhs(r) \not\subseteq w$.

Computations are sequence of evolution steps, and a computation terminates (halts) iff at the configuration w reached by this computation the only possible evolution step is $\mathbf{0}$, *i.e.*, $w \overset{\mathbf{0}}{\Rightarrow} w$. The result of the computation is $w|_{O'}$. The sets *calculated* by a flat P system Π_f are $NO_{max}(\Pi) = \{\#(w|_{O'}) \mid$ there exists a halting computation (with maximal parallelism) and w is the reached configuration$\}$ and $PsO_{max}(\Pi) = \{w|_{O'} \mid$ there exists a halting computation (with maximal parallelism) and w is the reached configuration$\}$. The class *calculated* by a suitable class of P systems will be denoted with NO_{max} and PsO_{max}, respectively, overloading the notation.

As we recalled previously, in [16] it is shown that the flattening preserves the computability power and expressiveness of P systems. With Π^n we denote the membrane system with n membranes, $n \geq 1$. Consider a membrane system Π^n (not a flat one), we call $flat(\Pi^n)$ its flattened version; flattening a catalytic P system gives a flat catalytic P system.

A research topic in membrane computing is to find more realistic P systems from a biological point of view, and one target in this respect is to relax the condition of using the rules in a maximally parallel way. *Minimal* parallelism was introduced in [17]; it describes another way of applying the rules: if at least a rule from a set of rules associated with a membrane (or a region) *can* be used, then at least one rule from that membrane (or region) *must* be used, without any other restriction (e.g., more rules can be used, but we do not care how many). Even if it might look weak, this minimal parallelism still leads to Turing completeness for a suitable class of P systems, called symport/antiport P systems. The *minimal parallelism* stems out from the consideration that this way of using the rules ensures that all compartments (or regions) of the system evolve in parallel by using at least one rule, whenever such a rule is applicable. Considering the class of flat catalytic P systems, Turing completeness may be achieved relaxing the maximal parallelism requirement with the weaker one stating that for each catalyst c at least one catalytic rule $r = ca \to cv$ is used, if possible. Thus the question is how many catalysts are needed to obtain Turing completeness. With RE we denote the set of recursively enumerable sets of numbers and with $PsRE$ the set of recursively enumerable sets of Parikh vectors. Finally with $(C\Pi, n)$ we indicate the number of catalysts in the catalytic P system $C\Pi$.

A catalytic P system with a single membrane and only two catalysts has the power of a Turing machine, i.e., it can generate all the computable enumerable sets of (vectors of) natural numbers [18]. With $([p-]cat, n)$ we denote the class of flat (purely) catalytic P system with at least n catalysts. Assuming that the computations obey to the *minimal* parallel rule (*i.e.*, at least one rule involving each catalyst is used, if possible), the following theorems present the results used in Section 5.

Theorem 1. *$NO_{min}((cat, n)) = RE$ and $PsO_{min}((cat, n)) = PsRE$, for $n \geq 2$.*

Three catalysts are needed in the case of purely catalytic systems.

Theorem 2. *$NO_{min}((p - cat, n)) = RE$ and $PsO_{min}((p - cat, n)) = PsRE$, for $n \geq 3$.*

We end this part recalling that chapter 4 of the handbook provides a good survey of the computability power of catalytic P systems ([12]). The above results are presented in that chapter, together with other interesting results.

4 Relating Petri nets and membrane systems

We describe here how to associate a P system to a Petri net, and vice-versa.

From membrane systems to Petri nets The intuition of encoding a membrane system into a Petri nets is exactly the same of the encoding presented in [19] and further developed thereof (see [20] and references therein): to each object of the membrane system a place is associated, and to each rule belonging to a membrane system a transition is associated. Such a translation from membrane systems to Petri nets (with localities) can be applied to (almost) any kind of membrane systems, including catalytic membrane systems where the objects are partitioned into two subsets and the rules comply with a specific pattern,

Definition 2. *Starting from a membrane system $\Pi = (O, \mu, w^0, R, O')$, we can associate to it a net structure $\mathcal{F}(\Pi) = ((S, T, F, m), \mathfrak{S})$ such that*

- *$S = O$ and $T = \{t^r \mid r \in R\}$,*
- *for all transitions $t = t^r \in T$ and all the places $s \in S$ with $r = u \to v$, we define*

$$F(s,t) = \begin{cases} u(a) & \text{if } s = a \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad F(t,s) = \begin{cases} v(a) & \text{if } s = a \\ 0 & \text{otherwise} \end{cases}$$

- $m(s) = \begin{cases} w^0(a) & \text{if } s = a \\ 0 & \text{otherwise} \end{cases}$, *and*
- $\mathfrak{S} = O'$.

The multiplicity of an object v is modeled by the number of tokens in the place v. Obviously, this construction leads to a Petri net.

Following [19], we introduce two functions: one associating to a configuration of a membrane system a marking of its corresponding net, and another associating to the rules applied in an evolution of the membrane system a step in the net.

Definition 3. *Let $\Pi = (O, \mu, w^0, R, O')$ be a membrane system, and $\mathcal{F}(\Pi) = ((S, T, F, m), \mathfrak{S})$ be the associated Petri net. Let w be a configuration of Π. Then $\nu(w)$ is the marking defined by $\nu(w) = w$.*

Definition 4. *Let $\Pi = (O, \mu, w^0, R, O')$ be a membrane system, and $\mathcal{F}(\Pi) = ((S, T, F, m), \mathfrak{S})$ be the associated Petri net. Let $C \stackrel{\boldsymbol{R}}{\Longrightarrow} C'$ be an evolution step of Π. Then $\sigma(\boldsymbol{R})$ is the multiset defined by $\sigma(\boldsymbol{R})(t^j) = \boldsymbol{R}(r^j)$, for all $t^j \in T$.*

We can now state the main result contained in [19].

Theorem 3. *Let $\Pi = (O, \mu, w^0, R, O')$ be a membrane system, and $\mathcal{F}(\Pi) = ((S, T, F, m), \mathfrak{S})$ be the associated Petri net. $w \stackrel{\boldsymbol{R}}{\Longrightarrow}_{fs} w'$ iff $\nu(w)\,[\sigma(\boldsymbol{R})\rangle_{fs}\,\nu(w')$ with $fs \in \{step, max, \mathcal{S}\}$.*

This theorem essentially says that the net associated to a membrane system behaves in the same way, provided that in the net steps are performed accordingly to the same way the evolution steps in the membrane system are defined.

From the previous section we know that the sets of natural numbers or of multisets *calculated* by a membrane system Π under a suitable evolution strategy e are denoted with $\mathcal{N}_e(\Pi)$ and $\mathcal{P}s_e(\Pi)$, respectively. A consequence of Theorem 3 is the following one:

Theorem 4. *Let $\Pi = (O, \mu, w^0, R, O')$ be a membrane system, and $\mathcal{F}(\Pi) = ((S, T, F, m), \mathfrak{S})$ be the associated Petri net. Then $\mathfrak{F}^{\mathfrak{S}}_{fs}(\mathcal{F}(\Pi)) = \mathcal{P}s_{fs}(\Pi)$ and $\mathfrak{F}^{\mathfrak{S}}_{fs}(\mathcal{F}(\Pi)) = \mathcal{N}_{fs}(\Pi)$, with $fs \in \{step, max, \mathcal{S}\}$.*

Thus the function calculated by the net associated to the membrane system is the same (again provided that evolutions are applied following the chosen firing sequence/evolution strategy).

From Petri nets to membrane systems Given a Petri net, it is always possible to associate a *flat* membrane system to it. In the translation from membrane systems to Petri nets we have associated to each object a place. Now we do vice versa: to each place we associate an object. As for the purpose of this paper we are interested in *what* is calculated by a net, we explicitly point out the set of final places of a net.

Definition 5. *Let $N = ((S, T, F, m_0), \mathfrak{S})$ be a Petri net.*
Then $\mathcal{K}(N) = (O, \mu, w^0, R, O')$ is the membrane system defined as follows:

- *$O = S$ and $w^0(s) = m(s)$,*

– *for all* $t \in T$ *define a rule* $r^t = u \to v$ *in* R *where* $u = {}^\bullet t$ *and* $v = t^\bullet$, *and*
– $O' = \mathfrak{S}$.

Moreover, we define two mappings ξ and η which associate the objects to the places and the rules to the transitions, respectively. Following Definition 5, we have $\xi(s) = s$, and $\eta(t)$ is the rule r_i^t with $\mathcal{L}(t) = i$.

The first result is that $\mathcal{K}(N)$ is indeed a flat membrane system.

Proposition 1. *Let* $N = ((S, T, F, m_0), \mathfrak{S})$ *be a Petri net. Then* $\mathcal{K}(N)$ *is a flat membrane system.*

Moreover, the net associated to $\mathcal{K}(N)$ is indeed N.

Proposition 2. *Let* $N = ((S, T, F, m_0), \mathfrak{S})$ *be a Petri net. Then* $\mathcal{F}(\mathcal{K}(N)) = N$.

When we consider flat membrane systems, we also have the following result.

Proposition 3. *Let* $\Pi = (O, \mu, w^0, R, O')$ *be a flat membrane system. Then* $\mathcal{K}(\mathcal{F}(\Pi)) = \Pi$.

We extend the two functions ξ and η to markings and multisets of transitions. These extensions relate markings to configurations, and steps in nets to evolution steps in membrane systems.

Definition 6. *Let* $N = ((S, T, F, m_0), \mathfrak{S})$ *be a Petri net and* $\mathcal{K}(N)$ *be its associate membrane system. Let* m *be a marking of* N. *Then* $\xi(m)$ *is the configuration defined by* $\xi(m)(s) = m(s)$, *for all* $s \in S$.

Definition 7. *Let* $N = ((S, T, F, m_0), \mathfrak{S})$ *be a Petri net and* $\mathcal{K}(N)$ *be the associate membrane system. Let* $m\,[U\rangle\, m'$ *be a step of* N. *Then* $\eta(U)$ *is the evolution step defined by* $\eta(U)(r^t) = U(t)$, *for all* $r^t \in R$.

We are now ready to present the main result of this section.

Theorem 5. *Let* $N = ((S, T, F, m_0), \mathfrak{S})$ *be a Petri net and* $\mathcal{K}(N)$ *be its associate membrane system. For* $fs \in \{step, max, \mathcal{S}\}$, *we have*

$$m\,[U\rangle_{fs}\, m'' \quad \textit{iff} \quad \xi(m) \stackrel{\eta(U)}{\Longrightarrow}_{fs} \xi(m').$$

A consequence of Theorem 5 is the following one:

Theorem 6. *Let* $N = ((S, T, F, m_0), \mathfrak{S})$ *be a Petri net and* $\mathcal{K}(N)$ *be the associated flat membrane system. Then*

$$\mathfrak{F}^{\mathfrak{S}}_{fs}(\mathcal{F}(\Pi)) = \mathcal{P}s_{fs}(\Pi) \quad \textit{and} \quad \mathfrak{F}^{\mathfrak{S}}_{fs}(\mathcal{F}(\Pi)) = \mathcal{N}_{fs}(\Pi),$$

where $fs \in \{step, max, \mathcal{S}\}$.

5 Catalytic Petri nets are Turing complete

In this section we present the main result of the paper, namely that there is a suitable class of Petri Nets (namely catalytic Petri Nets with at least two catalysts) which is able to calculate a recursive enumerable set under a suitable firing strategy. We first introduce the class of catalytic nets, then show the relations between catalytic nets and catalytic membrane system, and finally use the results of Section 3 to show that catalytic nets are Turing complete.

Catalytic Petri nets The main motivation behind catalytic net is to *control* the firing of suitable transitions *locally* by putting tokens in suitable places (that maintain the same number of tokens during the whole execution).

Definition 8. *Let $N = ((S, T, F, m), \mathfrak{S})$ be a Petri net. N is* catalytic *iff the set of places S is partitioned into two disjoint sets $\mathcal{C}$ and $\mathcal{V}$ such that*

1. *the subnet $N@\mathcal{V}$ is an input state machine, and*
2. *the subnet of $N@\mathcal{C}$ is a state machine, and for all $t \in T@\mathcal{C}$ we have ${}^\bullet t = t^\bullet$ and $\#({}^\bullet t) = 1$.*

A net $N = ((S, T, F, m), \mathfrak{S})$ is said to be purely catalytic *iff $T = T@\mathcal{C}$.*

Places in a catalytic net are partitioned into two subsets, the catalytic places ($\mathcal{C}$) and the non catalytic ones ($S \setminus \mathcal{C}$). Condition 1 states that each transition may consume tokens from only one non catalytic place, whereas condition 2 says that each transition may use at most one token from a catalytic place, and the used token is made again available for further use.

Catalytic Petri nets $N = ((S, T, F, m), \mathfrak{S})$ are abbreviated by CPN. Sometime we explicitly indicate the set of catalytic places and their number, and so a catalytic net is presented as $N = ((S, T, F, m, \mathcal{C}, n), \mathfrak{S})$ with $|\mathcal{C}| = n$.

The intuition behind catalytic Petri nets is the following: a transition t which uses a catalyst fires only if there is a token in the catalytic places associated to the transition. Tokens in catalytic places may be consumed/produced by transitions, but the transition using these tokens as catalysts do leave the token in the place. It is worth to stress the difference among catalytic nets and Petri nets with read arcs of Montanari and Rossi ([21]): in their case, if two transitions test for the presence of a token in place (with read arcs) it is enough to have a token in that place, whereas in catalytic nets if two transitions use the *same* catalytic place, this place must contain enough tokens for both transitions.

We *specialize* again the notion of enabling. Let $N = ((S, T, F, m, \mathcal{C}, n), \mathfrak{S})$ be a CPN. The step U is catalyticly enabled at the marking m iff it is enabled at m and $\forall c \in \mathcal{C}$ either there is a transition $t \in dom(U)$ such that ${}^\bullet t(c) \neq 0$ or $\forall t \in dom(c^\bullet)$ it holds that $\neg m\,[t\rangle$.

In other words, a step is catalyticly enabled whenever for each catalyst places in $\mathcal{C}$, either all the transitions using tokens from this catalyst place are not enabled or there is at least one transition using a token from a catalytic place. We write $m\,[U\rangle_{\mathcal{C}}$ to denote that U is catalyticly enabled, and we denote by $m\,[U\rangle_{\mathcal{C}}\, m'$ the firing of a catalyticly enabled step, where U is a catalytic step. A catalytic firing sequence is a step firing sequence where each step is a catalytic one. The set of reachable markings is defined accordingly, and denoted by $\mathfrak{M}_{\mathcal{C}}(N)$. In the same ways are defined the sets $\mathfrak{F}_{\mathcal{C}}(N)$, $\mathfrak{F}^{\mathfrak{S}}_{\mathcal{C}}(N)$ and $\mathfrak{F}^{\#^{\mathfrak{S}}}_{\mathcal{C}}(N)$.

Catalytic P systems and catalytic nets The results of Section 4 can be also used for catalytic P systems and catalytic nets. In particular, we can state the following two results.

Proposition 4. *Let $\Pi = (O, O_C, \mu, w^0, R, O')$ be a catalytic P system, and $\mathcal{F}(\Pi)$ be its associated structure. Then $\mathcal{F}(\Pi)$ is a catalytic net.*

Proposition 5. *Let $N = ((S, T, F, m_0, \mathcal{C}, n), \mathfrak{S})$ be a catalytic Petri net. Then $\mathcal{K}(N)$ is a flat catalytic P system.*

Turing completeness under the catalytic firing strategy We denote by $CPN(n)$ the class of catalytic Petri nets with n catalytic places, and by $PCPN(n)$ the class of purely catalytic Petri nets with n catalytic places. Then Theorems 1 and 2 can be specialized as follows:

Theorem 7.

$$\{\mathfrak{F}_{\mathcal{C}}^{\#^{\mathfrak{S}}}(N) \mid N \in CPN(n)\} = RE \quad \textit{and}$$
$$\{\mathfrak{F}_{\mathcal{C}}^{\mathfrak{S}}(N) \mid N \in CPN(n)\} = PsRE, \textit{ for } n \geq 2.$$

Theorem 8.

$$\{\mathfrak{F}_{\mathcal{C}}^{\#^{\mathfrak{S}}}(N) \mid N \in PCPN(n)\} = RE \quad \textit{and}$$
$$\{\mathfrak{F}_{\mathcal{C}}^{\mathfrak{S}}(N) \mid N \in PCPN(n)\} = PsRE \textit{ for } n \geq 3.$$

6 Conclusion and future work

In this paper we have presented a new class of Petri nets called *catalytic Petri nets*. This class turns out to be Turing complete by using a suitable firing strategy. The firing of a step is done by first checking, for each catalytic place, if one of the transitions using the place is enabled, and then firing at least one of these transitions. This means that the rule requires a *minimal* parallelism which is also locally confined. Establishing a connection with the membrane systems allows to state also the minimal number of catalytic places which are needed.

In this class of Petri nets the main characteristic of the basic models is retained: the firing is still local, though the notion of catalyst changes slightly the meaning of *local* because now a confined location to be checked depends on the catalyst and not only on the transition to be executed.

Other kinds of membrane systems enjoy minimal parallelism evolution steps, and the relations with other suitable classes of Petri nets have to be established. Furthermore we want to investigate further the class of catalytic Petri nets, which seems to be an interesting and promising class.

Acknowledgments. Part of this work has been done while the first author was Visiting Professor at the Dipartimento di Matematica e Informatica of the Università di Cagliari, with a grant of the Regione Autonoma della Sardegna.

References

1. Jantzen, M., Zetzsche, G.: Labeled step sequences in Petri nets. In van Hee, K.M., Valk, R., eds.: Proceedings of the 32rd International Conference on Application and Theory of Petri Nets. LNCS 5062, Springer (2008) 270–287

2. Kosaraju, S.R.: Decidability of reachability in vector addition systems (preliminary version). In: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, ACM (1982) 267–281
3. Esparza, J., Nielsen, M.: Decidability issues for Petri nets - a survey. Bulletin of the EATCS **52** (1994) 244–262
4. Esparza, J.: Decidability and complexity of Petri net problems - an introduction. In Reisig, W., Rozenberg, G., eds.: Lectures on Petri Nets I: Basic Models - Advances in Petri Nets 1996. LNCS 1491, Springer (1996) 374–428
5. Burkhard, H.D.: On priorities of parallelism. In Salwicki, A., ed.: Proceedings of Logics of Programs and Their Applications. LNCS 148, Springer (1980) 86–97
6. Burkhard, H.D.: Ordered firing in Petri nets. Elektronische Informationsverarbeitung und Kybernetik **17** (1981) 71–86
7. Hack, M.: Decidability Questions for Petri Nets. Outstanding Dissertations in the Computer Sciences. Garland Publishing, New York (1975)
8. Dufourd, C., Finkel, A., Schnoebelen, P.: Reset nets between decidability and undecidability. In Larsen, K.G., Skyum, S., Winskel, G., eds.: Proceedings of the 25th International Colloquium on Automata, Languages and Programming. LNCS 1443, Springer (1998) 103–115
9. Finkel, A., Geeraerts, G., Raskin, J.F., Begin, L.V.: On the *omega*-language expressive power of extended Petri nets. Theor. Comput. Sci. **356** (2006) 374–386
10. Geeraerts, G.: On the expressive power of Petri nets with transfer arcs vs. Petri nets with reset arcs. Technical Report 572, Université Libre de Bruxelles (2007)
11. Ciardo, G.: Petri nets with marking-dependent arc cardinality: Properties and analysis. In Valette, R., ed.: Proceeding of the 15th International Conference on Application and Theory of Petri Nets. LNCS 815, Springer (1994) 179–198
12. Păun, G., Rozernberg, G., Salomaa, A.: The Oxford Handbook of Membrane Computing. Oxford University Press (2010)
13. Păun, G.: Membrane Computing. An Introduction. Natural Computing Series. Springer (2002)
14. Ciobanu, G., Pérez-Jiménez, M.J., Păun, G., eds.: Applications of Membrane Computing. Natural Computing Series. Springer (2006)
15. Alberts, B., Johnson, A., Walter, P., Lewis, J.: Molecular Biology of the Cell. Garland Publishing, New York (2002)
16. Agrigoroaiei, O., Ciobanu, G.: Flattening the transition p systems with dissolution. In Gheorghe, M., Hinze, T., Păun, G., Rozenberg, G., Salomaa, A., eds.: 11th Int. Conf. on Membrane Computing. LNCS 6501, Springer (2010) 53–64
17. Ciobanu, G., Pan, L., Păun, G., Pérez-Jiménez, M.J.: P systems with minimal parallelism. Theoretical Computer Science **378** (2007) 117–130
18. Freund, R., Kari, L., Oswald, M., Sosík, P.: Computationally universal p systems without priorities: two catalysts are sufficient. Theoretical Computer Science **330** (2005) 251–266
19. Kleijn, J., Koutny, M., Rozenberg, G.: Towards a Petri net semantics for membrane systems. In: Proceedings of the 6th International Workshop on Membrane Computing. LNCS 3850, Springer (2005) 292–309
20. Kleijn, J., Koutny, M.: Petri nets and membrane computing. In Păun, G., Rozenberg, G., Salomaa, A., eds.: The Oxford Handbook of Membrane Computing, Oxford University Press (2010) 389–412
21. Montanari, U., Rossi, F.: Contextual nets. Acta Informatica **32** (1995)