Unbounded Petri Net Synthesis

Philippe Darondeau

IRISA, campus de Beaulieu, F35042 Rennes Cedex

Abstract. We address the problem of deciding uniformly for graphs or languages of a given class whether they are generated by unlabelled Place-Transition nets whose sets of reachable markings may be infinite.

1 Introduction

Initialized Petri nets may be seen alternatively as graph generators or as language generators. In the first case, the generated graph is the reachable state graph of the net, considered up to isomorphisms of graphs (*i.e.* any set in bijection with the set of reachable markings may be used equivalently to represent the vertices of this graph). In the second case, the generated language is the set of firing sequences of the net (we will not introduce in this paper any labelling of transitions nor any special subset of accepting states or markings). The Petri net synthesis problem consists in deciding uniformly for a fixed class of graphs or languages whether a given member of this class has a Petri net generator and in producing such a generator if it exists. For classes of graphs or languages where the decision is not possible, a connected problem is to produce from a given object a Petri net generator which approximates it at best.

The Petri net synthesis problem may be addressed for several classes of nets, including notably the Elementary Nets and the Place-Transition Nets. Synthesis was dealt with originally by Ehrenfeucht and Rozenberg in the context of finite graphs and *Elementary Nets* [22] [23]. As the number of (simple) elementary nets with a fixed set of transitions is finite, the decision problem has an obvious solution in this context. The goal of the cited authors was to put forward a graph theoretic and axiomatic solution. The seminal idea which they introduced for this purpose is the concept of *regions* of a graph. The regions of a graph are particular subsets of vertices. The regions of a graph edge-labelled on T correspond bijectively with the simulations of this graph by elementary nets of the *atomic* form $(\{p\}, T, F, M_0)$. A finite and reachable rooted graph (loopfree, deterministic and simple) is simulated by an elementary net if all walks in the graph are matched by similar firing sequences of the net, such that two walks ending at the same vertex are always simulated by firing sequences ending at the same marking. Each simulation induces thus a (unique) map from vertices to markings. The regions of the graph are the inverse images of the marking $p = 1$ under arbitrary simulations of the graph by atomic nets $(\{p\}, T, F, M_0)$. Ehrenfeucht and Rozenberg gave a purely graph theoretic characterization of these regions. Their logical structure was studied further in [11] and [12].

J. Desel, W. Reisig, and G. Rozenberg (Eds.): ACPN 2003, LNCS 3098, pp. 413–438, 2004.

Given any graph edge-labelled on T, one may *synthesize* an elementary net from this graph by gluing together on their common transitions all the simulating atomic nets $(\{p_l\}, T, F_l, M_{l,0})$. A graph is generated by some elementary net *if and only if* it is generated by the elementary net constructed in this way. It follows that the family of graphs with elementary net generators may be characterized by two axioms: *i)* for any two distinct vertices v and v', some region contains either v or v' (but not both); *ii)* for any $t \in T$ and for any vertex v, if no edge labelled with t leaves the vertex v, then the vertex v is outside some region that contains all sources of edges labelled with t and none of their targets.

Synthesis algorithms based on the above characterization were proposed in [20], [11], and [17]. In the context of Elementary Nets, synthesis is an NP-complete problem [3]. Efficient heuristic algorithms have been implemented in the tool Petrify, with application to Asynchronous Circuit Design [16]. On the side of theory, a categorical version of the correspondence between Elementary Graphs (finite and reachable rooted graphs, loopfree, deterministic and simple, satisfying axioms *(i)* and *(ii)*) and Elementary Nets was given in [38]. The latter work sheds additional light on synthesis: it indicates that morphisms of nets may also be synthesized from morphisms of graphs (*i.e.* net synthesis is functorial). For more on the synthesis of Elementary Nets, we refer the reader to [22] [23], to the papers mentionned above, and to the survey [6]. A closely related topic is the synthesis of labelled one-safe nets from Asynchronous Transition Systems, which was explored in [44] and [8].

The concept of regions, which was introduced in the context of Elementary Nets, was quickly adapted to *Place-Transition Nets*. In this different context, the regions of a graph edge-labelled on T are in bijective correspondence with the simulations of this graph by P/T-nets of the *atomic* form $(\{p\}, T, F, M_0)$. A rooted, reachable and deterministic graph is simulated by a P/T-net if all walks in the graph are matched by similar firing sequences of the net, such that two walks ending at the same vertex are simulated by firing sequences ending at the same marking. Each simulation of a graph by an atomic P/T-net $(\{p\}, T, F, M_0)$ induces thus a (unique) map from vertices to non-negative integers. Regions are no longer subsets of vertices. They are multisets of vertices: a region assigns to each vertex v the weight defined by this induced map. Multiset regions may still be given a graph-theoretic characterization, but their logical structure is unclear. This is compensated for by nice algebraic properties: the (multiset) sum of two regions is a region, and the (multiset) difference of two regions, when it is defined, is also a region. We shall intensively exploit this linear algebraic structure in the body of the paper.

Regions as multisets were introduced independently by several groups of researchers. Slightly different definitions of regions were given, depending on the amount of concurrency embedded in the classes of labelled graphs considered. Concurrency by steps was considered in [33] and [37] (the first two papers in which multiset regions were defined), and regions served there to characterize respectively the subclass of Local Trace Languages with P/T-net generators and the subclass of Step Transition Systems with P/T-net generators. Another form of concurrency (pairwise independence, which is weaker than step independence) was considered in [21], where regions served to characterize the subclass of Automata with Concurrency Relations that may be generated from P/T-nets. Multiset regions of ordinary graphs, *i.e.* graphs without concurrency, were defined in [9], where they served to characterize the subclass of finite graphs with P/T-net generators. All characterizations are expressed by two axioms akin to Ehrenfeucht and Rozenberg's axioms *(i)* and *(ii)*. The adaptation of the axiom *(i)* is immediate: a multiset region separates two vertices v and v' if they have different weights in this region. The adaptation of the axiom *(ii)* is not so immediate and it depends on the exact definition of regions that is used. Roughly, the modified axiom requires from any vertex v that, if no edge leaving v bears the label t, then some region assigns to the vertex v a weight strictly lower than the weights of all sources of edges labelled with t.

While the accent was set on categorical correspondences between graphs and nets in [37] and [21], it was set in [9] on the algebraic and combinatorial properties of regions. It was shown in the latter reference that the *minimal* regions of a finite graph provide all the information needed to determine whether this graph has a P/T-net generator. The decision problem was however left unsolved. It was actually shown in [2] that the synthesis problem is decidable for pure and bounded P/T-nets and for finite graphs or for regular languages. The main principle for the decision is to compute a *finite* set of regions that generate all other regions (the regions of a finite graph form a module, and the bounded regions of a regular language do the same). The synthesis algorithm is polynomial in the size of the graph, and it is exponential in the size of the regular expression. This algorithm was extended in [5] to bounded P/T-nets (which may be impure) and to finite Step Transition Systems. The synthesis algorithm for bounded P/T-nets has been implemented in the tool SYNET, with tentative applications to the distributed realization of protocols [4]. Another application of the bounded P/T-net synthesis is the computation of Petri net supervisory controllers [27]. All state-avoidance problems in one-safe Petri nets may in fact be solved in this way. For more on the synthesis of bounded P/T-nets and applications, we refer the reader to the above mentioned papers or to the survey [6].

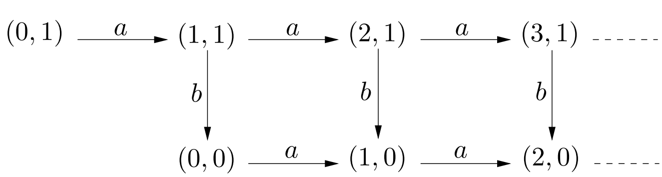
The remaining sections of the paper are devoted to the algorithmic synthesis of unbounded P/T-nets, a topic which was not covered in [6] because it was studied afterwards. Section 2 deals with the synthesis of unbounded P/T-nets from languages. Section 3 deals with the synthesis of unbounded P/T-nets from infinite graphs. The last section summarizes the results and indicates some directions for future work. The paper is self-contained. No familiarity with the synthesis of Elementary Nets nor with the synthesis of bounded Place-Transition Nets is assumed. The presentation of net synthesis given here is simpler than the general presentation given in [6], but it ignores most of the results reported there. The presentation below is based on the work of the author and his colleagues from IRISA (see references in the bibliography).

2 Net Synthesis from Languages

Let us recall the definition of Place-Transition nets (or P/T-nets for short).

Definition 1 (P/T-nets). *A P/T-net is a triple $N = (P, T, F)$ where P and T are finite disjoint sets of* places *and* transitions*, respectively, and F is a function, $F : (P \times T) \cup (T \times P) \to \mathbb{N}$. A* marking *of N is a map $M : P \to \mathbb{N}$. The* state graph *of N is a labelled graph, with markings as vertices, where there is an edge labelled with transition t from M to M' (in notation: $M[t\rangle M'$) if and only if, for every place $p \in P$, $M(p) \geq F(p, t)$ and $M'(p) = M(p) - F(p, t) + F(t, p)$. The* reachable state graph *of an* initialized *P/T-net $\mathcal{N} = (P, T, F, M_0)$, with* initial marking *M_0, is the restriction of its state graph induced by the subset of vertices that may be reached inductively from M_0. The net $\mathcal{N}$ is* unbounded *if its reachable state graph is infinite. The* language *of an* initialized *P/T-net is the set of sequences $w \in T^*$ that label walks from the root M_0 of this graph. Thus, the language of $\mathcal{N}$ is the set $\{w \in T^* \mid M_0[w\rangle\}$ where $M_0[w\rangle$ means that the sequence w may be fired inductively from M_0.*

Example 1. Let $P = \{1, 2\}$ with $M_0(1) = 0$ and $M_0(2) = 1$. Let $T = \{a, b\}$ with $F(a, 1) = 1 = F(1, b)$ and $F(2, b) = 1$. Let F evaluate to 0 for all the remaining arguments. The reachable state graph of the specified net is the infinite graph shown below. The language of this net is the regular language $a^* + aa^*ba^*$. □



The P/T-net synthesis problem for a class of languages is the problem whether one can decide *uniformly* from any language $\mathcal{L}$ in this class whether it coincides with the language of some (initialized) P/T-net, and construct such a net when it exists. Uniformity means that the same constructive procedure should apply to all languages in the considered class. For instance, the P/T-net synthesis problem has a (positive) solution for $\mathcal{L} = a^* + aa^*ba^*$ in the singleton class $\{\mathcal{L}\}$, but this does not mean that the P/T-net synthesis problem has a solution for $\mathcal{L} = a^* + aa^*ba^*$ in the class of regular languages over two letters a and b.

We propose in this section a uniform procedure that computes, for any class of semi-linear languages closed under right quotients with letters, the least over-approximation of a language in the class by the language of a P/T-net. We propose moreover a uniform procedure that solves the P/T-net synthesis problem for classes of semi-linear languages closed under right quotients with letters and under the *max* operation (w.r.t. the order prefix). We show that the synthesis problem is decidable for the regular or deterministic context-free languages, whereas it is undecidable for the context-free languages and for the languages of High-level Message Sequence Charts (or HMSCs for short). We argue finally about the practical relevance of approximating languages by P/T-net languages.

Before we describe the common principles under the two procedures, let us add two remarks about example 1. First, infinitely many different P/T-nets have the language $a^* + aa^*ba^*$. Therefore, one cannot require from an effective synthesis procedure to produce all of them. Second, any P/T-net with the language $a^* + aa^*ba^*$ has an infinite number of reachable markings. To see this, assume the opposite. Then the transition a should act as the identity on the markings of some initialized P/T-net $(P, \{a, b\}, F, M_0)$ generating this language. As $M_0[ab\rangle$, necessarily $M_0[b\rangle$, a contradiction. This remark shows that the synthesis of *unbounded* P/T-nets is a relevant problem for all reasonable classes of languages.

Henceforth in this section $T = \{t_1, \ldots, t_n\}$ is a fixed alphabet of transitions, and P/T-nets have always the set of transitions T. The languages $\mathcal{L}$ under consideration are always subsets of T^*. As we are mainly interested in languages of P/T-nets and these languages are non-empty and prefix-closed, it will always be assumed that $\mathcal{L}$ is non-empty and prefix-closed, *i.e.* $(\forall w \in \mathcal{L})\ w = u{\cdot}v \Longrightarrow u \in \mathcal{L}$ where $\cdot$ denotes the concatenation product in T^*. In particular, the empty word ε is always in $\mathcal{L}$. In the sequel, initialized P/T-nets are called P/T-nets for short. The language of the P/T-net $\mathcal{N}$ is denoted $L(\mathcal{N})$.

2.1 The Regions of a Language

The two essential facts on which is based the synthesis of P/T-nets from languages are stated in the (almost obvious) propositions 1 and 2 below.

Definition 2 (Atomic subnets). *A P/T-net $\mathcal{N} = (P, T, F, M_0)$ is a* subnet *of $\mathcal{N}' = (P', T, F', M_0')$ if $P \subseteq P'$ and F and M_0 are the induced restrictions of F' and M_0' (respectively on $(P \times T) \cup (T \times P)$ and on P). The net (P, T, F, M_0) is* atomic *if $|P| = 1$. An* atomic subnet *of $\mathcal{N}'$ is a subnet of $\mathcal{N}'$ which is atomic.*

Proposition 1. *The language of a P/T-net is the intersection of the languages of its atomic subnets.*

Definition 3 (P/T-regions). *Given a word $w \in T^*$, an atomic P/T-net $\mathcal{N} = (\{p\}, T, F, M_0)$ is a* P/T-region *of w if $w \in L(\mathcal{N})$. Given a language $\mathcal{L}$, an atomic P/T-net $\mathcal{N}$ is a P/T-region of $\mathcal{L}$ if it is a P/T-region of every word $w \in \mathcal{L}$.*

Proposition 2. *$\mathcal{L}$ is the language of a P/T-net if and only if the set of P/T-regions of $\mathcal{L}$ contains a finite subset $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$ such that, for every $t \in T$ and for every $w \in \mathcal{L}$, if $w \cdot t \notin \mathcal{L}$, then some $\mathcal{N}_l$* is not *a P/T-region of $w \cdot t$. When this condition is satisfied, $\mathcal{L} = L(\mathcal{N})$ where $\mathcal{N}$ is the P/T-net with the set of atomic subnets $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$.*

Example 2. The P/T-net described in the example 1 has two atomic subnets $\mathcal{N}_1$ and $\mathcal{N}_2$, where 1 is the unique place of $\mathcal{N}_1$ and 2 is the unique place of $\mathcal{N}_2$. Both nets are regions of the language $a^* + aa^*ba^*$. The word $\varepsilon \cdot b$ is not in this language, but it does not belong either to $L(\mathcal{N}_1)$, thus $\mathcal{N}_1$ is not a region of this word. Similarly, $\mathcal{N}_2$ is not a region of any word $a^k ba^l \cdot b$ for $k > 0$ and $l \geq 0$. Finally observe that the language $a^* + aa^*ba^*$ has an infinite set of regions. For instance, all the atomic P/T-nets $(\{p\}, \{a, b\}, F, M_0)$ such that $F(p, a) = 0$ and $F(p, b) = 0$ are regions of this language. □

In view of proposition 2, the feasability of a procedure for the decision of the P/T-net synthesis problem (with respect to a fixed class of languages) depends on the feasability of two subproblems. First, one should compute an effective representation of the set of regions of a language, notwithstanding the fact that this set is always *infinite*. Second, one should decide whether some *finite* subset of regions suffices to *reject* all minimal words (with respect to the order prefix) in the complement of the given language, even though these *unwanted* words may form an *infinite* set (*e.g.*, the set $b + aa^*bb$ in example 1).

The above problems cannot be solved without specific assumptions on the considered classes of languages. Fortunately, the first problem has an easy solution for semi-linear languages.

2.2 A Procedure for Computing Generating Regions

Let us recall two definitions.

Definition 4 (Commutative image). *The* commutative image *of a word $w \in T^*$ is the n-vector $[w]$ whose respective entries $[w]_i$ count for each $i \in [1, n]$ the occurrences of the letter t_i in w. The commutative image of a language $\mathcal{L} \subseteq T^*$ is the set $[\mathcal{L}] = \{[w] \mid w \in \mathcal{L}\}$.*

Definition 5 (Semi-linear subset). *Let $\mathcal{M} = (\mathcal{M}, \cdot, 1)$ be a monoid. A subset of $\mathcal{M}$ is* linear *if it may be expressed as $m \cdot \mathcal{F}^*$ where $m \in \mathcal{M}$, $\mathcal{F}$ is a finite subset of $\mathcal{M}$, and $\mathcal{F}^*$ is the least submonoid of $\mathcal{M}$ containing $\mathcal{F}$. A finite union of linear subsets of $\mathcal{M}$ is called a* semi-linear *subset.*

A language $\mathcal{L}$ is said to be *semi-linear* if its commutative image $[\mathcal{L}]$ is a semi-linear subset of $\mathbb{N}^n$, the commutative monoid where the product $\cdot$ is the addition of n-vectors and where the neutral element 1 is the all-zeroes n-vector.

Example 3. For $\mathcal{L} = a^* + aa^*ba^*$, where we let $a = t_1$ and $b = t_2$ for convenience, $[\mathcal{L}] = < 1, 0 >^* + < 1, 0 > \cdot < 1, 0 >^* \cdot < 0, 1 > \cdot < 1, 0 >^*$. By commutativity of the product (*i.e.* addition) in $\mathbb{N}^n$, $[\mathcal{L}] = < 1, 0 >^* + < 1, 1 > \cdot < 1, 0 >^*$, hence this set is semi-linear (in regular expressions, + denotes set union). □

The considerations in the above example may be generalized to all regular expressions. It should therefore be clear that for any language $\mathcal{L}$, $[\mathcal{L}]$ is semi-linear if and only if $[\mathcal{L}] = [\mathcal{R}]$ for some regular language $\mathcal{R}$. A celebrated theorem by Parikh shows that this condition holds for the context-free languages (see section 6.9 in [31] for the construction of $\mathcal{R}$ from a context-free grammar generating $\mathcal{L}$).

In order to achieve our goals, we shall actually require a little more than the semi-linearity of $[\mathcal{L}]$. Namely, we require that all right derivatives $\mathcal{L}/t_j$ are semi-linear, where $t_j \in T$ and $\mathcal{L}/t_j = \{v \in T^* \,|\, v \cdot t_j \in \mathcal{L}\}$. Under this stronger requirement (which is met by context-free languages), one can effectively compute a finite representation of the infinite set of P/T-regions of $\mathcal{L}$. Moreover, this representation yields for free a P/T-net $\mathcal{N}$ whose language $L(\mathcal{N})$ is the *least* net language larger than $\mathcal{L}$. The construction is explained in the rest of the section.

Recall that a P/T-region of $\mathcal{L}$ is an atomic P/T-net $\mathcal{N} = (\{p\}, T, F, M_0)$ such that $\mathcal{L} \subseteq L(\mathcal{N})$. An atomic P/T-net $\mathcal{N}$ as above may be represented equivalently as a $(2n+1)$-vector $< M_0(p), F(p,t_1), \ldots, F(p,t_n), F(t_1,p), \ldots, F(t_n,p) >$. We claim that a $(2n+1)$-vector $\boldsymbol{x} =< x_0, x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n} >$ defines a region of $\mathcal{L}$ if and only if all its entries x_k are non-negative integers, and for each (non empty) word $v \cdot t_j$ in $\mathcal{L}$

$$x_0 + \sum_{i=1}^{n} [v]_i \times (x_{(n+i)} - x_i) \geq x_j \tag{1}$$

Actually, if the vector $\boldsymbol{x}$ is seen as an atomic P/T-net, the above inequality may be read as $M[t_j\rangle$ where M is the marking of the net reached after firing the sequence of transitions v, *assuming that v may be fired.* Since $\mathcal{L}$ is prefix-closed, this will necessarily be the case if similar inequalities hold for all the non-empty prefixes $u \cdot t_k$ of v. Let us now use the assumption that all derivatives $\mathcal{L}/t_j$ are semi-linear. Thus, for each $t_j \in T$, the set $[\mathcal{L}/t_j]$ is a finite union of linear sets $\boldsymbol{e} \cdot \mathcal{F}^*$, where $\boldsymbol{e} \in \mathbb{N}^n$ and $\mathcal{F}$ is a finite subset of $\mathbb{N}^n$. For each $t_j \in T$ and for each linear set $\boldsymbol{e} \cdot \mathcal{F}^*$ in $[\mathcal{L}/t_j]$, the collection of instances of 1 generated from words $v \in \mathcal{L}/t_j$ such that $[v] \in \boldsymbol{e} \cdot \mathcal{F}^*$ may be replaced equivalently with the *finite* linear system:

$$\sum_{i=1}^{n} \boldsymbol{e}[i] \times (x_{(n+i)} - x_i) \geq x_j - x_0 \tag{2}$$

$$\sum_{i=1}^{n} \boldsymbol{f}[i] \times (x_{(n+i)} - x_i) \geq 0 \tag{3}$$

where f ranges over the finite set $\mathcal{F}$. Let us justify this claim. For any vector $\boldsymbol{x}$ which is a solution of the finite linear system, the inequality 1 is obviously satisfied for all $v \in \mathcal{L}/t_j$ such that $[v] \in \boldsymbol{e} \cdot \mathcal{F}^*$. Conversely, the conjunction of all such inequalities entails 2 and 3. To see that it entails 3, suppose for a contradiction that 3 does not hold for some $\boldsymbol{f} \in F$. Then, for h large enough, $\sum_{i=1}^{n} (\,(\boldsymbol{e}+h\boldsymbol{f})\,[i]\,) \times (x_{(n+i)} - x_i) < x_j - x_0$. As $[\boldsymbol{e}+h\boldsymbol{f}] = [v]$ for some $v \in \mathcal{L}/t_j$ and the inequality 1 cannot hold for the considered v, a contradiction has been reached. Therefore, the set of P/T-regions of $\mathcal{L}$, seen as vectors $\boldsymbol{x} \in \mathbb{N}^{2n+1}$, is the set of solutions of a finite system of linear inequalities (T is finite, and each set $[\mathcal{L}/t_j]$ is a finite union of linear subsets). Moreover, all inequalities in this system are *homogeneous*, *i.e.* they may be written equivalently in the form $\sum_{k=0}^{2n} \alpha_k \times x_k \geq 0$ (where the α_k are constants in $\mathbb{Z}$).

Example 4. For $\mathcal{L} = a^* + aa^*ba^*$, one obtains $\mathcal{L}/a = \mathcal{L}$ and $\mathcal{L}/b = aa^*$. Therefore if we let $a = t_1$ and $b = t_2$, $[\mathcal{L}/t_1] =< 1,0 >^* + < 1,1 > \cdot < 1,0 >^*$ and $[\mathcal{L}/t_2] =< 1,0 > \cdot < 1,0 >^*$. The P/T-regions $\mathcal{L}$ are the solutions of the system

$$0 \geq x_1 - x_0$$
$$(x_3 - x_1) + (x_4 - x_2) \geq x_1 - x_0$$
$$(x_3 - x_1) \geq x_2 - x_0$$
$$(x_3 - x_1) \geq 0$$

The atomic net $\mathcal{N}_1 = (\{1\}, T, F, M_0)$ given by $x_0 = M_0(1) = 0$, $x_1 = F(1,a) = 0$, $x_2 = F(1,b) = 1$, $x_3 = F(a,1) = 1$, and $x_4 = F(b,1) = 0$ is a P/T-region of $\mathcal{L}$, and similarly is the atomic net $\mathcal{N}_2 = (\{2\}, T, F, M_0)$ given by $x_0 = M_0(2) = 1$, $x_1 = F(2,a) = 0$, $x_2 = F(2,b) = 1$, $x_3 = F(a,2) = 0$, and $x_4 = F(b,2) = 0$. □

Let $\mathcal{S}$ be the finite system of linear inequalities in the variables $x_0, \ldots, x_{2n}$ which defines the regions of $\mathcal{L}$, augmented with inequalities $x_k \geq 0$ for all $k \in [0, 2n]$. If one lets the variables x_k range over the set $\mathbb{Q}$ of rational numbers, the solutions of $\mathcal{S}$ in $\mathbb{Q}^{2n+1}$ form a cone with a finite set of generators $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$ (see [41]). This means that a rational vector $\boldsymbol{x}$ is a solution of $\mathcal{S}$ if and only if $\boldsymbol{x} = \sum_{l=1}^{m} q_l \boldsymbol{x}_l$ for some non-negative rational coefficients q_l. Moreover, one can effectively compute a minimal set of generators $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$, *e.g.* using Chernikova's algorithm [15]. A finite representation of the set of P/T-regions of $\mathcal{L}$ is then obtained.

A vector $\boldsymbol{x}$ is a P/T-region of $\mathcal{L}$ if and only if $\boldsymbol{x} \in \mathbb{N}^{2n+1}$ and $\boldsymbol{x} = \sum_{l=1}^{m} q_l \, \boldsymbol{x}_l$ for some non-negative rational coefficients q_l.

Henceforth in the section, we assume that $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$ are vectors of integers (this may be assumed w.l.o.g. since $\boldsymbol{x}_l$ may be replaced equivalently with $q_l \, \boldsymbol{x}_l$ for any non-negative q_l), and we call them the *generating regions* of $\mathcal{L}$.

Example 5. For $\mathcal{L} = a^* + aa^*ba^*$, where $a = t_1$ and $b = t_2$, the generating regions are the columns of the following table (we let $\boldsymbol{x} = < m_0, ^\bullet a, ^\bullet b, a^\bullet, b^\bullet >$ for convenience):

m_0	1	0	0	1	1	0	1
$^\bullet a$	0	0	0	1	1	0	0
$^\bullet b$	0	0	0	1	0	1	1
$a^\bullet$	0	1	0	1	1	1	0
$b^\bullet$	0	0	1	1	0	0	0

The last two regions correspond to the atomic nets $\mathcal{N}_1$ and $\mathcal{N}_2$ already seen. □

Let $\mathcal{N}$ be the P/T-net formed by gluing together on transitions $t_j \in T$ the atomic P/T-nets $\mathcal{N}_1 \ldots \mathcal{N}_m$ defined by the generating regions $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$ of $\mathcal{L}$.

$L(\mathcal{N})$ is the least *net language larger than $\mathcal{L}$.*

Let us establish this claim. As each atomic net $\mathcal{N}_l$ ($l \in [1, m]$) is a region of $\mathcal{L}$, it should be clear from definition 3 and proposition 1 that $L(\mathcal{N})$ is larger than $\mathcal{L}$.

Now let $\mathcal{N}'$ be any P/T-net such that $\mathcal{L} \subseteq L(\mathcal{N}')$. Suppose for a contradiction that some word w belongs to $L(\mathcal{N}) \setminus L(\mathcal{N}')$. We may assume w.l.o.g. that w is minimal w.r.t. the order prefix among the words in that case. Then necessarily, $w = v \cdot t_j$ for some $t_j \in T$ and $v \in L(\mathcal{N}) \cap L(\mathcal{N}')$. As $v \in L(\mathcal{N}')$ and $v \cdot t_j \notin L(\mathcal{N}')$, the inequality 1 does certainly not hold for some $2n+1$-vector $\boldsymbol{x'}$ representing an atomic subnet of $\mathcal{N}'$. Since $\mathcal{L} \subseteq L(\mathcal{N}')$, this atomic subnet is a region of $\mathcal{L}$, thus $\boldsymbol{x'} = \sum_{l=1}^{m} q_l \, \boldsymbol{x}_l$ for some *non-negative* rational coefficients q_l. Therefore, the inequality 1 does not hold for some vector $\boldsymbol{x} = \boldsymbol{x}_l \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$. It follows that $v \cdot t_j \notin L(\mathcal{N}_l)$, and hence $v \cdot t_j \notin L(\mathcal{N})$, a contradiction.

The above construction may be applied to any class of languages with semi-linear right derivatives. This is the case of every semi-linear full TRIO (by definition, a full TRIO is closed under homomorphisms, inverse homomorphisms, and intersection with regular languages). Examples are the regular languages, the context-free languages, the simple matrix languages of fixed degree [35], the languages of flip-pushdown automata with a fixed number of reversals [32], and the full slip AFLs described in [30]. This is also the case of two other classes of languages generated by parallel systems, namely the languages of parallel communicating grammar systems with terminal transmission and with fully synchronized mode [25], and the languages of HMSCs [14].

2.3 A Procedure for the Decision of the Net Synthesis Problem

Deciding whether a given language $\mathcal{L}$ has a P/T-net generator amounts to deciding whether $\mathcal{L} = L(\mathcal{N})$ where $\mathcal{N}$ is the net constructed from the generating regions of $\mathcal{L}$. We propose now a decision procedure that works under additional requirements of semi-linearity on the considered class of languages. Namely, we require that the complements in $\mathcal{L}$ of the right derivatives are also semi-linear. This requirement is significant: the assumption that all sets $[\mathcal{L}/t_j]$ are semi-linear does *not* entail that all sets $[\mathcal{L} \setminus (\mathcal{L}/t_j)]$ are semi-linear (although $[\mathcal{L}]$ must be semi-linear in this case).

For convenience of notation, let $\mathcal{L} \ominus t_j = \mathcal{L} \setminus (\mathcal{L}/t_j)$, thus $\mathcal{L} \ominus t_j$ is the set of the words $v \in \mathcal{L}$ such that $v \cdot t_j \notin \mathcal{L}$. Clearly, $\mathcal{L} = L(\mathcal{N})$ if and only if $v \cdot t_j$ is *not* in $L(\mathcal{N})$ whenever $t_j \in T$ and $v \in (\mathcal{L} \ominus t_j)$. Seeing that $\mathcal{N}$ was built up from the atomic nets defined by the generating regions of $\mathcal{L}$, $v \cdot t_j$ is not in $L(\mathcal{N})$ if and only if

$$x_0 + \sum_{i=1}^{n} [v]_i \times (x_{(n+i)} - x_i) < x_j \tag{4}$$

for some generating region $\boldsymbol{x} \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$. Let $\boldsymbol{y} = [v]$ thus $\boldsymbol{y} \in [\mathcal{L} \ominus t_j]$, and let $\boldsymbol{y} = < y_1, \ldots, y_n >$, then relation 4 may be rewritten to the linear inequality

$$x_0 + \sum_{i=1}^{n} y_i \times (x_{(n+i)} - x_i) < x_j \tag{5}$$

When the x_k are fixed constants in $\mathbb{N}$ (for $k \in [0, 2n]$) and the y_i are variables in $\mathbb{N}$ (for $i \in [1, n]$), the formula 5 amounts to a Presburger formula (it may be expressed equivalently as a comparison between two sums), hence it defines an effective semi-linear subset of $\mathbb{N}^n$ [29]. For each $l \in [1, m]$ and for each $j \in [1, n]$, let $Y_{l,j}$ be the semi-linear subset of $\mathbb{N}^n$ which is defined with formula 5 for the constants $x_k = \boldsymbol{x_l}[k]$ (*i.e.* for $\boldsymbol{x} = \boldsymbol{x_l}$). Now, $\mathcal{L} = L(\mathcal{N})$ if and only if, for all t_j,

$$[\mathcal{L} \ominus t_j] \subseteq \cup_{l=1}^{m} Y_{l,j} \tag{6}$$

As we assumed that all sets $[\mathcal{L} \ominus t_j]$ are semi-linear, and the semi-linear subsets of $\mathbb{N}^n$ form an effective boolean algebra [28], one can compute $[\mathcal{L} \ominus t_j] \setminus \cup_{l=1}^{m} Y_{l,j}$ and decide whether this set is empty. Therefore, one can decide whether $\mathcal{L} = L(\mathcal{N})$. We have thus obtained a decision procedure for the P/T-net synthesis problem.

Assuming that all sets $[\mathcal{L}/t_j]$ and $[\mathcal{L} \setminus (\mathcal{L}/t_j)]$ are semi-linear, one can decide whether the language $\mathcal{L}$ has a P/T-net generator.

When the decision is successful, it may occur that $\mathcal{L} = L(\mathcal{N}')$ for some proper subnet $\mathcal{N}'$ of the net $\mathcal{N}$ constructed from all the generating regions of $\mathcal{L}$. The procedure may be adapted in order to produce directly some minimal subnet $\mathcal{N}'$ of $\mathcal{N}$ such that $\mathcal{L} = L(\mathcal{N}')$. The subsets of $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ should be explored in increasing order until discovering some subset $\{\boldsymbol{x}_{l_1}, \ldots, \boldsymbol{x}_{l_p}\}$ large enough to make relation 6 valid for all $j \in [1, n]$ when l ranges over $\{l_1, \ldots, l_p\}$. The solution net $\mathcal{N}'$ is then constructed from the atomic nets defined by $\boldsymbol{x}_{l_1} \ldots \boldsymbol{x}_{l_p}$.

Example 6. For $\mathcal{L} = a^* + aa^*ba^*$, one obtains $(\mathcal{L} \ominus a) = \emptyset$ and $(\mathcal{L} \ominus b) = \varepsilon + aa^*ba^*$. Let $a = t_1$ and $b = t_2$, then $[\mathcal{L} \ominus t_2] = < 0, 0 > + < 1, 1 >< 1, 0 >^*$. For the two regions $\boldsymbol{x}_1 = < 0, 0, 1, 1, 0 >$ and $\boldsymbol{x}_2 = < 1, 0, 1, 0, 0 >$ (see example 5), the sets $Y_{1,2}$ and $Y_{2,2}$ are defined by the respective formulas $y_1 - y_2 < 1$ and $1 - y_2 < 1$. Clearly, $< 0, 0 > \in Y_{1,2}$ and for any non-negative integer h, $< 1 + h, 1 > \in Y_{2,2}$. Therefore, $\mathcal{L}$ is the language of the net formed of the two atomic P/T-nets $\mathcal{N}_1$ and $\mathcal{N}_2$ from example 4. Seeing that $< 0, 0 > \notin Y_{2,2}$ and $< 2, 1 > \notin Y_{1,2}$, this net is a minimal net generator for $\mathcal{L}$. □

It remains to show classes of languages in which our working assumptions hold, *i.e.* where $[\mathcal{L}/t_j]$ and $[\mathcal{L} \ominus t_j]$ are semi-linear for every prefix-closed language $\mathcal{L}$ and for every $t_j \in T$. For any language $\mathcal{L}$ of T^*, define

$$max(\mathcal{L}) = \{u \in \mathcal{L} \mid (\forall v \in T^*)\, u \cdot v \in \mathcal{L} \Longrightarrow v = \varepsilon\}$$

Then, for any prefix-closed language $\mathcal{L}$ of T^*, $v \in \mathcal{L}$ and $v \cdot t_j \notin \mathcal{L}$ if and only if $v \cdot t_j \in max(\mathcal{L} \cdot t_j)$. Therefore, $\mathcal{L} \ominus t_j = (max(\mathcal{L} \cdot t_j))/t_j$. It follows that the sets $[\mathcal{L} \ominus t_j]$ are semi-linear in every class of semi-linear languages with the following properties:

i) the class is closed under right products and right quotients with letters,
ii) the class is closed under *max*.
Property *(i)* holds in any full TRIO, but property *(ii)* does not! As the following example shows, it does not hold *e.g.* for the context-free languages.

Example 7. ([18]) Define context-free languages on the alphabet $\{a, b, c, d, e\}$ as follows. First, let $A = \{a^n b\, c^m \mid n \neq m\}$, $B = b\, c^*$, $C = \{c^n b\, c^m \mid n \neq m\}$. Next, let $D = a^* B^* BB\, b\, d$, $E = AB^* B\, b\, d\, e + a^* B^* b\, CB^* b\, d\, e$, and $L = D + E$. Then $max(L) = E + F$, where $F = \{a^n (b\, c^n)^m b\, d \mid n \geq 0 \wedge m \geq 2\}$. Assume that $[max(L)]$ is semi-linear. Since $[E]$ is semi-linear and $[E]$ and $[F]$ are disjoint, $[F] = [max(L)] \setminus [E]$ and this set is semi-linear, hence it may be defined by a Presburger formula. Now $[F]$ is the set of the integer vectors of the form $< n, m+1, n \times m, 1, 0 >$. As multiplication cannot be defined in Presburger arithmetic, a contradiction has been reached, hence $[max(L)]$ is not semi-linear.

We know actually only two classes of semi-linear languages with properties *(i)* and *(ii)*: the regular languages (a full TRIO) and the deterministic context-free languages (which do not form a full TRIO). The deterministic context-free languages are indeed closed under right products and quotients with regular languages (see [31]), and they are closed under the *max* operation (see [34]).

The P/T-net synthesis problem is decidable for regular or deterministic *context-free languages.*

2.4 Two Undecidable Cases

We show that it is undecidable *i)* whether an arbitrary context-free language has a P/T-net generator, and *ii)* whether the language of an arbitrary HMSC has a P/T-net generator. The proofs for the two facts are similar.

We consider first context-free languages. Given any context-free language $\mathcal{L}$ of T^*, let $\mathcal{N}$ be the P/T-net defined by the generating regions of $\mathcal{L}$ (see section 2.2). Then $\mathcal{L} \neq T^*$ *if and only if* $\mathcal{L} \neq L(\mathcal{N})$ *or* $L(\mathcal{N}) \neq T^*$. The complement of a (*deterministic*) P/T-net language may be generated by a *labelled* P/T-net with a finite subset of *final* partial markings [39]. The reachability of partial markings is decidable [36]. Therefore one can decide whether $L(\mathcal{N}) = T^*$. If one could decide whether $\mathcal{L} = L(\mathcal{N})$, one could decide whether $\mathcal{L} = T^*$. Now it is undecidable for an arbitrary context-free language $\mathcal{L}$ of T^* whether $\mathcal{L} = T^*$ (see *e.g.* [34]). Therefore, the P/T-net synthesis problem is undecidable for context-free languages.

We consider now HMSC languages. In this case, the alphabet T has a specific structure. On the one hand, it is equipped with a map $\ell : T \rightarrow [1, K]$ that assigns a specific *location* to each transition. On the other hand, the transitions in T are divided into message *emissions* (emit towards location k), message *receptions* (receive from location k), and *internal* transitions. An HMSC $\mathcal{H}_1$ with K locations and with internal transitions only may be simulated by an HMSC $\mathcal{H}_2$ with $2K$ locations and with no internal transitions, such that $L(\mathcal{H}_1)$ is a P/T-net language *if and only if* $L(\mathcal{H}_2)$ is a P/T-net language: each internal transition at location k may be simulated by an emission from k to $k+K$ plus a matching reception at $k+K$. We shall assume below, for simplicity, that all transitions in T are internal.

Let $T = T_1 \cup T_2$ where $\ell(t) = k$ for $t \in T_k$, and let $E_k = T_k \setminus \{\$_k\}$ where $\$_k$ is a distinguished symbol in T_k. Whenever a relation $\mathcal{R} \subseteq (E_1^* \times E_2^*)$ is accepted by a finite automaton over the product monoid $E_1^* \times E_2^*$, the relation $\mathcal{R} \cdot (\$_1, \$_2)$ is accepted by a finite and trim automaton over $T_1^* \times T_2^*$. This trim automaton may be seen as an HMSC over T. For any *rational* relation $\mathcal{R} \subseteq (E_1^* \times E_2^*)$, there exists therefore an HMSC $\mathcal{H}$ over T with the language

$$L(\mathcal{H}) = pref\{w \mid \exists (u,v) \in \mathcal{R} : w \in (u \cdot \$_1) \sqcup\!\sqcup (v \cdot \$_2)\}$$

where $pref$ denotes prefix closure and $\sqcup\!\sqcup$ is the shuffle operation. Let $\sqcup\!\sqcup$ be defined on languages of T^* by an additive extension of the latter. There obviously exists a P/T-net $\mathcal{N}'$ such that $L(\mathcal{N}') = pref\,((T_1^* \cdot \$_1) \sqcup\!\sqcup (T_2^* \cdot \$_2))$. It follows from the definition of $L(\mathcal{H})$ that $L(\mathcal{H}) = L(\mathcal{N}')$ *if and only if* $\mathcal{R} = (E_1^* \times E_2^*)$.

Now let $\mathcal{N}$ be the P/T-net constructed from the generating regions of $L(\mathcal{H})$. Then $\mathcal{R} \neq (E_1^* \times E_2^*)$ *if and only if* $L(\mathcal{H}) \subsetneq L(\mathcal{N})$ *or* $L(\mathcal{N}) \subsetneq L(\mathcal{N}')$. From the results in [39] and [36] one can decide whether $L(\mathcal{N}) = L(\mathcal{N}')$. If one could decide whether $L(\mathcal{H}) = L(\mathcal{N})$, one could decide whether $\mathcal{R} = (E_1^* \times E_2^*)$. Now, provided that each subalphabet E_k contains at least two letters, it is undecidable for an arbitrary rational relation $\mathcal{R}$ whether $\mathcal{R} = (E_1^* \times E_2^*)$, see [26] or [10]. Therefore, the P/T-net synthesis problem is undecidable for HMSC languages.

The P/T-net synthesis problem is undecidable for context-free languages and for HMSC languages.

2.5 Comments and Complements

Many classes of semi-linear languages extend the context-free languages, or are based on rational relations. The undecidability results presented in section 2.4 apply in both cases. In order to extend the decidability results established in section 2.3, it appears suitable to focus on sub-classes of languages generated with *deterministic* automata, as we have done for the context-free languages. This way is still open for HMSC languages, since deterministic generators have not yet been thoroughly investigated in this context.

As concerns applications, one may argue that least over-approximations of languages by P/T-nets are often more suitable than exact realizations. Two cases in support of this thesis are discussed below.

Let us come again to HMSCs. As these are intended to serve at an early stage of design of distributed systems, collections of scenarios defined by HMSCs are usually seen as *incomplete specifications* of a system. P/T-net synthesis may be used to build a *prototype* of the specified system, *i.e.* a distributed scale model that may be run and model-checked before designing software. Now model-checking is undecidable for *general* HMSCs (see [1] or [14]). Therefore, one should accept that a prototype system may have a language larger than the language of the specifying HMSC. Approximating HMSC languages by P/T-net languages as indicated in section 2.2 is justified, because the model-checking of P/T-nets w.r.t. linear-time μ-calculus is decidable [24]. Moreover, relations $L(\mathcal{N}) \subseteq \mathcal{R}$ and $\mathcal{R} \subseteq L(\mathcal{N})$ are decidable for arbitrary P/T-nets $\mathcal{N}$ and regular languages $\mathcal{R}$ (because $\mathcal{R}$ and $\complement\mathcal{R}$ have labelled P/T-net generators with final markings). A matter not yet discussed is distribution. Recall that P/T-regions may be seen as vectors $\boldsymbol{x} =< x_0, \ldots, x_{2n} >$ in $\mathbb{N}^{2n+1}$, where $T = \{t_1, \ldots, t_n\}$. By simply imposing on vectors $\boldsymbol{x}$, for some $\mathcal{I} \in \mathcal{P}([1, n])$, the additional constraint $(\exists I \in \mathcal{I})(\forall i \in I)\,(x_i = 0)$, *distributable* P/T-nets may be produced by the synthesis procedure. In a distributable P/T-net (see [4]), the transitions have locations, and an input place is never shared by transitions with different locations. Because competitions for tokens are local, distributable P/T-nets may be cut to local subnets communicating by asynchronous message passing. Distributed prototypes of HMSCs may be obtained in this way.

Another field of application is *supervisory control*. Let us briefly recall the framework defined by Ramadge and Wonham [40]. A *plant* is a finite automaton over an alphabet A with two orthogonal partitions: $A = A_c \cup A_{uc}$ where the transitions in A_c and A_{uc} are respectively *controllable* and *uncontrollable*, and $A = A_o \cup A_{uo}$ where the transitions in A_o and A_{uo} are respectively *observable* and *unobservable*. Let $\mathcal{R}_p$ be the language of the plant, and let $\mathcal{R}_l \subseteq \mathcal{R}_p$ be a regular subset of *legal* firing sequences. For the sake of simplicity, assume that $\mathcal{R}_p$ and $\mathcal{R}_l$ are prefix-closed and $A_c \subseteq A_o$ (unobservable transitions are uncontrollable). A *controller* is then a (finite or infinite) automaton that defines a prefix-closed language $\mathcal{K}$ of A_o^*. The problem is to search for some $\mathcal{K}$ in a given class of languages such that $\{u \in \mathcal{R}_p \,|\, \pi_o(u) \in \mathcal{K}\} \subseteq \mathcal{R}_l$ where π_o projects A^* on A_o^*. An *admissible* controller $\mathcal{K}$ should moreover satisfy

$$\begin{gathered} \forall t \in (A_o \cap A_{uc}) \quad \forall u \in \mathcal{R}_p \quad \forall v \in \mathcal{K} \\ v = \pi_o(u) \ \wedge\ (u \cdot t) \in \mathcal{R}_p \quad \Longrightarrow \quad (v \cdot t) \in \mathcal{K} \end{gathered}$$

Deciding whether *maximally permissive* admissible controllers exist reduces to deciding whether for some $\mathcal{K}$,

$$\smile \ \cap\ \pi_o(\mathcal{R}_l) \ \subseteq\ \mathcal{K}\ \subseteq\ \smile$$

where $\smile$ is the largest subset of A_o^* containing no observation sequence $v = \pi_o(u)$ such that $uw \in \mathcal{R}_p$ and $uw \notin \mathcal{R}_l$ for some uncontrollable sequence $w \in A_{uc}^*$. Thus, if $\cdot/\cdot$ denotes quotient of languages, $\smile$ is the complement in A_o^* of the set

$$\frown = \pi_o\left(\,(\mathcal{R}_p \cap \complement\,\mathcal{R}_l)\,/A_{uc}^*\,\right)$$

As $\frown$ is a regular set, $\smile$ is regular, hence $\mathcal{L} = \smile\ \cap\ \pi_o(\mathcal{R}_l)$ is regular. The problem amounts to deciding whether there exists some $\mathcal{K}$ in the specified class of languages such that $\mathcal{L} \subseteq \mathcal{K} \subseteq \smile$ where $\mathcal{L}$ and $\smile$ are two regular languages. This problem may be posed w.r.t. the class of P/T-net languages. The solution is to compute $\mathcal{N}$ from $\mathcal{L}$ as shown in section 2.2, such that $\mathcal{K} = L(\mathcal{N})$ is the least P/T-net language larger than $\mathcal{L}$, and then to check whether $L(\mathcal{N}) \subseteq \smile$ (this is decidable). Maximally permissive P/T-net controllers are then obtained.

3 Net Synthesis from Infinite Graphs

The P/T-net synthesis problem for a class of graphs is the problem whether one can decide uniformly from any graph in this class whether it is isomorphic to the reachable state graph of some initialized P/T-net, and construct such a net when it exists. In this section, $T = \{t_1, \ldots, t_n\}$ is a fixed alphabet, all P/T-nets have the set of transitions T, and all graphs have directed edges with labels in T.

Let $\mathcal{G} = (V, E, v_0)$ denote a graph with respective sets of *vertices* and *edges* V and E, where $v_0 \in V$ is the *root* and $E \subseteq (V \times T \times V)$. An edge $(v, t, v') \in E$ has a *source* v, a *label* t, and a *target* v'. We consider deterministic and reachable graphs exclusively, *i.e.* we assume that every vertex v can be reached by some walk from v_0 to v, and that distinct edges with a common source have distinct labels. A *morphism* of graphs $\sigma : \mathcal{G} \to \mathcal{G}'$, where $\mathcal{G}' = (V', E', v'_0)$, is a map $\sigma : V \to V'$ such that $\sigma(v_0) = \sigma(v'_0)$ and $(\sigma(v), t, \sigma(v')) \in E'$ for every edge $(v, t, v') \in E$. Note that there is *at most one* morphism from $\mathcal{G}$ to $\mathcal{G}'$. Let $\mathcal{G} \leq \mathcal{G}'$ when this morphism exists. It is easily seen that $\leq$ is an order relation and that two graphs $\mathcal{G}$ and $\mathcal{G}'$ are isomorphic ($\mathcal{G} \cong \mathcal{G}'$) *if and only if* $\mathcal{G} \leq \mathcal{G}'$ and $\mathcal{G}' \leq \mathcal{G}$. Let $G(\mathcal{N})$ denote the reachable state graph of the P/T-net $\mathcal{N}$. The problem is to decide from a given graph $\mathcal{G}$ whether $\mathcal{G} \cong G(\mathcal{N})$ for some P/T-net $\mathcal{N}$.

This problem is a strengthening of the problem dealt with in section 2. Indeed, $\mathcal{G} \cong G(\mathcal{N}) \implies L(\mathcal{G}) = L(\mathcal{N})$ where $L(\mathcal{G})$ is the set of sequences in T^* labelling walks from v_0 to arbitrary vertices v in $\mathcal{G}$. The converse implication does not hold. We show in this section that the P/T-net synthesis problem for graphs may be solved by a modification of the techniques already presented. The leading idea is to replace the relation of language inclusion $\subseteq$ used in section 2 with the order relation $\leq$ on graphs. The development given hereafter mimics the development given in this earlier section.

3.1 The Regions of a Graph

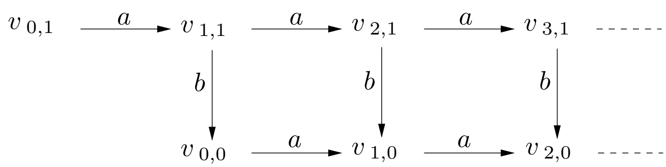
To begin with, let us observe that any finite family of graphs $\mathcal{G}_l = (V_l, E_l, v_{l,0})$, $l \in [1, m]$, has a *greatest lower bound* $\bigwedge_l \mathcal{G}_l$. This greatest lower bound is a graph (V, E, v_0) where $V \subseteq (V_1 \times \ldots \times V_m)$ and $v_0 = (v_{1,0}, \ldots, v_{m,0})$. Moreover V and E are the least sets such that $v_0 \in V$ and the following *closure* axiom is satisfied: if $v = (v_1, \ldots, v_m)$ is in V and for some $t \in T$, $(v_l, t, v'_l) \in E_l$ for all $l \in [1, m]$, then $v' = (v'_1, \ldots, v'_m)$ is in V and (v, t, v') is in E.

Proposition 3. *The reachable state graph of a P/T-net is isomorphic to the greatest lower bound of the reachable state graphs of its atomic subnets.*

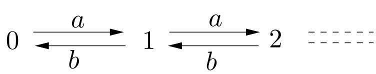
The proposition follows immediately from the firing rule of nets. In view of this fundamental property of reachable state graphs, the modified definition of P/T-regions which is proposed hereafter supplies a basis for the synthesis of P/T-nets from graphs.

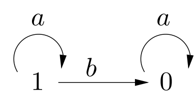
Definition 6 (P/T-regions of a graph). *A* P/T-region *of $\mathcal{G}$ is any atomic P/T-net $\mathcal{N} = (\{p\}, T, F, M_0)$ such that $\mathcal{G} \leq G(\mathcal{N})$.*

Example 8. Consider the following graph $\mathcal{G}$:



The atomic nets $\mathcal{N}_1$ and $\mathcal{N}_2$ defined in example 4 are two P/T-regions of $\mathcal{G}$. The respective graphs $G(\mathcal{N}_1)$ and $G(\mathcal{N}_2)$ are shown below, with $G(\mathcal{N}_1)$ on the left hand side.





The inequalities $\mathcal{G} \leq G(\mathcal{N}_1)$ and $\mathcal{G} \leq G(\mathcal{N}_2)$ are established by the respective morphisms $\sigma_1(v_{i,j}) = i$ and $\sigma_2(v_{i,j}) = j$. □

The next proposition follows from proposition 3 and the (obvious) fact that $G(\mathcal{N}) \leq G(\mathcal{N}_l)$ for every atomic subnet $\mathcal{N}_l$ of $\mathcal{N}$.

Proposition 4. *$\mathcal{G} \cong G(\mathcal{N})$ for some P/T-net $\mathcal{N}$ if and only if $\mathcal{G} \cong \bigwedge_l G(\mathcal{N}_l)$ for some* finite *collection $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$ of P/T-regions of $\mathcal{G}$.*

Definition 6 and proposition 4 are too abstract and they should be refined. We aim in the sequel at equivalent statements with better algorithmic contents. Because $\mathcal{G} \leq G(\mathcal{N}) \Rightarrow L(\mathcal{G}) \subseteq L(\mathcal{N})$, every region of a graph $\mathcal{G}$ is a region of the language $L(\mathcal{G})$. In example 8, all regions of $L(\mathcal{G})$ are regions of $\mathcal{G}$, but this is not true in general. For instance, if $\mathcal{G}$ has edges (v_0, a, v_1) and (v_0, b, v_1), a P/T-net $(\{p\}, T, F, M_0)$ such that $F(a,p) - F(p,a) \neq F(b,p) - F(p,b)$ may be a region of $L(\mathcal{G})$ but it cannot be a region of $\mathcal{G}$. This distinction is clarified below.

Definition 7. *Given $\mathcal{G} = (V, E, v_0)$ and $w \in L(\mathcal{G})$, let ∂w denote the vertex at the end of the walk with label w from the root v_0. Two words w and w' of $L(\mathcal{G})$ are said to* converge *in $\mathcal{G}$ if $\partial w = \partial w'$, and they are said to* diverge *otherwise.*

Proposition 5. *Given graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, $\mathcal{G}_1 \leq \mathcal{G}_2$ if and only if $L(\mathcal{G}_1) \subseteq L(\mathcal{G}_2)$ and every pair of words that converges in $\mathcal{G}_1$ converges in $\mathcal{G}_2$.*

Proof. The two conditions are clearly necessary to the existence of a morphism of graphs from $\mathcal{G}_1$ to $\mathcal{G}_2$. Conversely, when both conditions are satisfied, the map σ defined with $\sigma(\partial_1 w) = \partial_2 w$, where $w \in L(\mathcal{G}_1)$ and ∂_1 and ∂_2 are interpreted w.r.t. $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively, is a morphism of graphs. □

Corollary 1. *Let $\mathcal{N} = (\{p\}, T, F, M_0)$ be a region of $L(\mathcal{G})$, then $\mathcal{N}$ is a region of $\mathcal{G}$ if and only if every pair of words that converges in $\mathcal{G}$ converges in $G(\mathcal{N})$.*

By definition, $\mathcal{G} \leq G(\mathcal{N}_l)$ for every P/T-region $\mathcal{N}_l$ of $\mathcal{G}$, hence $\mathcal{G} \leq \bigwedge_l G(\mathcal{N}_l)$ for every finite collection $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$ of P/T-regions of $\mathcal{G}$. By proposition 5, the converse inequality $\bigwedge_l G(\mathcal{N}_l) \leq \mathcal{G}$ holds *if and only if* $\bigcap_l L(\mathcal{N}_l) \subseteq L(\mathcal{G})$ and every pair of words that diverges in $\mathcal{G}$ diverges in $G(\mathcal{N}_l)$ for some l. Proposition 4 may therefore be restated equivalently as follows.

Proposition 6. *$\mathcal{G} \cong G(\mathcal{N})$ for some P/T-net $\mathcal{N}$ if and only if there exists a finite collection $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$ of P/T-regions of $L(\mathcal{G})$ such that:*
i) *every pair of words that converges in $\mathcal{G}$ converges in $G(\mathcal{N}_l)$ for all $l \in [1, m]$,*
ii) *for every $t \in T$ and for every $w \in L(\mathcal{G})$, if $w \cdot t \notin L(\mathcal{G})$, then $w \cdot t \notin L(\mathcal{N}_l)$ for some $l \in [1, m]$,*
iii) *every pair of words that diverges in $\mathcal{G}$ diverges in $G(\mathcal{N}_l)$ for some $l \in [1, m]$.*
When these conditions are satisfied, $\mathcal{G} \cong G(\mathcal{N})$ where $\mathcal{N}$ is the P/T-net with the set of atomic subnets $\{\mathcal{N}_1, \ldots, \mathcal{N}_m\}$.

A comparison between proposition 6 and proposition 2 indicates that two new problems should be solved if one wants to decide on the P/T-net synthesis problem for classes of graphs. First, the computation of the generating regions defined in section 2.2 should be accomodated to the constraints induced by the requirement *(i)* in the above proposition. Second, the procedure defined in section 2.3 should be augmented so as to decide whether *both* requirements *(ii)* and *(iii)* in the proposition can be satisfied by the generating regions. The two problems are examined in sequence in the sections below.

3.2 A Procedure for Computing Generating Regions

Let us introduce two definitions.

Definition 8. *Given a graph $\mathcal{G} = (V, E, v_0)$, a prefix-closed language $\mathcal{L} \subseteq L(\mathcal{G})$* spans *$\mathcal{G}$ if $(\forall v \in V)\ (\exists w \in \mathcal{L})\ \ v = \partial w$ (in $\mathcal{G}$).*

Definition 9. *For any vector $\boldsymbol{\psi} \in \mathbb{N}^{2n}$, let $\boldsymbol{\psi}_L$ and $\boldsymbol{\psi}_R$ denote the respective vectors in $\mathbb{N}^n$ such that $\boldsymbol{\psi}$ decomposes to $(\boldsymbol{\psi}_L, \boldsymbol{\psi}_R)$ through the isomorphism $\mathbb{N}^{2n} \cong (\mathbb{N}^n \times \mathbb{N}^n)$. For any pair of words w_L and w_R in T^*, let $[w_L, w_R]$ denote the (unique) vector $\boldsymbol{\psi} \in \mathbb{N}^{2n}$ such that $\boldsymbol{\psi}_L = [w_L]$ and $\boldsymbol{\psi}_R = [w_R]$.*

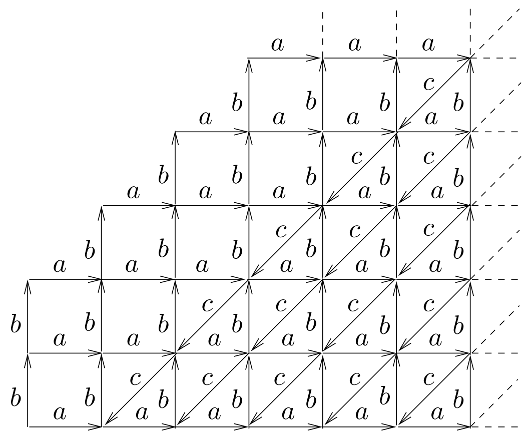
In order to compute effectively from a graph $\mathcal{G} = (V, E, v_0)$ a finite set of P/T-regions generating all regions of this graph, we require that $\mathcal{G}$ should be spanned by some (prefix-closed) language $\mathcal{L}$ such that:

for every edge label $t_j \in T$,

$$\Psi_j = \{\, [w, w'] \,|\, w, w' \in \mathcal{L} \wedge (\partial w, t_j, \partial w') \in E\}$$

is a semi-linear subset of $\mathbb{N}^{2n}$.

Example 9. Let $\mathcal{G}$ be the infinite graph depicted below. It is easily seen that this graph is spanned by the prefix-closed language $\mathcal{L} = (ab)^* a^* + (ab)^* b + (ab)^* bb$.



Let $a = t_1$, $b = t_2$, $c = t_3$ and define the following vectors in $\mathbb{N}^3 \times \mathbb{N}^3 \cong \mathbb{N}^6$ (where ; is used in place of , for better readability):

$$\begin{aligned} \mathbf{0} \quad &= < 0, 0, 0 \,; 0, 0, 0 > \\ \boldsymbol{\delta}_{ab} &= < 1, 1, 0 \,; 1, 1, 0 > \\ \boldsymbol{\delta}_{a} \quad &= < 1, 0, 0 \,; 1, 0, 0 > \\ \boldsymbol{\delta}_{b} \quad &= < 0, 1, 0 \,; 0, 1, 0 > \\ \boldsymbol{\delta}_{bb} &= < 0, 2, 0 \,; 0, 2, 0 > \end{aligned}$$

Let $\Psi = \mathbf{0} \cdot (\boldsymbol{\delta}_{ab} + \boldsymbol{\delta}_a)^* + \boldsymbol{\delta}_b \cdot (\boldsymbol{\delta}_{ab})^* + \boldsymbol{\delta}_{bb} \cdot (\boldsymbol{\delta}_{ab})^*$, thus Ψ is a semi-linear set. For $t \in \{a, b, c\}$, the respective sets $\Psi_t = \{\, [w, w'] \mid w, w' \in \mathcal{L} \wedge (\partial w, t, \partial w') \in E\}$ may be given the semi-linear expressions:

$$\Psi_a = < 0, 0, 0 \,; 1, 0, 0 > \cdot\, \Psi \tag{7}$$

$$\Psi_b = < 0, 0, 0 \,; 0, 1, 0 > \cdot\, (\boldsymbol{\delta}_{ab} + \boldsymbol{\delta}_a)^* + < 0, 0, 0 \,; 0, 1, 0 > \cdot\, \boldsymbol{\delta}_b \cdot (\boldsymbol{\delta}_{ab})^* \tag{8}$$

$$\Psi_c = < 2, 1, 0 \,; 1, 0, 0 > \cdot\, (\boldsymbol{\delta}_{ab} + \boldsymbol{\delta}_a)^* \tag{9}$$

The requested condition is fulfilled. □

We define now a procedure that computes the generating P/T-regions of a graph from a language $\mathcal{L}$ spanning this graph and from the associated semi-linear sets Ψ_j ($j \in [1, n]$). Recall that an atomic P/T-net $\mathcal{N} = (\{p\}, T, F, M_0)$ may be represented as a $(2n+1)$-vector $\boldsymbol{x} =< x_0, x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n} >$ where $x_0 = M_0(p)$ and for all $j \in [1, n]$, $x_j = F(p, t_j)$ and $x_{n+j} = F(t_j, p)$. We claim that a $(2n+1)$-vector $\boldsymbol{x} =< x_0, x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n} >$ defines a region of $\mathcal{G}$ if and only if all its entries x_k are non-negative integers, and the following inequalities and equations hold for all $t_j \in T$ and $\boldsymbol{\psi} \in \Psi_j$:

$$\sum_{i=1}^{n} \boldsymbol{\psi}_L[i] \times (x_{(n+i)} - x_i) \geq x_j - x_0 \tag{10}$$

$$\sum_{i=1}^{n} \boldsymbol{\psi}_R[i] \times (x_{(n+i)} - x_i) = x_{(n+j)} - x_j + \sum_{i=1}^{n} \boldsymbol{\psi}_L[i] \times (x_{(n+i)} - x_i) \tag{11}$$

In order to see that both conditions are necessary, let $\mathcal{N}$ be the atomic P/T-net defined by the vector $\boldsymbol{x}$. If $\mathcal{N}$ is a region of $\mathcal{G}$, then $L(\mathcal{G}) \subseteq L(\mathcal{N})$ and therefore $\mathcal{L} \subseteq L(\mathcal{N})$. The inequality 10 states that whenever $u \in \mathcal{L}$ and $u \cdot t_j \in L(\mathcal{G})$, if u can be fired in $\mathcal{N}$ then $u \cdot t_j$ can be fired in $\mathcal{N}$. This must be true since $L(\mathcal{G})$ cannot be included in $L(\mathcal{N})$ otherwise. Under the same assumptions, the equation 11 states that whenever $u \cdot t_j$ and v converge in $\mathcal{G}$ for some v in $\mathcal{L}$, $u \cdot t_j$ and v converge in $G(\mathcal{N})$. The corollary 1 states that this also must be true.

In order to establish the claim, it remains to show that whenever the conditions 10 and 11 hold for a vector $\boldsymbol{x}$, the atomic P/T-net $\mathcal{N}$ defined by $\boldsymbol{x}$ is a region of the graph $\mathcal{G}$. By corollary 1, it suffices to prove that $L(\mathcal{G}) \subseteq L(\mathcal{N})$ and that all pairs of words of $L(\mathcal{G})$ that converge in $\mathcal{G}$ converge in $G(\mathcal{N})$.

Proposition 7. *$\mathcal{L} \subseteq L(\mathcal{N})$ and moreover, the pairs of words of $\mathcal{L}$ that converge in $\mathcal{G}$ converge also in $G(\mathcal{N})$.*

Proof. As $\mathcal{L}$ is prefix-closed, $\mathcal{L} \subseteq L(\mathcal{N})$ follows from 10 by induction on words. Consider $v, w \in \mathcal{L}$ such that $\partial v = \partial w$ in $\mathcal{G}$. If $v = w$, they do converge in $G(\mathcal{N})$. If $v \neq w$, at least one of them is non-empty. Assume w.l.o.g. that $v = u \cdot t_j$ with $t_j \in T$. As $\mathcal{L}$ is prefix-closed, $u \in \mathcal{L}$. Therefore, $\boldsymbol{\psi} = [u, v]$ and $\boldsymbol{\psi'} = [u, w]$ are vectors in Ψ_j. It follows from the equation 11 that $\sum_{i=1}^{n} [v]_i \times (x_{(n+i)} - x_i) = \sum_{i=1}^{n} [w]_i \times (x_{(n+i)} - x_i)$. Therefore, v and w converge in $G(\mathcal{N})$. □

Proposition 8. *For all $v' \in L(\mathcal{G})$ and for all $w \in \mathcal{L}$ such that $\partial v' = \partial w$ in $\mathcal{G}$:*
i) *$v' \in L(\mathcal{N})$, and*
ii) *v' and w converge in $G(\mathcal{N})$.*

Proof. Since $L(\mathcal{G})$ is prefix-closed, one may use an induction on words. As $\varepsilon \in \mathcal{L}$, the basis of the induction is clear from proposition 7. For the induction step, let $v' = u' \cdot t_j$ where $t_j \in T$. Choose $u, v \in \mathcal{L}$ such that $\partial u = \partial u'$ and $\partial v = \partial v'$ (since $\mathcal{L}$ spans $\mathcal{G}$, such words must exist). As $v' \in L(\mathcal{G})$, $(\partial u', t_j, \partial v')$ is an edge of $\mathcal{G}$, and this edge is equal to $(\partial u, t_j, \partial v)$. By proposition 7, $u, v \in L(\mathcal{N})$ since $u, v \in \mathcal{L}$. Hence $u \cdot t_j \in L(\mathcal{N})$, in view of the inequality 10, and $u \cdot t_j$ and v converge in $G(\mathcal{N})$, in view of the equation 11. From the induction hypothesis, $u' \in L(\mathcal{N})$, and u and u' converge in $G(\mathcal{N})$. Therefore, $u' \cdot t_j \in L(\mathcal{N})$, and $u' \cdot t_j$ and $u \cdot t_j$ converge in $G(\mathcal{N})$. This entails that $v' \in L(\mathcal{N})$, and v' and v converge in $G(\mathcal{N})$. As $\partial v = \partial v' = \partial w$ and $v, w \in \mathcal{L}$, v and w converge in $G(\mathcal{N})$, by proposition 7. Therefore, v' and w converge in $G(\mathcal{N})$. □

Corollary 2. *$\mathcal{N}$ is a P/T-region of $\mathcal{G}$.*

Proof. Seeing that $\mathcal{L}$ spans $\mathcal{G}$, proposition 8 entails that whenever two words v', v'' of $L(\mathcal{G})$ converge in $\mathcal{G}$, they converge in $G(\mathcal{N})$. □

Using the assumption that all sets Ψ_j are semi-linear, the (possibly) infinite collection of linear homogeneous constraints that derive as instances of (10) or (11) for some $\boldsymbol{\psi} \in \Psi_j$ $(j \in [1, n])$ may be reduced to a finite linear system.

As regards the inequality (10), which is similar to the inequality (1), the reduction follows the same lines as in section 2.2 (the former set $[\mathcal{L}/t_j]$ is replaced with the semi-linear set $\Sigma_j = \{\boldsymbol{\psi}_L \mid \boldsymbol{\psi} \in \Psi_j\}$).

As regards the equation (11), let $\Delta_j = \{\boldsymbol{\psi}_R - \boldsymbol{\psi}_L \mid \boldsymbol{\psi} \in \Psi_j\}$ for each $j \in [1, n]$. Since Ψ_j is a semi-linear subset of $\mathbb{N}^{2n}$, and in view of the definition 5, Δ_j is a finite union of linear sets $\boldsymbol{e} \cdot \mathcal{F}^*$, where $\boldsymbol{e} \in \mathbb{Z}^n$ and $\mathcal{F}$ is a finite subset of $\mathbb{Z}^n$. For each linear subset $\boldsymbol{e} \cdot \mathcal{F}^*$ of Δ_j, the set of constraints that derive as instances of the equation 11 for some $\boldsymbol{\psi} \in \boldsymbol{e} \cdot \mathcal{F}^*$ may be replaced equivalently with the finite linear system:

$$\sum_{i=1}^{n} \boldsymbol{e}[i] \times (x_{(n+i)} - x_i) = x_{(n+j)} - x_j \tag{12}$$

$$\sum_{i=1}^{n} \boldsymbol{f}[i] \times (x_{(n+i)} - x_i) = 0 \tag{13}$$

where f ranges over the finite set $\mathcal{F}$. Therefore, the collection of instances of the equation 11 for all $j \in [1, n]$ and for all $\boldsymbol{\psi} \in \Psi_j$ reduces to a finite system.

Let $\mathcal{S}$ be the finite linear system in the variables $x_0, \ldots, x_{2n}$ formed of the reduced systems defined above, plus inequalities $x_k \geq 0$ for all $k \in [0, 2n]$. One can compute as was explained in section 2.2 a finite and minimal set of solutions $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$ of $\mathcal{S}$ in $\mathbb{N}^{2n+1}$, called the *generating regions* of the graph $\mathcal{G}$, such that the regions of this graph may be characterized as follows:

A vector $\boldsymbol{x}$ is a P/T-region of $\mathcal{G}$ if and only if $\boldsymbol{x} \in \mathbb{N}^{2n+1}$ and $\boldsymbol{x} = \sum_{l=1}^{m} q_l\, \boldsymbol{x}_l$ for some non-negative rational coefficients q_l.

Example 10. Let us compute the generating regions of the graph $\mathcal{G}$ from example 9. In order to enhance the readability, let $\boldsymbol{x} =< m_0, {}^\bullet a, {}^\bullet b, {}^\bullet c, a^\bullet, b^\bullet, c^\bullet >$ where $t_1 = a$, $t_2 = b$, and $t_3 = c$. The respective sets Δ_1, Δ_2, and Δ_3 are the singleton sets defined with $\Delta_1 = \Delta_a = \{< 1, 0, 0 >\}$, $\Delta_2 = \Delta_b = \{< 0, 1, 0 >\}$, and $\Delta_3 = \Delta_c = \{< -1, -1, 0 >\}$. The finite system derived from equation 11 is:

$$\begin{array}{l} a^\bullet - {}^\bullet a = a^\bullet - {}^\bullet a \\ b^\bullet - {}^\bullet b = b^\bullet - {}^\bullet b \\ {}^\bullet a - a^\bullet + {}^\bullet b - b^\bullet = c^\bullet - {}^\bullet c \end{array}$$

Two trivial equations may be dropped. The linear constraints generated from the instances of the inequality 10, where $\boldsymbol{\psi}_L$ ranges over the respective semi-linear sets $\Sigma_j = \{\boldsymbol{\psi}_L \mid \boldsymbol{\psi} \in \Psi_j\}$ $(j \in [1, 3])$, are as follows.
$\Sigma_1 = \Sigma_a = \langle 0, 0, 0\rangle \cdot (\langle 1, 1, 0\rangle + \langle 1, 0, 0\rangle)^* + \langle 0, 1, 0\rangle \cdot \langle 1, 1, 0\rangle^* + \langle 0, 2, 0\rangle \cdot \langle 1, 1, 0\rangle^*$

produces the constraints

$$\begin{array}{l} 0 \geq {}^\bullet a - m_0 \\ b^\bullet - {}^\bullet b \geq {}^\bullet a - m_0 \\ 2\,(b^\bullet - {}^\bullet b) \geq {}^\bullet a - m_0 \\ a^\bullet - {}^\bullet a \geq 0 \\ a^\bullet - {}^\bullet a + b^\bullet - {}^\bullet b \geq 0 \end{array}$$

$\Sigma_1 = \Sigma_b = \langle 0,0,0\rangle \cdot (\langle 1,1,0\rangle + \langle 1,0,0\rangle)^* + \langle 0,1,0\rangle \, \langle 1,1,0\rangle^*$ adds two constraints

$$0 \geq {}^\bullet b - m_0$$
$$b^\bullet - {}^\bullet b \geq {}^\bullet b - m_0$$

$\Sigma_3 = \Sigma_c = \langle 2,1,0\rangle \cdot (\langle 1,1,0\rangle + \langle 1,0,0\rangle)^*$ brings finally one more constraint

$$2\,(a^\bullet - {}^\bullet a) + b^\bullet - {}^\bullet b \geq {}^\bullet c - m_0$$

The generating regions computed by Chernikova's algorithm are the following:

m_0	2	1	1	1	0	2	1	0	0	2	1	1	2	1	1
${}^\bullet a$	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0
${}^\bullet b$	1	0	1	1	0	2	0	0	0	1	0	1	2	1	0
${}^\bullet c$	3	1	1	1	1	3	1	2	1	0	0	0	0	0	0
$a^\bullet$	1	1	0	1	1	1	0	1	0	1	1	1	1	0	0
$b^\bullet$	0	0	1	1	0	1	0	0	1	0	0	1	1	1	0
$c^\bullet$	3	1	1	1	0	3	1	1	0	0	0	0	0	0	0

Many generating regions are useless, as we will show later on. □

We claim that the P/T-net $\mathcal{N}$ built up from the atomic subnets $\mathcal{N}_1 \ldots \mathcal{N}_m$ defined by the generating regions $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$ is the best net-approximation of the graph $\mathcal{G}$ in the following sense:

$$\mathcal{G} \leq G(\mathcal{N}) \quad \textit{and} \quad \forall \mathcal{N}' \quad \mathcal{G} \leq G(\mathcal{N}') \Longrightarrow G(\mathcal{N}) \leq G(\mathcal{N}')$$

The relation $\mathcal{G} \leq G(\mathcal{N})$ is easily established, as $\mathcal{G} \leq G(\mathcal{N}_l)$ for all $l \in [1,m]$ (by definition of the regions of a graph) and $G(\mathcal{N}) = \bigwedge_l G(\mathcal{N}_l)$ (by proposition 3). The two propositions below aim at establishing the second part of the claim.

Proposition 9. $\forall \mathcal{N}' \quad \mathcal{G} \leq G(\mathcal{N}') \Longrightarrow L(\mathcal{N}) \subseteq L(\mathcal{N}')$

Proof. Assuming the converse, let $w \in L(\mathcal{N}) \cap L(\mathcal{N}')$ and $t_j \in T$ such that $w \cdot t_j \in L(\mathcal{N})$ and $w \cdot t_j \notin L(\mathcal{N}')$. Necessarily, $\sum_i [w]_i \times (\boldsymbol{x}'[n+i] - \boldsymbol{x}'[i]) < \boldsymbol{x}'[j] - \boldsymbol{x}'[0]$ for some $(2n+1)$-vector $\boldsymbol{x}'$ representing an atomic subnet of $\mathcal{N}'$. As $\mathcal{G} \leq G(\mathcal{N}')$ and $G(\mathcal{N}') \leq G(\mathcal{N}'')$ for every subnet $\mathcal{N}''$ of $\mathcal{N}'$, this atomic subnet of $\mathcal{N}'$ is a region of $\mathcal{G}$. Therefore, $\boldsymbol{x}' = \sum_{l=1}^{m} q_l \, \boldsymbol{x}_l$ for some non-negative rational coefficients q_l. Owing to the sign of the coefficients, it must be true for some $l \in [1,m]$ that $\sum_i [w]_i \times (\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i]) < \boldsymbol{x}_l[j] - \boldsymbol{x}_l[0]$. But this inequality entails $w \cdot t_j \notin L(\mathcal{N}_l)$ and hence $w \cdot t_j \notin L(\mathcal{N})$, a contradiction. □

Proposition 10. *If $\mathcal{G} \leq G(\mathcal{N}')$, then two words of $L(\mathcal{N})$ converge in $G(\mathcal{N}')$ whenever they converge in $G(\mathcal{N})$.*

Proof. Assuming the converse, let w, w' converge in $G(\mathcal{N})$ and diverge in $G(\mathcal{N}')$. Necessarily, $\sum_i [w]_i \times (\boldsymbol{x}'[n+i] - \boldsymbol{x}'[i]) \neq \sum_i [w']_i \times (\boldsymbol{x}'[n+i] - \boldsymbol{x}'[i])$ for some

$(2n+1)$-vector $\boldsymbol{x}'$ representing an atomic subnet of $\mathcal{N}'$. Like in the proof of the former proposition, $\boldsymbol{x}' = \sum_{l=1}^{m} q_l\, \boldsymbol{x}_l$, and it must be true for some $l \in [1, m]$ that $\sum_i [w]_i \times (\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i]) \neq \sum_i [w']_i \times (\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i])$. As a consequence, w and w' diverge in $G(\mathcal{N}_l)$ and hence in $G(\mathcal{N})$, a contradiction. □

In view of these propositions, the second part of the claim follows from the proposition 5. Optimal net-approximations may therefore be computed in any class of graphs (V, E, v_0) spanned by languages $\mathcal{L}$ such that, for every $t_j \in T$, the *transition relation* $\mathcal{T}_j = \{\, (w, w') \,|\, w \in \mathcal{L} \wedge w' \in \mathcal{L} \wedge (\partial w, t_j, \partial w') \in E \,\}$ is semi-linear (*i.e.* $\{\, [w, w'] \,|\, (w, w') \in \mathcal{T}_j \,\}$ is semi-linear). A trivial example is the class of finite graphs. Two other examples are the classes of *labelled* domains induced by recognizable sets of Mazurkiewicz traces, or by Finite Automata with Concurrency Relations [13]. In both cases, the language of a labelled domain $\mathcal{G}$ is a regular language $\mathcal{L}$, and $\{\, [w, w'] \,|\, (w, w') \in \mathcal{T}_j \,\} = \{\, [w, w \cdot t_j] \,|\, (w \cdot t_j) \in \mathcal{L}\}$, hence the transition relations are semi-linear. Optimal net-approximations may also be computed in any class of graphs where the transition relations $\mathcal{T}_j$ may be defined with finite 2-tape automata (this is the case for deterministic pushdown graphs [42][43]), or more generally with (non-deterministic) 2-tape pushdown automata (this particular use of 2-tape pda's is a suggestion of ours).

3.3 A Procedure for the Decision of the Net Synthesis Problem

We show in this section that under additional conditions on $\mathcal{G}$ and the spanning language $\mathcal{L}$, one can decide whether $\mathcal{G}$ has a P/T-net generator, *i.e.* whether $\mathcal{G} \cong G(\mathcal{N})$ where $\mathcal{N}$ is the net constructed in section 3.2.

Because $\mathcal{G} \leq G(\mathcal{N})$ and $G(\mathcal{N})$ is the least net-approximation of $\mathcal{G}$, the graph $\mathcal{G}$ has a P/T-net generator *if and only if* $G(\mathcal{N}) \leq \mathcal{G}$. By proposition 5, the following two conditions are necessary and sufficient:

i) for every $w \in L(\mathcal{G})$ and $t_j \in T$, if $w \cdot t_j \notin L(\mathcal{G})$, then $w \cdot t_j \notin L(\mathcal{N})$,
ii) every pair of words of $L(\mathcal{N})$ that diverges in $\mathcal{G}$ diverges in $G(\mathcal{N})$.

The condition *(i)* reads as $L(\mathcal{N}) \subseteq L(\mathcal{G})$, hence when it holds, $L(\mathcal{N}) = L(\mathcal{G})$ because $\mathcal{G} \leq G(\mathcal{N}) \Longrightarrow L(\mathcal{G}) \subseteq L(\mathcal{N})$. We thus retrieve the respective conditions *(ii)* and *(iii)* stated in proposition 6 (the atomic subnets of $\mathcal{N}$ are the generating regions $\mathcal{N}_l$ of $\mathcal{G}$, hence they are regions of $L(\mathcal{G})$ and they satisfy condition *(i)* in prop. 6). Recalling that $\mathcal{L} \subseteq L(\mathcal{G})$, the above conditions may be simplified to:

i') for every $w \in \mathcal{L}$ and $t_j \in T$, if $w \cdot t_j \notin L(\mathcal{G})$, then $w \cdot t_j \notin L(\mathcal{N})$,
ii') every pair of words of $\mathcal{L}$ that diverges in $\mathcal{G}$ diverges in $G(\mathcal{N})$.

(i') entails (i) : Let $w \in L(\mathcal{G})$ and $w \cdot t_j \notin L(\mathcal{G})$. As $\mathcal{L}$ spans $\mathcal{G}$, $\partial w = \partial u$ for some u in $\mathcal{L}$. As $w \cdot t_j \notin L(\mathcal{G})$, $u \cdot t_j \notin L(\mathcal{G})$, hence $u \cdot t_j \notin L(\mathcal{N})$. As $\mathcal{G} \leq G(\mathcal{N})$ and w and u converge in $\mathcal{G}$, they lead to the same marking of $\mathcal{N}$, hence $w \cdot t_j \notin L(\mathcal{N})$.
(ii') entails (ii) : Let w and w' be words of $L(\mathcal{G})$ such that $\partial w \neq \partial w'$. As $\mathcal{L}$ spans $\mathcal{G}$, $\partial w = \partial u$ and $\partial w' = \partial u'$ for some u and u' in $\mathcal{L}$. As $\partial u \neq \partial u'$, u and u' lead

to distinct markings of $\mathcal{N}$. As $\mathcal{G} \leq G(\mathcal{N})$, w and u lead to the same marking of $\mathcal{N}$, and similarly do w' and u', hence w and w' diverge in $G(\mathcal{N})$.

In order to decide whether $\mathcal{G} \cong G(\mathcal{N})$, we add the following requirements on $\mathcal{G}$ and its spanning language $\mathcal{L}$:

1. $Dis = \{\, [w, w'] \mid w \in \mathcal{L} \wedge w' \in \mathcal{L} \wedge \partial w \neq \partial w' \,\}$ should be semi-linear,
2. $Inh_j = \{\, [w] \mid w \in \mathcal{L} \wedge w \cdot t_j \notin L(\mathcal{G}) \,\}$ should be semi-linear for all $j \in [1, m]$.

Assuming these requirements are fulfilled, we propose a decision procedure. Recall that $\mathcal{N}$ has m atomic subnets $\mathcal{N}_1 \ldots \mathcal{N}_m$, *viz.* the generating regions of $\mathcal{G}$, represented with $(2n+1)$-vectors $\boldsymbol{x}_1 \ldots \boldsymbol{x}_m$. Thus, for any $w \in \mathcal{L}$ and for any $l \in [1, m]$, the marking reached after firing w in the atomic subnet $\mathcal{N}_l = (\{p\}, T, F, p = m_0)$ is defined with $p = m_0 + \sum_i [w]_i \times (\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i])$.

The condition *(ii')* is satisfied *if and only if* $Dis \subseteq \cup_{l=1}^{m} Dis_l$ where:

$$Dis_l = \{\, \boldsymbol{\psi} \in \mathbb{N}^{2n} \mid \textstyle\sum_{i=1}^{n} (\boldsymbol{\psi}_R[i] - \boldsymbol{\psi}_L[i]) \times (\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i]) \neq 0 \,\}$$

Now, for fixed l, all coefficients $(\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i])$ are constants in $\mathbb{Z}$, hence the above formula is a Presburger formula and Dis_l is a semi-linear subset of $\mathbb{N}^{2n}$. Such subsets form an effective boolean algebra. Therefore, when Dis is semi-linear, one can decide whether the condition *(ii')* is satisfied.

The condition *(i')* is satisfied *if and only if*, for all $j \in [1, m]$, $Inh_j \subseteq \cup_{l=1}^{m} Inh_j^l$ where:

$$Inh_j^l = \{\, \boldsymbol{\psi} \in \mathbb{N}^{n} \mid \textstyle\sum_{i=1}^{n} \boldsymbol{\psi}[i] \times (\boldsymbol{x}_l[n+i] - \boldsymbol{x}_l[i]) < \boldsymbol{x}_l[j] - \boldsymbol{x}_l[0]$$

As Inh_j^l is defined by a Presburger formula, Inh_j^l is a semi-linear subset of $\mathbb{N}^{2n}$. Therefore, when all inhibitor sets Inh_j are semi-linear, one can decide whether the condition *(i')* is satisfied.

The P/T-net synthesis problem is decidable in classes of graphs spanned by languages $\mathcal{L}$ such that the set Dis and all sets Ψ_j and Inh_j are semi-linear

---

Like in section 2, it may occur that $\mathcal{G} \cong G(\mathcal{N}')$ for some proper subnet $\mathcal{N}'$ of $\mathcal{N}$. Minimal nets $\mathcal{N}'$ may be derived from minimal subsets of generating regions such that $Dis \subseteq \cup_l Dis_l$ and $Inh_j \subseteq \cup_l Inh_j^l$ for l ranging over indices of regions in these subsets.

Example 11. For the graph $\mathcal{G}$ in the example 9, $\mathcal{L} = (ab)^* a^* + (ab)^* b + (ab)^* bb$, and $Dis = \{\boldsymbol{\psi} \mid \boldsymbol{\psi}_L \in [\mathcal{L}] \wedge \boldsymbol{\psi}_R \in [\mathcal{L}] \wedge \boldsymbol{\psi}_L \neq \boldsymbol{\psi}_R\}$ where $[\mathcal{L}]$ is the commutative image of $\mathcal{L}$. As $\mathcal{L}$ is a regular language, $[\mathcal{L}]$ is a semi-linear set, hence Dis is a semi-linear set (because it is defined by a Presburger formula). Consider now the two regions of $\mathcal{G}$ represented respectively by the columns 1 and 9 in the table at the end of the example 10. The respective sets Dis_1 and Dis_9 are given by the semi-linear expressions:

$Dis_1 = \{< n_a, n_b, n_c \,;\, n'_a, n'_b, n'_c > \mid n_a - n_b \neq n'_a - n'_b \,\}$

$Dis_9 = \{< n_a, n_b, n_c\,;\, n'_a, n'_b, n'_c > \mid n_b - n_c \neq n'_b - n'_c\,\}$

Clearly, for any $\psi =< n_a, n_b, n_c\,;\, n'_a, n'_b, n'_c >\in Dis$, $n_c = 0$ and $n'_c = 0$ because $\mathcal{L} \subseteq \{a, b\}^*$, and $n_a \neq n'_a$ or $n_b \neq n'_b$. If $n_b \neq n'_b$ then $\psi \in Dis_9$. If $n_a \neq n'_a$ and $n_b = n'_b$ then $\psi \in Dis_1$. Thus, $Dis \subseteq Dis_1 \cup Dis_9$.

The respective inhibitor sets $Inh_1 = Inh_a$, $Inh_2 = Inh_b$, and $Inh_3 = Inh_c$ are given by the semi-linear expressions:

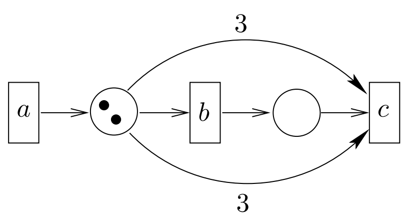
$Inh_1 = \emptyset$
$Inh_2 = \langle 0, 2, 0\rangle \cdot \langle 1, 1, 0\rangle^*$
$Inh_3 = \langle 0, 0, 0\rangle \cdot \langle 1, 0, 0\rangle^* + (\langle 0, 0, 0\rangle + \langle 0, 1, 0\rangle + \langle 0, 2, 0\rangle) \cdot \langle 1, 1, 0\rangle^*$

As $Inh_2^1 = \{\langle n_a, n_b, n_c\rangle \mid n_a - n_b < -1\}$, it follows that $Inh_2 \subseteq Inh_2^1$.

Now $Inh_3^1 = \{\langle n_a, n_b, n_c\rangle \mid n_a - n_b < 1\}$, and $Inh_3^9 = \{\langle n_a, n_b, n_c\rangle \mid n_b - n_c < 1\}$.

Clearly, $\langle 0, 0, 0\rangle \cdot \langle 1, 0, 0\rangle^* \subseteq Inh_3^9$, and $Inh_3 \subseteq Inh_3^1 \cup Inh_3^9$.

The graph $\mathcal{G}$ is therefore isomorphic to the reachable state graph of the net built from the atomic nets defined by the respective vectors $\boldsymbol{x}_1 = \langle 2, 0, 1, 3, 1, 0, 3\rangle$ and $\boldsymbol{x}_9 = \langle 0, 0, 0, 1, 0, 1, 0\rangle$. This net is shown in the figure below. □



A well known class of graphs where the requirements 1 and 2 are fulfilled is the class of the deterministic pushdown graphs. This assertion is not trivial and it follows from the results establihed by Sénizergues in his unpublished work [43]. Therefore, the general decision procedure presented in this section may be considered as an extension of the specific procedure proposed in [19] for the deterministic pushdown graphs. This extension owes much to Sénizergues's view of graphs with an automatic structure. Building on his ideas, a wide class of graphs where the P/T-net synthesis problem is decidable was proposed in [7].

4 Conclusion

In this paper, we focussed on the problem whether a language or an infinite graph may be realized exactly by an unbounded P/T-net, a problem which was ignored in [6]. We have shown that this problem is decidable under strong requirements of semi-linearity, met by deterministic pushdown languages and graphs, and by graphs in wider families. We have shown that the exact net-realization problem is undecidable for pushdown languages and for HMSC languages. These negative results, and the strong constraints imposed for deciding on the synthesis problem when this is possible, indicate that approximate net-realizations of languages or graphs is often the best one can expect. We have shown that (least) over-approximations by nets may be computed under mild assumptions of semi-linearity on languages or graphs. It was argued that such approximations are particularly adequate in the context of supervisory control problems.

It might be objected that the procedures we have proposed are too limited, since the P/T-nets produced by these procedures have always semi-linear sets of reachable markings. We are conscious of this limitation, but we do not see how it could be removed consistently with our approach.

If one agrees that the exact realization of languages or graphs by nets is not the central problem, there are two ways for further research. One is to search for approximate realizations of languages or graphs by nets, as was proposed in this paper. A second way is to change the data of the P/T-net synthesis problem, by taking *sets* of graphs or languages as inputs, in place of individuals. Then, the problem is to search for a net $\mathcal{N}$ such that $G(\mathcal{N})$ or $L(\mathcal{N})$ belongs to the given set. This problem has been solved in [7] for sets of graphs defined by path-automatic specifications, a combination of modal transition systems and automatic graphs. We are currently working on a similar problem in the context of languages.

References

1. Alur, R., Yannakakis, M.: Model Checking of Message Sequence Charts. Proc. Concur, LNCS **1664** (1999) 114-129.
2. Badouel, E., Bernardinello, L., Darondeau, Ph.: Polynomial Algorithms for the Synthesis of Bounded Nets. Proc. CAAP, LNCS **915** (1995) 647-679.
3. Badouel, E., Bernardinello, L., Darondeau, Ph.: The Synthesis Problem for Elementary Net Systems is NP-complete. *Theoretical Computer Science* **186** (1997) 107-134.
4. Badouel, E., Caillaud, B., Darondeau, Ph.: Distributing finite automata through Petri net synthesis. *Formal Aspects of Computing* **13** (2002) 447-470.
5. Badouel, E., Darondeau, Ph.: On the Synthesis of General Petri Nets. Inria Research Report 3025 (1996).
6. Badouel, E., Darondeau, Ph.: Theory of regions. In W. Reisig and G. Rozenberg, editors, *Lectures on Petri Nets I: Basic Models*, Advances in Petri Nets, LNCS **1491** (1998) 529-586.
7. Badouel, E., Darondeau, Ph.: The Synthesis of Petri Nets from Path-Automatic Specifications. draft, submitted (2003).
8. Bednarczyk, M., Borzyszkowski, A.: On Concurrent Realization of Reactive Systems and their Morphisms. In H. Ehrig, G. Juhas, J. Padberg and G. Rozenberg, editors, *Unifying Petri Nets*, Advances in Petri Nets, LNCS **2128** (2001) 346-379.
9. Bernardinello, L., De Michelis, G., Petruni, K., Vigna, S.: On the Synchronic Sructure of Transition Systems. In J. Desel, editor, *Structures in Concurrency Theory*, Workshops on Computing, Springer Verlag (1996) 11-31.
10. Berstel, J.: *Transductions and Context-Free Languages*. Teubner Verlag (1979).
11. Bernardinello, L.: Synthesis of Net Systems. Proc. ATPN, LNCS **691** (1993) 11-31.

12. Bernardinello, L., Ferrigato, C., Pomello, L.: An Algebraic Model of Observable Properties in Distributed Systems. *Theoretical Computer Science* **290** (2003) 637-678.
13. Bracho, F., Droste, M.: Labelled Domains and Automata with Concurrency Relations. *Theoretical Computer Science* **135**,2 (1994) 289-318.
14. Caillaud, B., Darondeau, Ph., Helouet, L., Lesventes, G.: HMSCs as Partial Specifications ... with PNs as Completions. In F. Cassez, C. Jard, B. Rozoy, M.D. Ryan, editors, *Modeling and Verification of Parallel Processes*: 4th summer school / MOVEP'2000, LNCS **2067** (2001) 125-152.
15. Chernikova, N.: Algorithm for finding a general formula for the non-negative solutions of a system of linear inequalities. *USSR Computational Mathematics and Mathematical Physics* **5**,2 (1965) 228-233.
16. Cortadella, J., Kishinevsky, M., Kondratyev, A., Lavagno, L., Yakovlev, A.: Hardware and Petri Nets: Application to Asynchronous Circuit Design. Proc. ATPN, LNCS **1825** (2000) 1-15.
17. Cortadella, J., Kishinevsky, M., Lavagno, L., Yakovlev, A.: Deriving Petri Nets from Finite Transition Systems. *IEEE Transactions on Computers* **47**,8 (1998) 859-882.
18. Darondeau, Ph.: Deriving Petri Nets from Formal Languages. Proc. CONCUR, LNCS **1466** (1998) 533-548.
19. Darondeau, Ph.: On the Petri net realization of context-free graphs. *Theoretical Computer Science* **258** (2001) 573-598.
20. Desel, J., Reisig, W.: The Synthesis Problem of Petri Nets. *Acta Informatica* **33** (1996) 297-315.
21. Droste, M., Shortt, R.M.: Petri Nets and Automata with Concurrency Relations - an Adjunction. In M. Droste and Y. Gurevich, editors, *Semantics of Programming Languages and Model Theory* (1993) 69-87.
22. Ehrenfeucht, A., Rozenberg, G.: Partial (Set) 2-Structures; *Part I:* Basic Notions and the Representation Problem. *Acta Informatica* **27** (1990) 315-342.
23. Ehrenfeucht, A., Rozenberg, G.: Partial (Set) 2-Structures; *Part II:* State Spaces of Concurrent Systems. *Acta Informatica* **27** (1990) 343-368.
24. Esparza, J.: Decidability of model-checking for infinite-state concurrent systems. *Acta Informatica* **34**,2 (1997) 85-107.
25. Fernau, H.: Parallel Communicating Grammar Systems with Terminal Transmission. *Acta Informatica* **37** (2001) 511-540.
26. Fischer, P.C., Rosenberg, A.L.: Multitape One-Way Nonwriting Automata. *Journal of Computer and System Sciences* **2** (1968) 88-101.
27. Ghaffari, A., Rezg, N., Xie, X.: Live and Maximally Permissive Controller Synthesis using Theory of Regions. In B. Caillaud, Ph. Darondeau, L. Lavagno and X. Xie, editors, *Synthesis and Control of Discrete Event Systems*, Kluwer Academic Publishers (2002)
28. Ginsburg, S., and Spanier, E.H.: Bounded Algol-like Languages. *Transactions of the American Mathematical Society* (1964) 333-368.
29. Ginsburg, S., and Spanier, E.H.: Semigroups, Presburger formulas, and Languages. *Pacific Journal of Mathematics* **16** (1966) 285-296.
30. Ginsburg, S., and Spanier, E.H.: AFL with the Semilinear Property. *Journal of Computer and System Sciences* **5** (1971) 365-396.
31. Harrisson, M.A.: *Introduction to Formal Language Theory.* Addison-Wesley (1978).
32. Holzer, M., Kutrib, M.: Flip-Pushdown Automata: k+1 Pushdown Reversals are Better than k. IFIG Research Report 0206, Universitaet Giessen (2002).

33. Hoogers, P.W., Kleijn, H.C.M., Thiagarajan, P.S.: A Trace Semantics for Petri Nets. Proc. ICALP, LNCS **623** (1992) 595-604.
34. Hopcroft, J., Ullman, J.: *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley (1979).
35. Ibarra, O.: Simple Matrix Languages. *Information and Control* **17** (1970) 359-394.
36. Mayr, E.: An Algorithm for the General Petri Net Reachability Problem. *SIAM Journal on Computing* **13** (1984) 441-460.
37. Mukund, M.: Petri Nets and Step Transition Systems. *International Journal of Foundations of Computer Science* **3**,4 (1992) 443-478.
38. Nielsen, M., Rozenberg, G., Thiagarajan, P.S.: Elementary Transition Systems. *Theoretical Computer Science* **96**,1 (1992) 3-33.
39. Pelz, E.: Closure Properties of Deterministic Petri Nets. Proc. Stacs, LNCS **247** (1987) 373-382
40. Ramadge, P.J, Wonham, W.M.: Supervisory Control of a Class of Discrete Event Processes. *SIAM Journal of Control and Optimization* **25**,1 (1987) 206-230.
41. Schrijver, A.: *Theory of Linear and Integer Programming*. John Wiley (1986).
42. Sénizergues, G.: Definability in weak monadic second order logic of some infinite graphs. Dagstuhl seminar on Automata theory: Infinite computations, Wadern **28** (1992) 16.
43. Sénizergues, G.: *unpublished work communicated to the author*
44. Vogler, W.: Concurrent Realization of Asynchronous Transition Systems. Proc. ATPN, LNCS **1639** (1999) 284-303.