J. Inform. Process. Cybernet. **EIK 30** (1994) 3, 143–160
(formerly: Elektron. Inform.verarb. Kybernet.)

Decidability Issues for Petri Nets – a survey[1]

By Javier Esparza[2] and Mogens Nielsen

Abstract: We survey 25 years of research on decidability issues for Petri nets. We collect results on the decidability of important properties, equivalence notions, and temporal logics.

1. Introduction

Petri nets are one of the most popular formal models for the representation and analysis of parallel processes. They are due to C.A. Petri, who introduced them in his doctoral dissertation in 1962. Some years later, and independently from Petri's work, Karp and Miller introduced vector addition systems [47], a simple mathematical structure which they used to analyse the properties of "parallel program schemata', a model for parallel computation. In their seminal paper on parallel program schemata, Karp and Miller studied some decidability issues for vector addition systems, and the topic continued to be investigated by other researchers. When Petri's ideas reached the States around 1970, it was observed that Petri nets and vector addition systems were mathematically equivalent, even though their underlying philosophical ideas were rather different (a computational approach to the physical world in Petri's view, a formal model for concurrent programming in Karp and Miller's). This gave more momentum to the research on decidability questions for Petri nets, which has since continued at a steady pace.

In the following we have collected some highlights of decidability issues for Petri nets from the 70's, 80's and 90's. As you will see, they form a nice mixture of old celebrated breakthroughs, and a recent burst of exciting new developments.

We have decided to group our selected results in three sections, covering respectively the decidability of specific *properties*, various (behavioural) *equivalences*, and finally the model checking problem for *temporal logics*.

It should be noted that we have selected our highlights also aiming at some coherence in our presentation. In other words, we do not claim to cover all important contributions on decidability for Petri nets, but still our selection covers a pretty comprehensive part of existing results, also compared to other similar surveys, e.g. [44]. We have not included results on extensions of the Petri net model. In particular, for decidability results on timed Petri nets we refer the reader to [46, 78, 79].

[1] A former version of this paper appeared in the Concurrency Column of the EATCS Bulletin 52.

[2] This paper was written while this author was at the University of Edinburgh.

2. Basic definitions

We give, in a somewhat informal way, the basic definitions on Petri nets that we need in order to state the results of this overview.

A *net* N is a triple (S, T, F), where S and T are two disjoint, finite sets, and F is a relation on $S \cup T$ such that $F \cap (S \times S) = F \cap (T \times T) = \emptyset$. The elements of S and T are called *places* and *transitions*, respectively, and the elements of F are called *arcs*. A *marking* of a net $N = (S, T, F)$ is a mapping $M: S \to \mathbb{N}$. A marking M *enables* a transition t if it marks all its input places. If t is enabled at M, then it can *occur*, and its occurrence leads to the successor marking M', which is defined for every place s as follows: a token is removed from each input place of t and a token is added to each output place of t (if a place is both input and output place of a transition, then its number of tokens does not change). This is denoted by $M \xrightarrow{t} M'$.

A *Petri net* is a pair (N, M_0), where N is a net and M_0 a marking of N, called *initial marking*. A sequence $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n$ is a *finite occurrence sequence* leading from M to M_n and we write $M_0 \xrightarrow{t_1 \ldots t_n} M_n$. A sequence $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots$ is an *infinite occurrence sequence*. An occurrence sequence is *maximal* if it is infinite, or it leads to a marking which does not enable any transition. A marking M of N is *reachable* if $M_0 \xrightarrow{\sigma} M$ for some sequence σ. The *reachability graph* of a Petri net is a labelled graph whose nodes are the reachable markings; given two reachable markings M, M', the reachability graph contains an edge from M to M' labelled by a transition t if and only if $M \xrightarrow{t} M'$.

A *labelled* net is a quadruple (S, T, F, ℓ), where (S, T, F) is a net and ℓ is a labeling function which assigns a letter of some alphabet to each transition. This function need not be injective. The reachability graph of a labelled net is defined like that of unlabelled nets; the only difference is that if $M \xrightarrow{t} M'$ then the corresponding edge from M to M' is labelled by $\ell(t)$.

Sometimes we refer to the "normal" Petri nets as *unlabelled* Petri nets. Unlabelled Petri nets can also be seen as labelled Petri nets in which the labelling function is injective.

Given a Petri net (N, M_0) and a marking M^f of N (called *final marking*), we define the *language* of (N, M_0) with respect to M^f as

$$L(N, M_0, M^f) \;=\; \{\sigma \mid M_0 \xrightarrow{\sigma} M^f\}$$

and the *trace set* of (N, M_0) as

$$T(N, M_0) \;=\; \{\sigma \mid M_0 \xrightarrow{\sigma} M \text{ for some marking } M\}$$

(sometimes the terms "language" and "terminal language" are used instead of "trace set" and "language"). Please note that the term "trace" is also used in the theory of Petri nets for elements of a free partially commutative monoid, following an idea originally due to Mazurkiewicz [60]. In order to focus our presentation, we have chosen not to include decidability results on "Mazurkiewicz traces".

Given a labelled Petri net (N, M_0), where $N = (S, T, F, \ell)$, and a marking M^f of N, the *language* of (N, M_0) with respect to M^f as

$$L(N, M_0, M^f) \;=\; \{\ell(\sigma) \mid M_0 \xrightarrow{\sigma} M^f\}$$

and the *trace set* of (N, M_0) is defined as

$$T(N, M_0) \quad = \quad \{\ell(\sigma) \mid M_0 \xrightarrow{\sigma} M \text{ for some marking } M\}.$$

We now define those classes of nets that are mentioned several times along the survey. Some others, which appear only once, are defined on the fly (or a reference is given).

A Petri net (N, M_0) is:

- *persistent* if for any two different transitions t_1, t_2 of N and any reachable marking M, if t_1 and t_2 are enabled at M, then the occurrence of one cannot disable the other.
- *conflict–free* if, for every place s with more than one output transition, every output transition of s is also one of its input transitions. All conflict–free nets are persistent; in fact, (N, M_0) is conflict–free if and only if (N, M) is persistent for every marking M of N. For most purposes, this class is equivalent to the class of nets in which each place has at most one output transition.
- *sinkless* if any cycle of N which is marked at M_0 (meaning that $M_0(s) > 0$ for some place of the cycle) remains marked at every reachable marking; i.e., cycles cannot be emptied of tokens by the occurrence of transitions.
- *normal* if, for any cycle of the net, every input transition of some place of the cycle is also an output transition of some place of the cycle. All normal nets are sinkless; in fact, normal is to sinkless what conflict–free is to persistent: (N, M_0) is normal if and only if (N, M) is sinkless for every marking M of N.
- *single–path* if it has a unique maximal occurrence sequence.
- a *BPP–net* if every transition has exactly one input place. BPP stands for Basic Parallel Process. This is a class of CCS processes defined by Christensen [11] (see also the Concurrency column of the EATCS Bulletin 51). BPPs can be given a net semantics in terms of BPP–nets. BPP–nets are also computationally equivalent to commutative context–free grammars, as defined by Huynh in [39].
- *free–choice* if, whenever an arc connects a place s to a transition t, either t is the unique output transition of s, or s is the unique input place of t.
- *1–safe* if, for every place s and every reachable marking M, $M(s) \leq 1$; i.e., no reachable marking ever puts more than one token in any place.
- *symmetric* if for every transition t there is a reverse transition t' whose occurrence "undoes" the effect of the occurrence of t, i.e., the input places of t are the output places of t', and vice versa.
- *cyclic* if M_0 can be reached from any reachable marking; i.e., it is always possible to return to the initial marking.[3]

3. Properties

In spite of the rather large expressive power of Petri nets, we shall see in this section that most of the usual properties of interest for verification purposes are decidable. On the other hand, we shall also see that they tend to have very large complexities. In fact, Petri nets are an important source of natural non–primitive recursive problems!

[3] Symmetric and cyclic nets have sometimes been called *reversible*.

So far, all decidability proofs in the net literature are carried out by reduction to the boundedness or the reachability problem: these are the only two with a direct decidability proof, and we are thus obliged to begin the section with them.

Boundedness A Petri net is *bounded* if its set of reachable markings is finite. Karp and Miller proved in [47] that boundedness is decidable. This result follows from the following characterization of the unbounded Petri nets, not difficult to prove. A Petri net is unbounded if and only if there exists a reachable marking M and a sequence of transitions σ such that $M \xrightarrow{\sigma} M + L$, where L is some non–zero marking, and the sum of markings is defined place–wise. The sequence σ is a sort of "token generator" which, starting from a marking M, leads to a bigger one $M + L$.

Karp and Miller showed how to detect "token generators" by constructing what was later called the *coverability tree*. Their algorithm turns out to be surprisingly inefficient: token generators may have non–primitive recursive length in the size of the Petri net, which implies that the construction of the coverability tree requires non–primitive recursive space! Rackoff gave a better algorithm in [67]. He showed that there always exists one token generator of "only" double exponential length in the size of the Petri net. This result leads to an algorithm which requires at most space $2^{cn \log n}$ for some constant c. This complexity is almost optimal, because Lipton proved [55] that deciding boundedness requires at least space $2^{c\sqrt{n}}$.

In [70], Rosier and Yen carried out a multiparameter analysis of the boundedness problem. They used three parameters: k, the number of places; l, the maximum number of inputs or outputs of a transition; and n, the number of transitions. They refined Rackoff's result, and gave an algorithm that works in $2^{ck \log k}(l + \log n)$ space. Among other results, they also showed that, if k is kept constant, then the problem is PSPACE–complete for $k \geq 4$.

Boundedness can be decided at a lower cost for several classes of nets. It is

- PSPACE–complete for single–path Petri nets [31];
- co–NP–complete for sinkless and normal Petri nets [37];
- polynomial (quadratic) for conflict–free Petri nets [35].

Some problems related to boundedness have also been studied. A Petri net is *k–bounded* if no reachable marking puts more than k tokens in any place (since we assume that the set of places of a net is finite, k–bounded Petri nets are bounded). The k–boundedness problem is PSPACE–complete [46].

A net N is *structurally bounded* if (N, M) is bounded for all possible markings M of N. It can be shown that a net N is structurally bounded if and only if the system of linear inequations $Y \cdot C \leq 0$, where C is the so called *incidence matrix* of N, has a solution [62]. This result implies that the structural boundedness problem can be solved in polynomial time using Linear Programming.

Reachability The reachability problem for Petri nets consists of deciding, given a Petri net (N, M_0) and a marking M of N, if M can be reached from M_0. It was soon observed by Hack [27] and Keller [48] that many other problems were recursively equivalent to the reachability problem, and so it became a central issue of net theory. In spite of important efforts, the problem remained elusive. Sacerdote and Tenney

claimed in [71] that reachability was decidable, but did not give a complete proof. This was not done until 1981 by Mayr [56]; later on, Kosaraju simplified the proof [50], basing on the ideas of [71] and [56]. The proof is very complicated. A detailed and self–contained description can be found in Reutenauer's book [69], which is devoted to it. In [51], Lambert has simplified the proof further.

Petri nets can be added *inhibitor arcs* which make the firability of a transition dependent upon the condition that a place contains no tokens. It is well known that the reachability problem for Petri nets with at least two inhibitor arcs is undecidable [27]. Very recently, Reinhardt has shown that the problem for nets with only one inhibitor arc is decidable [68].

Hack showed in [27] that several variations and subproblems of the reachability problem are in fact recursively equivalent to it:

- The submarking reachability problem. A submarking is a partially specified markings (only the number of tokens that some of the places have to contain is given). It can also be seen as the set of markings that coincide on a certain subset of places. The problem consists of deciding if some marking of this set is reachable.
- The zero reachability problem. To decide if the zero marking — the one that puts no tokens in any place — is reachable.
- The single–place zero reachability problem. To decide, given a place s, if there exists a reachable marking which does not put any token on s.

The complexity of the reachability problem has been open for many years. Lipton proved an exponential space lower bound [55], while the known algorithms require non–primitive recursive space. The situation is therefore similar to that of the boundedness problem before Rackoff's result. However, tight complexity bounds of the reachability problem are known for many net classes. Reachability is

- EXPSPACE–complete for symmetric Petri nets; this result was first announced in [8], and a proof was first given in [59];
- solvable in double exponential time for Petri nets with at most five places [34];
- PSPACE–complete for nets in which every transition has the same number of input and output places [46];
- PSPACE–complete for 1–safe Petri nets [10];
- PSPACE–complete for single–path Petri nets [31];
- NP–complete for Petri nets without cycles [74];
- NP–complete for sinkless and normal Petri nets [37];
- NP–complete for conflict–free Petri nets [32];
- NP–complete for BPP–nets [39, 20];
- polynomial for bounded conflict–free Petri nets [32];
- polynomial for marked graphs [14, 16]; a Petri net is a *marked graph* if every place has exactly one input transition and one output transition (notice that marked graphs are conflict–free);
- polynomial for live, bounded and cyclic free–choice nets [15] (liveness is defined in the next paragraph).

Liveness Hack showed in [27] that the liveness problem is recursively equivalent to the reachability problem (see also [1]), and thus decidable. Loosely speaking, a Petri net is *live* if every transition can always occur again; more precisely, if for every reachable marking M and every transition t, there exists an occurrence sequence $M \xrightarrow{\sigma} M'$ such that M' enables t. The computational complexity of the liveness problem is open, but there exist partial solutions for different classes. The liveness problem is

- PSPACE–complete for 1–safe Petri nets [10];
- co–NP–complete for free–choice nets [46];
- polynomial for bounded free–choice nets [22];
- polynomial for conflict–free Petri nets [33].

Deadlock–freedom A Petri net is *deadlock–free* if every reachable marking enables some transition. Deadlock–freedom can be easily reduced in polynomial time to the reachability problem [10]. The deadlock–freedom problem is:

- PSPACE–complete for 1–safe Petri nets, even if they are single–path [10];
- NP–complete for 1–safe free–choice Petri nets [10];
- polynomial for conflict–free Petri nets [35].

Home states and home spaces A marking of a Petri net is a *home state* if it is reachable from every reachable state. The home state problem consists in deciding, given a Petri net (N, M_0) and a reachable marking M, if M is a home state. It was shown to be decidable by Frutos [23]. The subproblem of deciding if the *initial* marking of a Petri net is a home state, which is the problem of deciding if a Petri net is cyclic, was solved much earlier by Araki and Kasami [2]. The home state problem is polynomial for live and bounded free–choice Petri nets [5, 15].

The home state problem is a special case of the home space problem. A set of markings $\mathcal{M}$ of a Petri net is a *home space* if for every reachable marking M, some marking of $\mathcal{M}$ is reachable from M. The home space problem for linear sets is decidable [24] (for the definition of linear set, see the semilinearity problem).

Promptness and strong promptness In a Petri net model of a system, the transitions represent the atomic actions that the system can execute. Some of these actions may be silent, i.e., not observable. A Petri net whose transitions are partitioned into silent and observable is *prompt* if every infinite occurrence sequence contains infinitely many observable transitions. It is *strongly prompt* if there exists a number k such that no occurrence sequence contains more than k consecutive silent transitions. Promptness is thus strongly related to the notion of divergence in process algebras. The promptness and strong promptness problems were shown to be decidable by Valk and Jantzen [80]. It follows easily from a result of [77] that the promptness problem is polynomial for live and bounded free–choice Petri nets.

Persistence The persistence problem (to decide if a given Petri net is persistent) was shown to be decidable by Grabowsky [25], Mayr [56] and Müller [65]. It is not known if the problem is primitive recursive. It is PSPACE–complete for 1–safe nets [10].

Regularity and context–freeness The regularity and context–freeness problems are in fact a collection of problems of the form:

to decide if the X of a Y–Petri net is Z

where X can be "trace set" or "language", Y can be "labelled" or "unlabelled", and Z can be "regular" or "context–free". Ginzburg and Yoeli [26] and Valk and Vidal–Naquet [81] proved independently that the regularity problem for trace sets of unlabelled Petri nets is decidable (see also [72]). Other results of [81] are that this problem is not primitive recursive, and that the regularity problem for languages of labelled Petri nets is undecidable (see also [43]).

The decidability of the context–freeness problem for trace sets of unlabelled Petri nets has been proved by Schwer [73].

Semilinearity Markings can be seen, once an arbitrary ordering of the set of places is taken, as vectors over $\mathbb{N}^n$, where n is the number of places of the net. A subset of $\mathbb{N}^n$ is *linear* if it is of the form

$$\{u + \sum_{i=1}^{p} n_i v_i \mid n_i \in \mathbb{N}\}$$

where $u, v_1, \ldots, v_p$ belong to $\mathbb{N}^n$. A subset of $\mathbb{N}^n$ is *semilinear* if it is a finite union of linear sets.

Some interesting problems are decidable for Petri nets whose set of reachable markings is semilinear. Many net subclasses, unfortunately all of them quite restrictive, are known to have semilnear reachability sets, as we shall see in section 4. It was also proved by Kleine Büning, Lettmann and Mayr that the *projection* of the set of reachable markings on a place of the net is always semilinear [49].

The semilinearity problem is the problem of deciding if the set of reachable markings of a given Petri net is semilinear. Its decidability was proved independently by Hauschildt [28] and Lambert [52].

Non–termination Much effort has been devoted to the decidability of termination in Petri nets under fairness conditions. This study was initiated by Carstensen [9], where he proved that the fair non–termination problem is undecidable. An infinite occurrence sequence is *fair* if a transition which is enabled at infinitely many markings of the sequence appears infinitely often in it. If every maximal fair occurrence sequence of a Petri net is finite (i.e., it ends at a deadlocked marking), then we say that the Petri net is guaranteed to *terminate* under the fairness assumption. The fair non–termination problem consists in deciding if a given Petri net is *not* guaranteed to terminate, i.e., if it has an infinite fair occurrence sequence.

In [36], Howell, Rosier and Yen conducted an exhaustive study of the decidability and complexity of non–termination problems for 24 different fairness notions. In particular, they studied the three notions of impartiality, justice and fairness introduced in [54]. An infinite occurrence sequence is *impartial* if every transition of the net occurs infinitely often in it; it is *just* if every transition that is enabled almost everywhere along the sequence occurs infinitely often in it; fair infinite occurrence sequences were defined

above. The just non–termination problem was left open in [36], and was later solved by Jančar [41]. The final picture is the following:

- The fair non–termination problem is complete for the first level of the analytical hierarchy. The restriction of this problem to bounded Petri nets is decidable, but non–primitive recursive.
- The impartial non–termination problem can be reduced in polynomial time to the boundedness problem, and can therefore be solved in exponential space.
- The just non–termination problem is decidable, and at least as hard as the reachability problem.

Two other interesting results of [36] concern the notions of i–fairness and ∞–fairness introduced by Best [4]. A transition t is *i–enabled* at a marking if there is an occurrence sequence no longer than i transitions which enables t. A transition is *∞–enabled* at a marking if there is an occurrence sequence, no matter how long, which enables t. An infinite occurrence sequence is *i–fair* (*∞–fair*) if every transition which is infinitely often i–enabled (∞–enabled) along the sequence occurs infinitely often in it.

Observe that 0–fairness coincides with fairness in the sense of [9] and [36]. Therefore, the 0–fair non–termination problem is undecidable. The i–fair non–termination problem turns out to be undecidable as well for every i. However, the ∞–fair non–termination problem is reducible in polynomial time to the reachability problem, and thus decidable.

4. Equivalences

As opposed to the results from the previous section, the main message from the study of decidability of behavioural equivalences of Petri nets is that almost all results are negative. However, many interesting and nontrivial subclasses of nets have been identified for which equivalences become decidable, thus shedding some light on the sources of the complexity of net behaviours.

The first undecidability result for equivalences of Petri nets dates back to the early seventies.

Marking equivalence Two Petri nets having the same set of places are said to be *marking equivalent* iff they have the same set of reachable markings.

Marking equivalence is undecidable for Petri nets. This result was proved by Hack [27], using a former result by Rabin, proving that the marking inclusion problem is undecidable. (Rabin never published his result; for a description of the proof, see [3]). The idea relies on a rather subtle way of computing functions by nets in a weak sense. It is then proved that diophantine polynomials may be computed, and then Hilbert's tenth problem is reduced to the marking inclusion/equivalence problem.

The more straightforward approach to prove undecidability, by attempting to simulate some universal computing device like Counter Machines by nets (representing counters and their values by places and their number of tokens) fails because of the inability of nets to "test for zero". But there is an obvious and simple way of semi–simulating Counter Machines by nets, simulating the counter–manipulations step by

step, *but* allowing some computational branches conditioned on a counter having the value zero to be followed in the simulation, even though the corresponding place is nonempty.

Recently, Jančar [42] came up with a set of ingenious, simple and elegant proofs of undecidability of equivalence problems following the pattern:

to prove undecidability of X–equivalence, construct two modifications of the simple nets semi–simulating a given Counter Machine, CM, satisfying that CM halts iff the two constructed nets are not X–equivalent.

(actually, the first proof of this kind can be found, to our knowledge, in [1], but Jančar has generalized the principle to other equivalences). In [42] the reader may find a simple and elegant proof of undecidability of marking equivalence (among others) for nets following exactly this pattern. It shows that the problem is undecidable even for nets with five unbounded places (i.e., places s such that for every number k there exists a reachable marking M such that $M(s) > k$).

For certain restricted classes of nets the marking equivalence problem has been shown to be decidable. For instance, it was noticed very early that for nets with a semilinear set of reachable markings the problem is decidable. This is due to a connection between semilinear sets and Pressburger arithmetic, a decidable first order theory. And many nontrivial restricted classes of Petri nets have been shown to have effectively computable semilinear reachable markings. A few examples:

- persistent [53, 25, 56, 65] and weakly persistent nets [83];
- nets with at most five places [30] (there exist nets with six places having a non–semilinear reachability set);
- regular nets [26, 81]; a Petri net is *regular* if its trace set is regular;
- cyclic nets [1];
- BPP–nets [20];

For some classes, the complexity of the problem has been determined. It is:

- solvable in $c^{n \log n}$ space for symmetric Petri nets [40];
- solvable in double exponential time for nets with at most five places [34];
- Π_2^P–complete for conflict–free Petri nets, where Π_2^P is the class of languages whose complements are in the second level of the polynomial–time hierarchy [32];
- Π_2^P–complete for sinkless and normal Petri nets [37];
- PSPACE–complete for single–path Petri nets [31].

Also, the marking equivalence problem is obviously decidable for bounded nets, which only have finitely many reachable markings. It was shown by Mayr and Meyer [58] that the problem is not primitive recursively decidable. This result has since been strengthened by McAloon [61] and Clote [13], who showed that it is complete for DTIME in the Ackerman–function. McAloon also showed that the restriction of the problem to Petri nets with at most a fixed number k of places is primitive recursive. The restriction to 1–safe Petri nets is PSPACE–complete [10].

Most – if not all – of these results also hold for the inclusion problem.

Trace and language equivalences Another bulk of results are concerned with equivalences of nets in terms of occurrence sequences. Two (labelled) Petri nets are said to be *trace equivalent* (*language equivalent*) if they have the same trace set (language). Hack proved in [27] that the problems of deciding if two labelled Petri nets are language equivalent or trace equivalent are undecidable, by means of a reduction from the marking equivalence problem. Araki and Kasami gave another proof [1] by reduction from the halting problem for Counter Machines. Stronger results are:

- trace equivalence is undecidable for labelled Petri nets with at most two unbounded places [42];
- language equivalence is undecidable for labelled Petri nets, one of them having one unbounded place and the other none [81];
- trace and language equivalence are undecidable for BPP–nets [29]. This is a remarkable result, since BPP–nets are a class with rather limited expressive power.

The trace equivalence problem of Petri nets with exactly one unbounded place is, to the best of our knowledge, open.

If we restrict ourselves to unlabelled nets, both problems become decidable. Hack [27] gave a reduction to the reachability problem, and hence today we conclude decidability.

It is well–known that any trace set of a labelled net is also a language of some labelled net, but not vice versa. This raises the interesting question, whether there exists some class of nets which distinguishes the two equivalence problems with respect to decidability.

A labelled net is said to be *deterministic up to bisimilarity* iff for all reachable markings M, if two transitions t' and t'' carrying the same label are enabled, $M \xrightarrow{t'} M'$ and $M \xrightarrow{t''} M''$, then M' and M'' are strongly bisimilar (for the definition of strong bisimulation, see below).

Clearly any unlabelled net is deterministic up to bisimilarity, but not vice versa. Furthermore, it has been shown that the property of being deterministic up to bisimilarity is decidable (reduced to the reachability problem in [42]). Christensen has shown [11] that for nets which are deterministic up to bisimilarity, trace equivalence is indeed decidable, but language equivalence is not!

Bisimulation equivalence This brings us to the question of *bisimulation equivalence* [63] for nets. We recall the definition of bisimilar markings and bisimilar nets. A relation R between the nodes of two labelled graphs is a *(strong) bisimulation* if it is symmetric, and for every element (n_1, n_2) of R, the following condition holds:

if $n_1 \xrightarrow{a} n_1'$, then there exists a node n_2' such that $n_2 \xrightarrow{a} n_2'$ and (n_1', n_2') belongs to R.

Let (N_1, M_{01}) and (N_2, M_{02}) be two Petri nets, let $\mathcal{R}_1$, $\mathcal{R}_2$ be their reachabilty graphs, and let M_1, M_2 be nodes of $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively. We say that M_1 and M_2 are (strongly) bisimilar if some bisimulation between the nodes of $\mathcal{R}_1$ and $\mathcal{R}_2$ contains the pair (M_1, M_2). The Petri nets (N_1, M_{01}) and (N_2, M_{02}) are (strongly) bisimilar if the initial markings M_{01} and M_{02} are bisimilar.

Notice that the definition of the reachability graph is different for labelled and unlabelled nets, and therefore the corresponding notions of bisimulation also differ. It is

easy to see that for unlabelled nets bisimulation and trace equivalance coincide. For labelled nets, bisimulation equivalence implies trace equivalence, but not vice versa.

Some results for this problem are:

- undecidable for labelled nets, even with only two unbounded places [42], proof following the "Jančar–pattern" [42];
- decidable for labelled BPP–nets [12];
- decidable for labelled nets, if just one of them is deterministic up to bisimulation [42];
- decidable for unlabelled nets (because trace equivalence is decidable, and bisimulation and trace equivalence coincide).

Other equivalences Hüttel has recently shown in [38] that all the equivalences of the linear/branching time hierarchy [82] below bisimulation equivalence are undecidable for Basic Parallel Processes. This result implies that they are undecidable for labelled BPP–nets. Together with the undecidability of bisimulation for labelled Petri nets, we then have that all the interleaving equivalences described so far in the literature are undecidable.

On the other hand, all problems from the linear/branching time hierarchy become decidable if we restrict ourselves to bounded nets. The complexity of these problems has been studied by several people, and some of the clever algorithms invented are parts of various constructed tools for reasoning about concurrent computations. Here we just mention the following results from [45] for 1–safe nets:

- the language and trace equivalences are both complete for EXPSPACE; interestingly, the same complexity result holds for their "true concurrency" counterparts in terms of (Pratt–)pomset equivalences;
- the bisimulation equivalence is complete for DEXPTIME; interestingly, the same complexity result holds for its "true concurrency" counterparts, like *history preserving bisimulation* [66].

5. Temporal Logics

The very positive balance of section 3 (in spite of the considerable expressive power of Petri nets, most properties are decidable), has encouraged researchers to study decidability issues for specification languages in which a large set of properties can be expressed. Mostly, these languages take the shape of a *temporal logic*. The problem of deciding, given a Petri net and a formula of a temporal logic, if the net satisfies the formula, is called the *model checking* problem.

Temporal logics can be classified into two groups: *linear time* and *branching time* logics. Linear time logics for Petri nets are usually interpreted on the set of maximal occurrence sequences[4]. Branching time logics are interpreted on the reachability graph. It is well known that some properties can be more naturally expressed in a linear time logic than in a branching time one, and vice versa.

[4] Other equivalent interpretations are also used.

The results on branching time temporal logics are mostly negative. Esparza shows in [19] that the model checking problem for (a Petri net version of) the logic UB$^-$ [17] is undecidable. This is one of the weakest branching time logics described in the literature. It has basic predicates of the form $ge(s,c)$, where s is a place of the net and c is a nonnegative constant. A predicate $ge(s,c)$ is read "the number of tokens of s is greater than or equal to c"; accordingly, it holds at a marking M if $M(s) \geq c$. The operators of the logic are the usual boolean connectives, **EX** ("existential next") and **EF** ("possibly"). A reachable marking satisfies a property **EX**ϕ if it enables some transition t and the marking reached by the occurrence of t satisfies ϕ; a marking satisfies **EF**ϕ if it enables an occurrence sequence σ such that some marking visited along the execution of σ satisfies ϕ [5].

UB$^-$ is decidable for any net whose set of reachable markings is effectively semilinear, because the model checking problem can be then reduced to the satisfiability problem of the first order logic of the natural numbers with addition, also known as Presburger Arithmetic. This includes, for instance, BPP–nets or conflict–free nets. For 1–safe conflict–free nets it is even decidable in polynomial time [18] (for the subclass of 1–safe marked graphs the same result had been proven in [6]).

The logic UB is obtained by adding the operator **EG** to UB$^-$. A marking satisfies a property **EG**ϕ if it enables some infinite occurrence sequence σ and *all* the markings visited along the execution of σ satisfy ϕ. Esparza has recently showed that UB is undecidable for BPP–nets [21]. The result can be transferred to Basic Parallel Processes.

Other branching temporal logics, such as CTL and CTL* [17], or the mu–calculus [75], are more expressive than UB, and therefore the undecidability results carry over (see also [7]).

The conclusion that can be derived is that no natural and useful branching time temporal logic for Petri nets seems to be decidable.

There has been more research on linear time temporal logics for Petri nets. To provide a unifying framework in which to survey the results we add two more basic predicates to the predicates $ge(s,c)$, and then build different temporal logics on top of them. The predicates are now interpreted on the markings of a maximal occurrence sequence. We say that an occurrence sequence satisfies a formula of a logic if its initial marking satisfies it. Finally, a Petri net satisfies a formula if at least one maximal occurrence sequences satisfy it (or, equivalently, if every maximal occurrence sequence satisfies its negation). The new predicates are:

- *first*(t), where t is a transition of the net. It holds at a marking M if the transition that succeeds M in the occurrence sequence is t.
- $en(t)$, where t is a transition of the net. It holds at a marking M if M enables t[6].

Esparza shows in [19] that the linear time μ–calculus [76] with *first*(t) as only basic predicate is decidable. If the predicates $ge(s,c)$ are added, then the logic becomes undecidable, even for BPP–nets.

[5] The logic described in [19] is in fact slightly weaker than UB$^-$. We have chosen it to better compare results.

[6] The predicate $en(t)$ can be derived as the conjunction of $ge(s,1)$ for every input place s of t. We include it as a basic predicate for convenience.

Howell and Rosier studied in [33] the logic with all three basic predicates and an eventuality operator **F**, where a marking of an occurrence sequence satisfies $\mathbf{F}\phi$ if some later marking satisfies ϕ. They showed that the model checking problem is undecidable, even for conflict–free Petri nets (notice that the fair non–termination problem can be reduced to the model checking problem for this logic: a Petri net satisfies the formula $\mathbf{GF}en(t) \Rightarrow \mathbf{GF}first(t)$, where $\mathbf{G} = \neg\mathbf{F}\neg$, if some occurrence sequence that enables t infinitely often contains t infinitely often). It follows from results of [21] that it is also undecidable for BPP–nets.

The model checking problem is, however, decidable for two fragments:

- The fragment in which negations are only applied to predicates [36].
  This fragment contains the formula $\mathbf{F}first(t)$, which expresses that t eventually occurs, but not $\mathbf{GF}first(t)$, which expresses that t is bound to occur infinitely often. The model checking problem for this fragment can be reduced in polynomial time to the reachability problem. For the class of conflict–free nets, the model checking problem is NP–complete.
- The fragment in which the composed operator **GF** is the only one allowed, and negations are only applied to predicates [41].
  This fragment contains the formula $\mathbf{GF}first(t)$, but not, for instance, the formula $\mathbf{GF}first(t) \Rightarrow \mathbf{GF}first(t')$ (after replacing the implication by its definition, a negation appears in front of an operator). Jančar [41] reduces the model checking problem for this fragment to an exponential number of instances of the reachability problem. If the formula is of the form $\mathbf{GF}\phi$, where ϕ is a boolean combination of basic predicates, then a better result exists: the model checking problem can be reduced in polynomial time to the reachability problem [36].

These results show that the presence or absence of place predicates is decisive for the decidability of a linear time logic. When they are absent, even rather powerful logics as the linear time μ–calculus are decidable. When they are present, no natural logic is decidable, only fragments in which some restrictions are applied to the use of boolean connectives.

All the decidable fragments of these logics are at least as hard as the reachability problem, which, as mentioned in the first section, is EXPSPACE–hard, and could well be non–primitive recursive. Yen has defined in [84] a class of path formulas which can be decided in exponential space. The class is of the form

$$\begin{array}{ll} \exists\, M_1, M_2, \ldots, M_k\ \exists\, \sigma_1, \sigma_2, \ldots, \sigma_k & (M_0 \xrightarrow{\sigma_1} M_1 \xrightarrow{\sigma_2} M_2 \cdots \xrightarrow{\sigma_k} M_k) \\ & \wedge\ F(M_1, \ldots, M_k, \sigma_1, \ldots, \sigma_k) \end{array}$$

where F belongs to a certain set of predicates. This set includes arbitrary conjunctions and disjunctions of both place predicates, such as

- $M(s) \geq c$ for a marking M, place s and constant c,
- $M(s) \geq M'(s) + c$, for markings M and M', place s and constant c,

and transition predicates, such as

- $\#_\sigma(t) \geq c$ for a transition sequence σ, transition t and constant c, which is true if the sequence σ contains t at least c times.

Recently, Yen, Wang and Yang have shown that deciding this class of formulas is NP–complete for sinkless nets and polynomial for conflict–free nets [85].

Acknowledgements

Thanks to David de Frutos, Matthias Jantzen, Jean–Luc Lambert, Elisabeth Pelz, Hsu–Chun Yen, and two anonymous referees for providing very useful information.

References

[1] *Araki, T. and T. Kasami:* Some Decision Problems Related to the Reachability Problem for Petri Nets. Theoretical Computer Science **3** (1977), 85–104.

[2] *Araki, T. and T. Kasami:* Decidable Problems on the Strong Connectivity of Petri Net Reachability Sets. Theoretical Computer Science **4** (1977), 99–119.

[3] *Baker, H.G.:* Rabin's Proof of the Undecidability of the Reachability Set Inclusion problem of Vector Addition Systems. M.I.T., Project MAC, CSGM **79** (1973).

[4] *Best, E.:* Fairness and Conspiracies. Information Processing Letters **18** (1984), 215–220. Addendum **19** (1984), 162.

[5] *Best, E., J. Desel and J. Esparza:* Traps characterize home states in free choice systems. Theoretical Computer Science **101** (1992), 161–176.

[6] *Best, E. and J. Esparza:* Model Checking of Persistent Petri Nets. CSL '91, LNCS **626** (1992), 15–34.

[7] *Bradfield, J.C.:* Verifying Temporal Properties of Systems. Birkhäuser, Boston, Mass. ISBN 0-8176-3625-0 (1991).

[8] *Cardoza, E., R.J. Lipton and A.R. Meyer:* Exponential Space Complete Problems for Petri Nets and Commutative Semigroups. 8th Annual Symposium on Theory of Computing (1976), 50–54.

[9] *Carstensen, H:* Decidability Questions for Fairness in Petri nets. STACS '87, LNCS **247** (1987), 396–407.

[10] *Cheng, A., J. Esparza and J. Palsberg:* Complexity Results for 1–safe Nets. 13th Conference on Foundations of Software Technology and Theoretical Computer Science, Bombay (1993). To appear in Theoretical Computer Science.

[11] *Christensen, S.:* Decidability and Decomposition in Process Algebras. Ph.D. Thesis, University of Edinburgh, CST-105-93, 1993.

[12] *Christensen, S., Y. Hirshfeld and F. Moller:* Bisimulation equivalence is decidable for basic parallel processes. CONCUR '93, LNCS **715** (1993), 143–157.

[13] *Clote, P.:* On the Finite Containment Problem for Petri Nets. Theoretical Computer Science **43** (1986), 99–105.

[14] *Commoner, F., A.W. Holt, S. Even, and A. Pnueli:* Marked Directed Graphs. Journal of Computer and System Science **5** (1971), 511–523.

[15] *Desel, J. and J. Esparza:* Reachability in cyclic extended free–choice systems. Theoretical Computer Science **114** (1993), 93–118.

[16] *Desel J. and J. Esparza:* Free Choice Petri Nets. To appear in the series Cambridge Tracts on Theoretical Computer Science (1994).

[17] *Emerson, E.A. and J. Srinivasan:* Branching Time Temporal Logic. Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, LNCS **354** (1988).

[18] *Esparza, J.:* Model Checking Using Net Unfoldings. TAPSOFT '93, LNCS **668** (1993), 613–628. An extended version will appear in Science of Computer Programming (1994).

[19] *Esparza, J.:* On the decidability of model checking for several μ–calculi and Petri nets. CAAP'94, LNCS **787** (1994), 115–129.

[20] *Esparza, J.:* On the uniform word problem for commutative context–free grammars. Submitted (1993).

[21] *Esparza, J.:* Unpublished manuscript (1994).

[22] *Esparza, J. and M. Silva:* A polynomial–time algorithm to decide liveness of bounded free choice nets. Theoretical Computer Science **102** (1992), 185–205.

[23] *Frutos Escrig, D.:* Decidability of Home States in Place Transition Systems. Internal Report. Dpto. Informatica y Automatica. Univ. Complutense de Madrid (1986).

[24] *Frutos Escrig, D. and C. Johnen:* Decidability of Home Space Property. Report LRI-503, Univ. de Paris–Sud, Centre d'Orsay, Laboratoire de Recherche en Informatique (1989).

[25] *Grabowsky, J.:* The Decidability of Persistemce for Vector Addition Systems. Information Processing Letters **11** (1980) 1, 20–23.

[26] *Ginzburg, A. and M. Yoeli:* Vector Addition Systems and Regular Languages. Journal of Computer and System Sciences **20** (1980), 277–284.

[27] *Hack, M.H.T.:* Decidability Questions for Petri Nets. Ph. D. Thesis, M.I.T. (1976).

[28] *Hauschildt, D.:* Semilinearity of the Reachability Set is Decidable for Petri Nets. Report FBI-HH-B-146/90, University of Hamburg (1990).

[29] *Hirshfeld, Y.:* Petri Nets and the Equivalence Problem. To appear in the proceedings of CSL '93.

[30] *Hopcroft, J. and J. Pansiot:* On the reachability problem for 5–dimensional vector addition systems. Theoretical Computer Science **8** (1979), 135–159.

[31] *Howell, R., P. Jančar and L. Rosier:* Completeness results for Single–Path Petri Nets. Information and Computation **106** (1993), 253–265.

[32] *Howell, R. and L. Rosier:* Completeness results for conflict–free vector replacement systems. Journal of Computer and System Sciences **37** (1988), 349–366.

[33] *Howell, R. and L. Rosier:* Problems concerning fairness and temporal logic for conflict–free Petri nets. Theoretical Computer science **64** (1989), 305–329.

[34] *Howell, R., L. Rosier, D. Huynh and H. Yen:* Some complexity bounds for problems concerning finite and 2–dimensional vector addition systems with states. Theoretical Computer Science **46** (1986), 107–140.

[35] *Howell, R., L. Rosier and H. Yen:* An $O(n^{1.5})$ algorithm to decide boundedness for conflict–free vector replacement systems. Information Processing Letters **25** (1987), 27–33.

[36] *Howell, R., L. Rosier and H. Yen:* A taxonomy of fairness and temporal logic problems for Petri nets. Theoretical Computer Science **82** (1991), 341–372.

[37] *Howell, R., L. Rosier and H. Yen:* Normal and Sinkless Petri Nets. Journal of Computer and System Sciences **46** (1993), 1–26.

[38] *Hüttel, H.:* Undecidable Equivalences for Basic Parallel Processes. 13th Conference on Foundations of Software Technology and Theoretical Computer Science, Bombay (1993).

[39] *Huynh, D.T.:* Commutative Grammars: The Complexity of Uniform Word Problems. Information and Control **57** (1983), 21–39.

[40] *Huynh, D.T.:* The Complexity of the Equivalence Problem for Commutative Semigroups and Symmetric Vector Addition Systems. Proceedings of the 17th Annual ACM Symposium on Theory of Computing (1985), 405–412.

[41] *Jančar, P.:* Decidability of a Temporal Logic Problem for Petri Nets. Theoretical Computer Science **74** (1990), 71–93.

[42] *Jančar, P.:* Decidability Questions for Bisimilarity of Petri Nets and Some Related Problems. STACS '94, LNCS **775** (1994), 581–594.

[43] *Jantzen, M.:* Eigenschaften von Petrinetzsprachen. Technical Report IFI-HH-B-64/79, FB–Informatik, Univ. Hamburg (1979).

[44] *Jantzen, M.:* Complexity of Place/Transition Nets. Advances in Petri Nets 86, LNCS **254** (1986), 413–435.

[45] *Jategaonkar, L. and A. Meyer:* Deciding true concurrency equivalences on finite safe nets. ICALP '93, LNCS **700** (1993), 519–531.

[46] *Jones, N.D., L.H. Landweber and Y.E. Lien:* Complexity of Some Problems in Petri Nets. Theoretical Computer Science **4** (1977), 277–299.

[47] *Karp, R.M. and R.E. Miller:* Parallel Program Schemata. Journal of Computer and system Sciences **3** (1969), 147–195.

[48] *Keller, R.M.:* A Fundamental theorem of Asynchronous Parallel Computation. Parallel Processing, LNCS **24** (1975), 102–112.

[49] *Kleine Büning, H., T. Lettmann and E.W. Mayr:* Projections of Vector Addition System Reachability Sets are Semilinear. Theoretical Computer Science **64** (1989), 343–350.

[50] *Kosaraju, S.R.:* Decidability of Reachability in Vector Addition Systems. 14th Annual ACM Symposium on Theory of Computing, San Francisco (1982), 267–281.

[51] *Lambert, J.L.:* A structure to decide reachability in Petri nets. Theoretical Computer Science **99** (1992), 79–104.

[52] *Lambert, J.L.:* Vector addition systems and semi–linearity. To appear in the SIAM Journal of Computing (1994).

[53] *Landweber, L.H. and E.L. Robertson:* Properties of conflict–free and persistent Petri nets. Journal of the ACM **25** (1976) 3, 352–364.

[54] *Lehman, D., A. Pnueli and J. Stavi:* Impartiality, justice, and fairness: the ethics of concurrent termination. ICALP '81, LNCS **115** (1981), 264–277.

[55] *Lipton, R.J.:* The Reachability Problem Requires Exponential space. Department of Computer Science, Research Report 62, Yale University (1976).

[56] *Mayr, E.W.:* Persistence of Vector Replacement Systems is Decidable. Acta Informatica **15** (1981), 309–318.

[57] *Mayr, E.W.:* An Algorithm for the General Petri net Reachability Problem. SIAM Journal of Computing **13** (1984) 3, 441–459.

[58] *Mayr, E.W. and A.R. Meyer:* The Complexity of the Finite Containment Problem for Petri Nets. Journal of the ACM **28** (1981) 3, 561–576.

[59] *Mayr, E.W. and A.R. Meyer:* The Complexity of the Word Problems for Commutative Semigroups and Polynomial Ideals. Advances in Mathematics **46** (1982), 305–329.

[60] *Mazurkiewicz, A.:* Basic Notions of Trace Theory. Linear Time, Branching Time and Partial Orders in Logics and Models for Concurrency, LNCS **354** (1988), 285–363.

[61] *McAloon, K.:* Petri Nets and Large Finite Sets. Theoretical Computer Science **32** (1984), 173–183.

[62] *Memmi, G. and G. Roucairol:* Linear Algebra in Net Theory. Net Theory and Applications, LNCS **84** (1980), 213–223.

[63] *Milner, R.:* Communication and Concurrency. Prentice–Hall (1989).

[64] *Minsky, M.:* Computation: Finite and Infinite Machines. Prentice–Hall (1967).

[65] *Müller, H.:* On the Reachability Problem for Persistent Vector Replacement Systems. Computing Supplements **3** (1981), 89–104.

[66] *Rabinowich, A. and B. Trakhtenbrot:* Behaviour Structures and nets of processes. Fundamenta Informaticae **11** (1988) 4, 357–404.

[67] *Rackoff, C.:* The covering and boundedness problem for vector addition systems. Theoretical Computer Science **6** (1978), 223–231.

[68] *Reinhardt, K.:* Das Erreichbarkeitproblem bei Petrinetzen mit inhibitorischen Kanten. Unpublished manuscript (1994)

[69] *Reutenauer, C.:* The Mathematics of Petri Nets. Masson (1988).

[70] *Rosier, L.E. and H.C. Yen:* A Multiparameter Analysis of the Boundedness Problem for Vector Addition Systems. Journal of Computer and System Sciences **32** (1986), 105–135.

[71] *Sacerdote, G.S. and R.L. Tenney:* The Decidability of the Reachability Problem for Vector Addition Systems. 9th Annual Symposium on Theory of Computing, Boulder (1977), 61–76.

[72] *Schwer, S.:* On the rationality of Petri net languages. Information Processing Letters **22** (1986), 145–146.

[73] *Schwer, S.:* The context–freeness of the languages associated with vector addition systems is decidable. Theoretical Computer Science **98** (1992), 199–247.

[74] *Stewart, I.A.:* On the reachability problem for some classes of Petri nets. Research Report, University of Newcastle upon Tyne (1992). To appear in Fundamenta Informaticae.

[75] *Stirling, C.:* Modal and Temporal Logics. In Handbook of Logic in Computer Science, Oxford University Press (1991).

[76] *Stirling, C. and D. Walker:* CCS, liveness, and local model checking in the linear time mu–calculus. Workshop on automatic verification methods for finite state systems, LNCS **407** (1990), 166–178.

[77] P.S. Thiagarajan and K. Voss. A Fresh Look at Free Choice Nets. Information and Control **61** (1984) 2, 85–113.

[78] *Valero Ruiz, V., D. Frutos Escrig and F. Cuartero Gomez:* Simulation of Timed Petri Nets by Ordinary Petri Nets and Applications to Decidability of the Timed Reachability problem and other Related Problems. Petri Nets and Perfomance Models 91, 154–163. IEEE Computer Society Press (1991).

[79] *Valero Ruiz, V., D. Frutos Escrig and F. Cuartero Gomez:* Decidability of the Strict Reachability problem for Timed Petri Nets with Rational and Real Durations. Petri Nets and Perfomance Models 93. IEEE Computer Society Press (1993).

[80] *Valk, R. and M. Jantzen:* The Residue of Vector Sets with Applications to Decidabilty problems in Petri Nets. Acta Informatica **21** (1985), 643–674.

[81] *Valk, R. and G. Vidal–Naquet:* Petri Nets and regular Languages. Journal of Computer and system Sciences **23** (1981) 3, 299–325.

[82] *van Glabbeek, R.J.:* The linear time – branching time spectrum. CONCUR '90, LNCS **458** (1990), 278–297.

[83] *Yamasaki, H.:* On weak persistency of Petri nets. Information Processing Letters **13** (1981) 3, 94–97.

[84] *Yen, H.C.:* A Unified Approach for Deciding the Existence of certain Petri Net Paths. Information and Computation **96** (1992), 119–137.

[85] *Yen, H.C., B.Y. Wang and M.S. Yang:* A Unified Approach for Reasoning About Conflict–Free Petri Nets. Application and Theory of Petri Nets 1993, LNCS **691** (1993), 513–531.

(Received: April 18, 1994; revised version: November 24, 1994)

Authors' addresses:

Javier Esparza
Institut für Informatik
Technische Universität
Arcisstr. 21
D–80290 München
Germany
e–mail: esparza@informatik.tu-muenchen.de

Mogens Nielsen
BRICS
Computer Science Department
Aarhus University
Ny Munkegade, Bldg. 540
DK–8000 Aarhus C
Denmark
e–mail: mnielsen@daimi.aau.dk