Polynomial-time Many-one Reductions for Petri Nets

Catherine Dufourd and Alain Finkel

LSV, CNRS URA 2236; ENS de Cachan, 61 av. du Pdt. Wilson
94235 Cachan Cedex, FRANCE.
{Catherine.DUFOURD, Alain.FINKEL}@lsv.ens-cachan.fr

Abstract. We apply to Petri net theory the technique of polynomial-time many-one reductions. We study boundedness, reachability, deadlock, liveness problems and some of their variations. We derive three main results. Firstly, we highlight the power of expression of reachability which can polynomially give evidence of unboundedness. Secondly, we prove that reachability and deadlock are polynomially-time equivalent; this improves the known recursive reduction and it complements the result of Cheng and al. [4]. Moreover, we show the polynomial equivalence of liveness and t-liveness. Hence, we regroup the problems in three main classes: boundedness, reachability and liveness. Finally, we give an upper bound on the boundedness for post self-modified nets: $2^{O(size(N)^2 * \log size(N))}$. This improves a decidability result of Valk [18].

Key words: Petri net theory; Complexity Theory; Program Verification; Equivalences.

1 Introduction

The *boundedness*, the *reachability*, the *deadlock*, the *t-liveness* and the *liveness* problems are among the main problems studied in Petri nets. Solving these problems requires huge space and time resources. For boundedness, Lipton [13] proved that a lower space-bound is $2^{c \cdot \sqrt{|N|}}$, improved with $2^{c \cdot |N|}$ by Bouziane [2] (where c is some constant and $|N|$ is the size of the input net); Rackoff [17] proved that an upper space-bound for this problem is $2^{O(|N| * \log |N|)}$. For reachability, decidability has been proved by Mayr [14] and Kosaraju [12]; Cardoza, Lipton, Mayr and Meyer [3,15] established that this problem is Expspace-hard. However, until now, it is not known whether the reachability, the deadlock and the liveness problems are primitive recursive or not. In this paper, our aim is to compare these problems, to regroup similar problems into classes and to order these classes.

We use *polynomial-time many-one reductions* [9]. The idea is to take one instance of a problem A and to polynomially transform it into one instance of another problem B. The problem B is seen as an *oracle* used to solve the problem A. In the literature, we often find two other kinds of reductions: *polynomial-time Turing reductions* which allow to consult the oracle not only once, but a polynomial number of times and *recursive reductions*. We obtain two sorts of results. Firstly, we prove three main theorems:

- Boundedness is polynomially reducible to reachability,
- Reachability and deadlock are polynomially equivalent,
- Liveness and t-liveness are polynomially equivalent.

For instance, we show that a Petri net N is unbounded if and only if a marking M_N is reachable in the net $\widehat{N}$ which is polynomially constructed from N. Let us note that our second theorem strengthens a recent result of Cheng, Esparza and Palsberg [4] who showed that reachability is polynomially reducible to deadlock. Secondly, we establish a strong relation between Petri nets and *Post Self-Modifying nets* (PSM-nets) on the boundedness problem. Post self-modifying nets, defined by Valk [18], are extended Petri nets in which a transition may add a "dynamic number" of tokens (which is an affine function, with a specific form, of the current marking) in its output places. Valk has proven that the boundedness problem is decidable for post self-modifying nets. Here, we improve his decidability result by giving $2^{O(|N|^2 * \log |N|)}$ as an upper space-bound. Moreover, this upper bound is not so far from the lower bound $\Omega(2^{|N|})$.

There are four advantages in grouping problems together. Firstly, even if we still do not know the exact complexity of reachability and deadlock, it is instructive to know that they have the same complexity, modulo a polynomial transformation. Secondly, when we know that seven problems are polynomially equivalent, as for the ones of the reachability class, we may focus our attention on only one of these problems, to produce a good implementation of an algorithm solving it; this unique program may be used for solving the sixth other problems. Thirdly, the obtained results confirm our intuition about the hardness of problems in Petri nets. Basically we obtain the following order:

$$\text{Boundedness} \leq \text{Reachability} \equiv \text{Deadlock} \leq \text{Liveness}$$

Fourthly, we obtain a new complexity result in using the equivalence between boundedness for Petri nets and boundedness for post self-modifying nets.

In the next section, we give the basic definitions of Petri nets and polynomial-time reductions; then we make an overview of the known many-one polynomial-time reductions. In section 3, we reduce boundedness to reachability. In section 4, we prove that reachability is polynomially equivalent to deadlock; moreover, both are polynomially equivalent to reachability and deadlock for normalized Petri nets (for which valuations over arcs and initial marking are upper-bounded with 1). In section 5, we show that liveness is equivalent to t-liveness. In section 6, we prove that boundedness for Petri nets and boundedness for post self-modifying nets are polynomially equivalent; we deduce from there the upper-bound on the boundedness problem for PSM-nets. We conclude in section 7.

2 Petri nets and polynomial-time reductions

Let $\mathbb{N}$ be the set of nonnegative integers and let $\mathbb{N}^k$ $(k \geq 1)$ be the set of k-dimensional column vectors of elements in $\mathbb{N}$. Let $X \in \mathbb{N}^k$, $X(i)$ $(1 \leq i \leq k)$ is the i^{th} component of X. Let $X, Y \in \mathbb{N}^k$, we have $X \prec Y$ iff the two conditions hold : (a) $X(i) \leq Y(i)$ $(1 \leq i \leq k)$ and (b) $\exists j,\ 1 \leq j \leq k,\ s.t.\ X(j) < Y(j)$. Let Σ be a finite alphabet, Σ^* is the set of all finite words (or sequences) over Σ. We note $|S|$, the cardinal of a finite set S. We note $|N|$, the size of a Petri net N.

2.1 Petri nets, properties and complexity

A Petri net is a 4-tuple $N =< P, T, F, M_0 >$ where P is a finite set of places, T is a finite set of transitions with $P \cap T = \emptyset$, $F : (P \times T) \cup (T \times P) \longrightarrow \mathbb{N}$ is a flow function and $M_0 \in \mathbb{N}^{|P|}$ is the initial marking. A Petri net is *normalized* or *ordinary* if F is a function into $\{0, 1\}$ and M_0 is a function into $\{0, 1\}^{|P|}$. A transition t is *firable* from a marking $M \in \mathbb{N}^{|P|}$, written $M \xrightarrow{t}$, if for every place p, we have $F(p, t) \leq M(p)$. Firing t from M leads to a new marking M', written $M \xrightarrow{t} M'$, defined as follows : for every place p, we have $M'(p) = M(p) - F(p, t) + F(t, p)$. A marking M' is *reachable* from M, written $M \xrightarrow{*} M'$, if there exists a sequence $\sigma \in T^*$ such that $M \xrightarrow{\sigma} M'$. A marking is *dead* if no transition is firable from it. The *reachability set* of N, denoted $RS(N)$, contains all the markings reachable from M_0. A Petri net is *unbounded* if its reachability set is infinite. A transition t is *quasi-live* from M if it is firable from a marking M' with $M \xrightarrow{*} M'$. A transition $t \in T$ is *live* if it is quasi-live from any marking in $RS(N)$. A Petri net is *live* if all transitions are live.

Definition 1. Given a Petri net $N =< P, T, F, M_0 >$, $t \in T$ and $M \in \mathbb{N}^{|P|}$:
- The Boundedness Problem (BP) is to determine whether N is bounded or not.
- The Reachability Problem (RP) is to determine whether $M \in RS(N)$ or not.
- The Deadlock Problem (DP) is to determine whether $RS(N)$ contains a dead-marking or not.
- The t-Liveness Problem (t-LP) is to determine whether the transition t is live or not.
- The Liveness Problem (LP) is to determine whether N is live or not.

These problems have been widely studied. They are all decidable [11,8,12,14,7], but intractable in practice. A lower space-bound for the RP and BP is $2^{c \cdot \sqrt{|N|}}$ [13]. Reachability is EXPSPACE-hard [3,15], but we don't know yet if the *RP* is primitive recursive or not. There exists a family of bounded Petri nets such that every net N of the family has a reachability set with a non-primitive recursive size in $|N|$ [10]. An upper space-bound for deciding the *BP* is $2^{O(|N| * \log |N|)}$ [17]. This bound comes from the following theorem:

Theorem 2. *[11,17] A Petri net $N =< P, T, F, M_0 >$ is unbounded if and only if there exists two sequences $\sigma_1, \sigma_2 \in T^*$ such as $M_0 \xrightarrow{\sigma_1} M_1 \xrightarrow{\sigma_2} M_2$ with*

$M_1 \prec M_2$. *The net is unbounded if and only if there exists such an execution of length less than a double exponential in the size of* N.

If we talk about complexity, we need to determine what is the size of a Petri net. The representation we have chosen which is slightly different from the one in [20] commonly used. Let V be the greatest integer found over the flow function and the initial marking. We propose to encode the flow function of a Petri net with two matrices of size $(|P| \times |T|)$ containing $\Theta(\log V)$ bits: one matrix for *input* arcs and an other for *output* arcs. A Petri net is encoded with a sequence of bits giving the number of places, the number of transitions, the size of V, the flow function and finally the initial marking. The total size belongs to:

$$\Theta(\log|P| + \log|T| + \log\log V + 2*|P|*|T|*\log V + |P|*\log V) = \Theta(|P|*|T|*\log V)$$

2.2 Known polynomial-time reductions for Petri nets

Reductions [9] are used to compare different problems for which, most of the time, no efficient algorithm is known. We manipulate decision problems which are problems requiring *Yes* or *No* as output. We ask questions of the kind : *"Does Petri net N possess the property P or not ?"*. The net given in input is called the *instance* of problem P. Most of the time, instances of our problems are Petri nets but it may happen that we need to specify a marking (as for the RP) or a transition (as for the t-LP). We note I_P the set of instances associated to problem P. We say that P is *many-one polynomial-time reducible* to Q, written $P \leq_{poly} Q$, if we can exhibit a polynomial-time computable function f such as $I_P \in P \;\Leftrightarrow\; f(I_p) \in Q$. We say *many-one* because the function f is not necessarily a bijection. Sometimes, we have to take the complement of usual problems: for instance, we talk about the reduction from reachability to not-liveness and not to liveness.

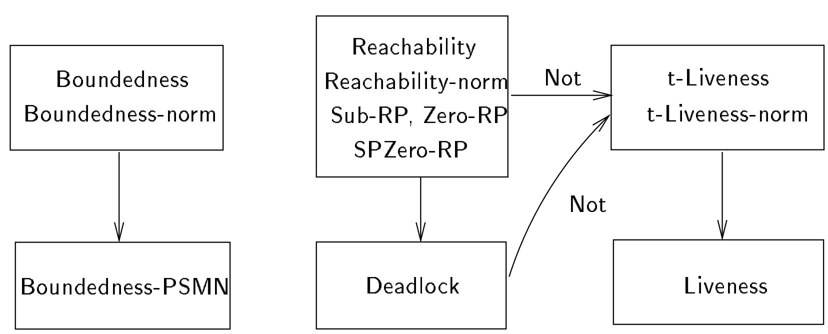

Fig. 1. Summary of known polynomial-time many-one reductions.

We give in the current section an overview of known many-one polynomial-time reductions focusing on the BP, RP, DP, LP and t-LP and some of their variations. The Fig. 1 summarizes the relation between the problems with a diagram. All problems put in a same box are equivalent. An arrow from a box to another indicates the existence of a reduction from the first class to the other. An arc labeled with "not" refers to a reduction to the complement of a problem. The boundedness problem for post self-modifying nets is written Boundedness-PSMN (the definition of PSM-nets is recalled in section 6).

Normalization: The normalization proposed in [6] is performed in quadratic time and preserves boundedness, reachability and t-liveness. We add the suffix *-norm* to design the classical problems over normalized, or ordinary, Petri nets. We have BP equivalent to BP-norm, RP equivalent to RP-norm and t-LP equivalent to t-LP-norm.

Reachability: Many polynomial reductions were given by Hack [8], [16] about reachability properties. Hack pointed out three problems equivalent to the RP. The Submarking Reachability Problem (Sub-RP) over $< N, M' >$, where M' is a marking over the subset $P' \subseteq P$, is to decide whether there exists a marking M reachable such that for all $p \in P',\ M(p) = M'(p)$. The Zero-Reachability Problem (Zero-RP) over $< N >$ is to decide whether there exists a reachable marking in which all the places are empty. The Single-Place Zero-Reachability (SPZero-RP) over $< N, p >$ is to decide whether there exists a reachable marking for which place p is empty. Cheng and al. [4] showed that reachability is polynomially reducible to deadlock.

Liveness: Reachability is polynomially reducible to not-liveness [16]. The other sense of the reduction is known recursive but we do not know a polynomial reduction. More recently, Cheng and al. [4] showed that the deadlock problem is polynomially reducible to not-liveness. But as for RP, the other sense is not known. Liveness appears to be a very expressive property. Hack [8] mentions a reduction from t-LP to LP performed in almost linear-time.

3 From unboundedness to reachability

Let us compare the current state of knowledge about boundedness and reachability. Firstly, about complexity, we know an upper space-bound for solving boundedness [17] but we still do not know if reachability is primitive recursive or not. Moreover, this last question remains one of the hardest open questions in Petri net theory. Secondly, we know that if we increase the power of Petri nets a little bit then reachability becomes undecidable while boundedness seems more resistant. An illustrative example is the class of the post self-modifying nets for which boundedness is decidable but not reachability (see section 6). Reachability seems to be a stronger property than boundedness because BP is in EXPSPACE and RP is EXPSPACE-hard; in the current section, we explicitly give the reduc-

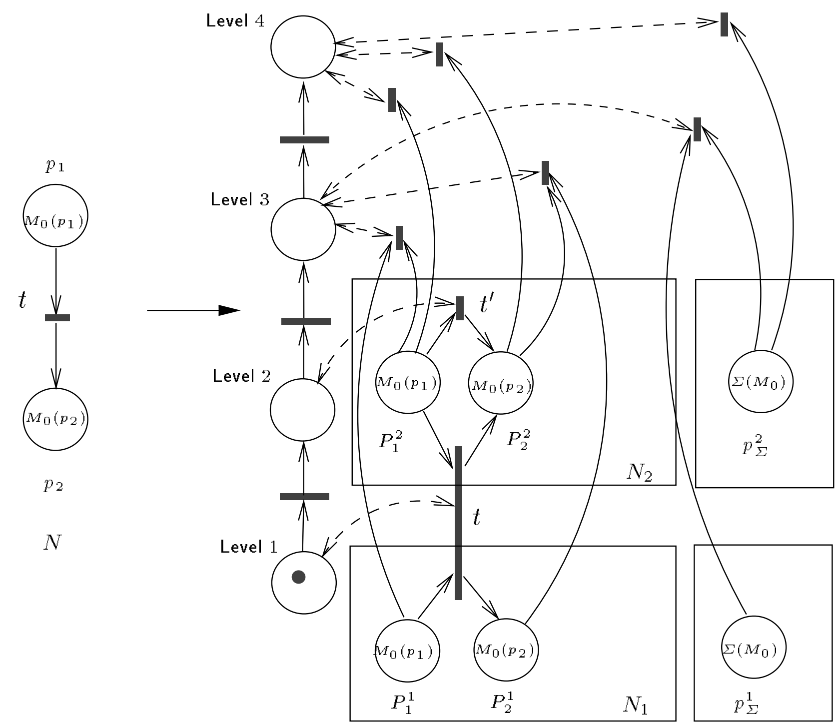

Fig. 2. Reduction from boundedness to reachability.

tion from BP to RP. The other sense, reachability to unboundedness, is probably false otherwise we would obtain a surprising upper bound complexity on solving reachability.

Theorem 3. *Unboundedness is polynomially reducible to reachability*

Proof : Let $N = < P, T, F, M_0 >$ be a Petri net. Recall that N is not bounded if and only if there exists an execution of N, $M_0 \overset{\sigma_1}{\to} M' \overset{\sigma_2}{\to} M''$, such that $M' \prec M''$ [11]. The difference $M_d = M'' - M'$ is a nonnegative vector, with at least one strictly positive component. Our strategy is to look for such a marking M_d. But we want to detect M_d through reachability, by asking whether a specific marking is reachable, and this implies that we need to characterize M_d in a standard way. Let us suppose that we add a summing-place that contains at any step the sum over all the places (a summing-place can easily be implemented in a Petri net by adding to each transition an arc labeled with the total effect of the transition). The marking M_d is certainly strictly greater than marking with 0 in all the places except 1 in the summing-place. We use this characterization for the final question of the reduction. Let us explain our reduction with the help of Fig. 2. We build $\widehat{N} = < \widehat{P}, \widehat{T}, \widehat{F}, \widehat{M_0} >$ as follows:

– Make two copies of N in $N_1 =< P_1, T_1, F_1, M_{01} >$ and $N_2 =< P_2, T_2, F_2, M_{02} >$ with $M_0 = M_{01} = M_{02}$;
– Add two summing-places. At first, p^1_Σ contains the sum over all the places of N_1 and p^2_Σ contains the sum over all the places of N_2;
– Each transition $t \in T_2$ is duplicated leading to a new transition t' in T_2 (note that now N_2 is not anymore an exact copy of N);
– Make the fusion of N_1 and N_2 over pairs (t_1, t_2) where $t_1 \in N_1$ and $t_2 \in N_2$ are copies of the same original transition in N;
– Add four levels of control, which are activated successively during an execution. Levels are managed with permission-places labeled explicitly in the picture. Control is first given at level 1 and moves as follows: **Level** 1 $\rightarrow$ **level** 2 $\rightarrow$ **level** 3 $\rightarrow$ **level** 4. The dashed arcs link a permission-place to the set of transitions that it allows to be fired:
  - **Level 1** allows the two nets N_1 and N_2 to fire the original transitions together;
  - **Level 2** allows only N_2 to continue to fire the original transition while N_1 and its summing-place are frozen;
  - **Level 3** allows simultaneous emptying of two associated places (p_1, p_2) where $p_1 \in P_1 \cup \{p^1_\Sigma\}$ and $p_2 \in P_2 \cup \{p^2_\Sigma\}$ is its corresponding place;
  - **Level 4** allows to empty places of $N_2 \cup \{p^2_\Sigma\}$ only.

Correctness: N is unbounded if and only if $M_r = (0, 0, 0, 1, 0 \cdots 0, 0, 0 \cdots 0, 1)$ is reachable in $\widehat{N}$. The first four positions in M_r are related to the four levels. The last position in M_r is related to summing-place p^2_Σ. The other positions, all equal to 0, are related to the remaining places of N_1 and N_2. Note that M_r is a marking at level 4 ($M_r(4) = 1$). By construction, in $\widehat{N}$, at any time M' in N_1 and M'' in N_2 are two markings appearing along an execution of N. The only way to empty correctly P_1 and p^1_Σ and to keep at least one token in p^2_Σ is to have $M' \prec M''$; this happens if and only if N is unbounded. Finally, level 4 allows to clean up the remaining places in order to exactly reach M_r when N is unbounded.

Complexity: The net $\widehat{N}$ contains $O(|P|)$ places and $O(|P| + |T|)$ transitions. The greatest value in $\widehat{N}$ is $(|P| * V)$, because of the summing-places (recall that V is the greatest value of N). The total size is thus $O(|P| * (|P| + |T|) * \log(|P| * V))$ and the construction is linear on this size. We conclude that the time-complexity of the reduction is $O(\log |P| * |N|^2)$ and this concludes the proof. ■

4 Polynomial equivalence of reachability and deadlock

Reachability and deadlock are decidable and thus recursively equivalent [4]. In the current section, we prove that reachability, deadlock, reachability for normalized Petri nets and deadlock for normalized Petri nets are polynomially equivalent. Recall that a Petri net is normalized if the flow function returns an integer in $\{0, 1\}$ and the initial marking belongs to $\{0, 1\}^{|P|}$. The reachability set however may be infinite and thus, normalized Petri nets should not be confused with 1-safe nets for which any reachable marking contains only 0 or 1 as values. Normalization provides a simpler representation of Petri nets; in this sense, it is interesting to notice that studying RP or DP may be restricted to this class modulo a polynomial transformation. Our proofs use some known results but we explain in detail the main reduction "from deadlock to reachability".

Proposition 4. *Reachability, deadlock, reachability for normalized PN and deadlock for normalized PN are polynomially-time equivalent.*

Proof: We prove that RP $\leq_{poly}$ RP-*norm* $\leq_{poly}$ DP-*norm* $\leq_{poly}$ DP $\leq_{poly}$ RP.
$\hookrightarrow$ The first reduction, from RP to RP-*norm*, is true by the normalization in [6] which is performed in quadratic time and preserves reachability. To make an efficient normalization, the main idea is to use the binary strings encoding integers appearing in F and M_0, instead of using their values.
$\hookrightarrow$ The second reduction, from RP-*norm* to DP-*norm*, is true from the reduction in Cheng and al. [4]. The main idea of the reduction is the following: let the original net run with dummy self-loop transitions. At any time, the current marking can be tested. The expected marking (which is part of the input) is subtracted from current marking. If the current marking was the expected one, the dummy transitions are not firable anymore and this leads to a deadlock. However, to preserve the normalization, we need to perform a pre-normalization over the expected marking.
$\hookrightarrow$ The third reduction, from DP-*norm* to DP, is trivial.
$\hookrightarrow$ We explain in detail the fourth reduction, from DP to RP. A natural Turing reduction is to list all the partial dead markings and to ask for each of them, whether it is reachable or not. However, there exists an exponential number of dead markings and this strategy is not polynomial.

Construction, from DP to RP. Let $N =< P, T, F, M_0 >$ be a Petri net. A deadlock M_d in N is a reachable marking allowing no transition to be fired. This means that for every transition t, there exists a place p such that $M_d(p) < F(p,t)$. It is not necessary to describe the marking M_d over all the places; a subset of places is sufficient. The main idea is to guess a partial marking, to validate it as a good candidate for a deadlock, to let run the original net and finally to compare M_d guessed with the current marking M of the original net. For that, M_d is subtracted from M (token by token). If the markings are the same, 0 is reachable in the chosen places for M and M_d.

Fig. 3 gives the general skeleton of the reduction. We construct a net $\widehat{N}$ with 4 levels of control. Each level controls a specific subnet, isolated in a box. However, two boxes may have common transitions, and this is illustrated with non-oriented dashed arcs. Control is given first at level 1 and moves as follows: **level** 1 $\rightarrow$ **level** 2 $\rightarrow$ **level** 3 $\rightarrow$ **level** 4. We explain in detail the four levels.

- At Level 1, a subset of places $P' \subseteq P$ is chosen, and a marking M_d is guessed over P'. In Fig. 3, the guessed marking appears in the central box. If the place p is chosen, then a place Yes_p is marked; otherwise, a place No_p is marked.

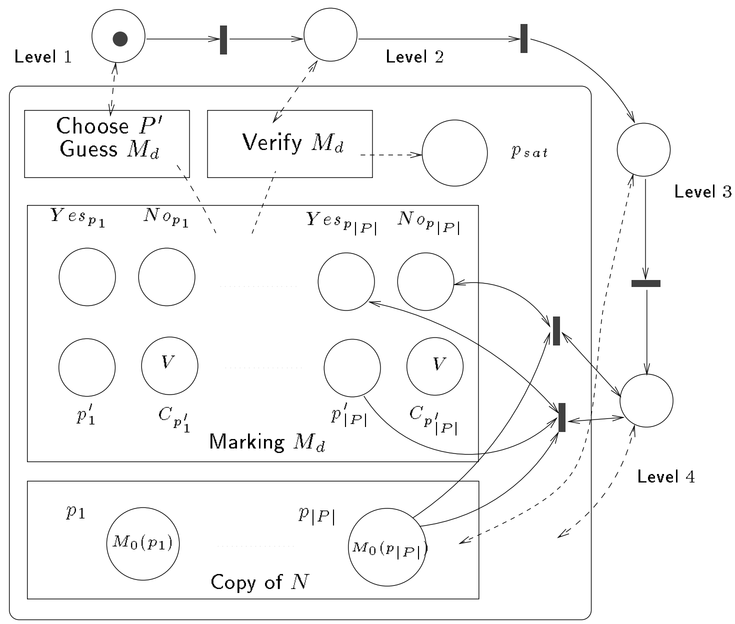

Fig. 3. Reduction from deadlock to reachability.

For each original place $p \in P'$, the aimed $M_d(p)$ is stored into a place labeled with p'. Fig. 4 gives the details of the implementation for place p. An $M_d(p)$ cannot be greater than V, where V is the greatest valuation of the original net. To guess $M_d(p)$, i.e. the content of p', we use a *complementary place*, labeled with $C_{p'}$. Places of kind p' are initialized with 0, and complementary places with V. At any time, the sum over a place and its complementary place is the constant V.

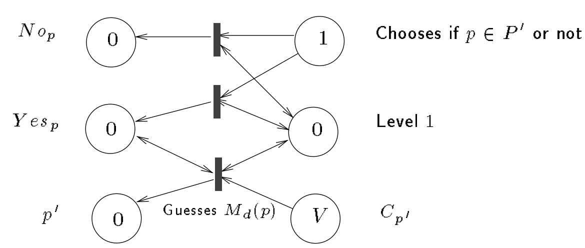

Fig. 4. From deadlock to reachability : to choose p and to guess $M_d(p)$ into p'.

- At level 2, the net verifies that M_d is a *good candidate*: M_d must under-evaluate for any transition the number of tokens required by at least one input place. If the condition holds, then the place p_{sat} is marked. To confirm that M_d is a good candidate, we verify the following boolean equation :

$$\wedge_{t\in T}\ \vee_{p\in {}^\bullet t}\,[(M_d(p) < F(p,t)) \wedge (p \in P')]$$

Condition $M_d(p) < F(p,t)$ is easily implemented using the complementary places: in fact, if p', i.e. $M_d(p)$, contains less than $F(p,t)$ tokens then its complementary place contains at least $V - F(p,t) + 1$ tokens. The condition $p \in P'$ is verified by using the Yes_p places. We illustrate the construction in Fig. 5, where we focus on transition t_1 which has here as input places: p_1 and an arbitrary p_i. If the guessed marking is dead for t_1, then a place "Dead for t_1" is marked. The same implementation is done for all the transitions. Note that we use reflexive arcs, because places of kind Yes_p or $C_{p'}$ may be used for more than one original transition. When M_d is recognized as dead for all transitions, then p_{sat} may be marked (once here, but this is only a choice of construction).

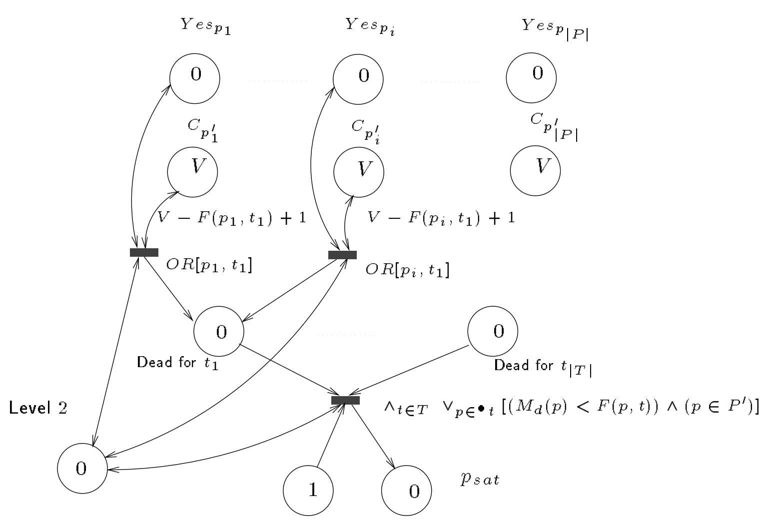

Fig. 5. From deadlock to reachability : to verify M_d.

- At level 3, the net emulates the behavior of N. A copy of N is included in the current construction with a permission-token to level 3.
- At level 4, the net stops the emulation and tests whether M_d and the current marking M in the copy of N coincide. For that, the Yes_p places are used to debit the chosen places simultaneously in M and M_d. The other non chosen places of M are emptied using the No_p places. The remaining non relevant places of the construction are emptied without condition.

Correctness: N reaches a dead marking if and only if $M_r = (1,0,0,0,1,0\cdots 0)$ is reachable in $\widehat{N}$, where the first position of M_r refers to p_{sat} and the fifth one refers to level 4. It is evident that if a dead-marking is reachable in N, then it is possible to choose it as a good candidate and to finally reach M_r. In the other sense, if no dead marking is reachable in N then there are two cases: either p_{sat} is not marked; or p_{sat} is marked but this means that the guessed marking is not reachable and that current marking in the copy of N and M_d will never coincide.

Complexity: The net $\widehat{N}$ finally contains $O(|P|+|T|)$ places and $O(|P|*|T|)$ transitions (because of the module which verifies M_d). The greatest value in $\widehat{N}$ is V. The total size is thus in $O(|N|^2)$, and the construction is linear on this size then quadratic. This concludes the proof. ∎

5 Polynomial equivalence of liveness and t-liveness

There exists a polynomial reduction from reachability to not-liveness [16] using the variation of RP which asks whether a place p may be emptied. A similar reduction exists from deadlock to not-liveness [4]. The other senses of the reductions, from not-LP to RP and from not-LP to DP, are not known. Hack [8] gave a reduction from t-liveness to liveness. In the current section we show the other sense of the reduction, from liveness to t-liveness, making the two problems many-one polynomially equivalent. Note that we do not have this equivalence for the subclass of bounded free-choice net where t-liveness is NP-complete, while liveness is polynomial [5].

Theorem 5. *Liveness is polynomially reducible to t-liveness.*

Proof: Let $N =< P,T,F,M_0 >$ be a Petri net. The construction of $\widehat{N}$ is as follows: (1) Add a place p_t in output of every transition $t \in T$; (2) Add a transition t_{test} having as input places the set of places $\{p_t | t \in T\}$. All the original transitions are live if and only if t_{test} is quasi-live from any reachable marking in $\widehat{N}$. In $\widehat{N}$ we add $|T|$ places and $O(|T|)$ transitions. The total size of the net is $O((|T|+|P|)*|T|)$ and this size is quadratic in $|N|$. The total time is linear in this size and thus polynomial. ∎

6 An upper-bound on solving boundedness for Post Self-Modifying nets

Post Self-Modifying nets (PSM-nets), defined by Valk [18,19], are more powerful than Petri nets. In this model, transitions have *extended arcs* and/or classical arcs. Extended arcs are only in output of transitions. Let us suppose that there exists an extended arc from t to place p_2 labeled with $21*p_1+4*p_3$. Firing t from M, leads to a new marking M' such that $M'(p_2) = M(p_2)+21*M(p_1)+4*M(p_3)$. Thus, the next marking depends narrowly on the current one and this is why one uses the qualifier *"self-modifying"*.

A PSM-net is a 5-tuple $< P, T, F, M_0, E >$. The four first components are the same as in Petri nets and the fifth one, component E, is a function $(T \times P \times P) \longrightarrow \mathbb{N}$ which returns a multiplicative coefficient, given a transition, an output place and a place to be consulted. In our example, we have $E(t, p_2, p_1) = 21$. Although PSM-nets are more expressive, the boundedness is still decidable and this is what makes this model attractive. The proof [18] is similar to the original one for Petri nets. However, reachability is undecidable. Let us define a lower bound on the size of an PSM-net. Let V be the greatest integer found over F, M_0 and E. We encode the flow functions with matrices as for Petri nets. The size of a PSM-net belongs to $\Omega(|P| * |T| * \log V)$.

In the current section, we give an upper bound on solving boundedness for PSM-nets. We prove that we have a polynomial-time equivalence between boundedness for Petri nets and boundedness for post self-modifying nets. The non trivial sense of the reduction, from BP to BP-PSMN, requires quadratic time. As boundedness for Petri nets is decidable in space $2^{O(|N|\log|N|)}$, we obtain $2^{O(|N|^2\log|N|)}$ as an upper space-bound for BP-PSMN. The main idea is to build a net $\widehat{N}$ that emulates the behavior of N but computes the number of tokens output of extended arcs in a *weak sense*. This means that, in the best case, the computation will be the right one but, in any other case, the computation will under-evaluate the number of tokens to be produced. Any marking reachable in $\widehat{N}$ is, in some sense, *covered* by a marking reachable in N and this implies that N is unbounded if and only if $\widehat{N}$ is unbounded.

Theorem 6. *Boundedness for PSM-nets is decidable in space* $2^{O(|N|^2\log|N|)}$

Proof: Let $N =< P, T, F, E, M_0 >$ be a post self-modifying net. We reduce BP-PSMN to BP; the time complexity is $O(|N|^2)$ leading to the theorem above. To construct $\widehat{N} =< \widehat{P}, \widehat{T}, \widehat{F}, \widehat{M_0} >$, we decompose the effect of any original transition for the weak computing of the tokens to be produced in output. Every transitions are replaced by a subnet as illustrated on an example in Fig. 6. For that, we need to associate every original place p to a place *reservoir-p* initialized with 0. We ensure the mutual exclusion between the $|T|$ subnets, such that as long as a current decomposition is not over, it is impossible to emulate another original transition. In Fig. 6, transition t has p_4 as input place, p_5 as classical output place and p_2, p_6 as "extended" output places. The arc to p_2 is labeled with $21 * p_1 + 4 * p_3$ and the arc to p_6 with $7 * p_1$. This implies that firing t from M has for consequence the addition of $21 * M(p_1) + 4 * M(p_3)$ tokens in p_2 and $7 * M(p_1)$ tokens in p_6. The emulation of t is performed in four steps :

- **Start** t **:** the decomposition begins with the update of input places (here p_4) and classical output places (here p_5). Control is given to the next step.
- **Update by** p_1 **:** the weak computations of $21 * M(p_1)$ and $7 * M(p_1)$ take place here. As long as desired, t_{p_1} debits p_1 of 1 token, crediting at the same time its reservoir place of 1 token, p_2 of 21 tokens and p_6 of 7 tokens. If the process ends when p_1 is empty, then p_2 and p_6 received the exact number of tokens; otherwise they received less tokens than aimed.

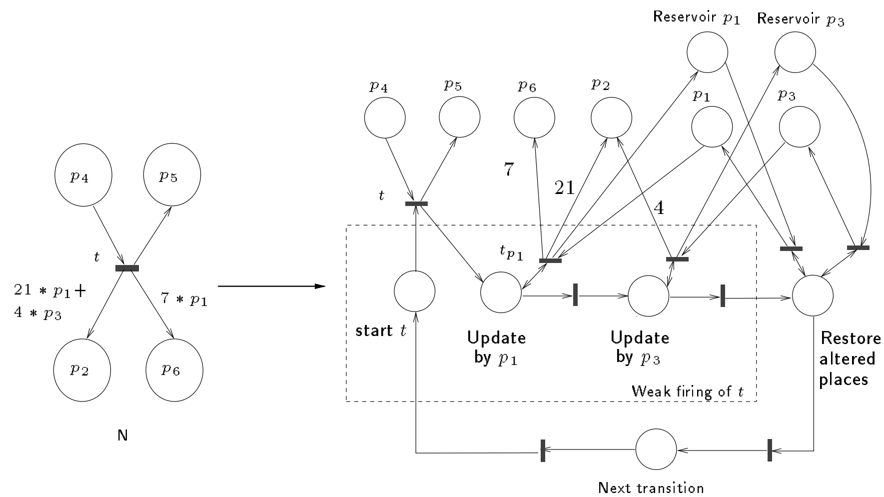

Fig. 6. Reduction from boundedness–PSMN to boundedness : weak firing of t.

- **Update by** p_3 **:** the weak computation of the multiplicative coefficient for p_3 takes place here. The value $4 * M(p_3)$ is calculated in a weak sense, debiting p_3 but keeping a trace in reservoir-p_3 by the same time.
- **Restore altered places :** we have now to restore the original contents of places p_1 and p_3. As long as desired, the contents of the reservoirs are put back into the original places. If the process continues up to empty the reservoirs, then p_1 and p_3 are restored; otherwise, they receive less tokens than aimed. Note that in this last case, we have not however lost any tokens because the remaining ones are in the reservoir places. Control is given to the next transition to be emulated.

When all the steps are fully processed, we find in places p_6 and p_2 the right number of tokens, and we leave p_1 and p_3 unaltered. At any time, and this is the interesting point, if we "merge" any pair p and its reservoir by making their sum, we find a marking which is covered by a marking that is reachable in the original PSM-net. Moreover when the decompositions are well performed we find a marking reachable in the original PSM-net. These two facts are sufficient to make the reduction correct. Note that the construction needs to be a bit adapted for other cases such as reflexive extended arc.

Correctness: The original net N is unbounded if and only if the built net $\widehat{N}$ is unbounded. If N is unbounded, then $\widehat{N}$ is unbounded because there is always a way to emulate correctly the original net. If N is bounded then: either $\widehat{N}$ fully performs the decomposition steps and produces as many tokens as N produces at any step; or it produces less tokens.

Complexity: The original places, the reservoirs and the mechanism which restores the places are common to all the decompositions of original transitions. Each decomposition of an original transition requires $O(|P|)$ places and transitions in worst case. The whole net $\widehat{N}$ contains $O(|P| + (|T| * |P|))$ places and $O(|T| + (|T| * |P|))$ transitions. The greatest value in $\widehat{N}$ is $\log V$. The total size is thus $O((|T| * |P|)^2 * \log V)$ and the construction is linear on this size, thus $O(|N|^2)$ and this concludes the proof. ∎

7 Conclusion

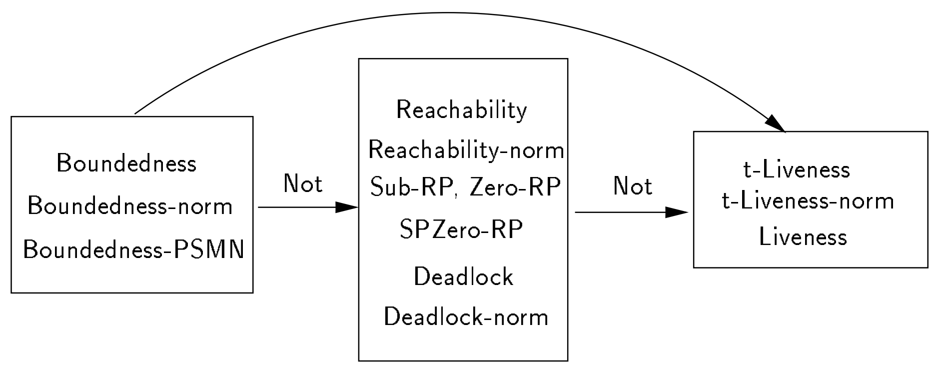

Fig. 7. Summary of polynomial-time many-one reductions.

In this paper we were interested in ordering Petri net problems, boundedness, reachability, deadlock, liveness and t-liveness, through their complexity. The Fig. 6 summarizes the contribution of our work. The main results are the following:

We give an illustration of the expressive power of reachability by reducing to it the not-boundedness and the deadlock problems. Reachability is a very vulnerable property in term of decidability and often becomes undecidable, as soon as the power of Petri nets is increased. An example of an extended model for which RP is undecidable, is the class of Petri nets allowing *Reset* arcs [1]; a Reset arc clears a place as a consequence of a firing.

We put in the same class the reachability and the deadlock problems. These problems were known to be recursively equivalent and thus, our comparison is more precise.

We give $2^{O(|N|^2 * \log |N|)}$ as an upper-bound on the space-complexity for boundedness in post self-modifying nets, and this bound is not so far from the one for Petri nets, even though PSM-nets are strictly more powerful than Petri nets.

Acknowledgments. Thanks to the anonymous referees for their useful remarks.

References

1. T. Araki and T. Kasami. Some decision problems related to the reachability problem for Petri nets. *TCS*, 3(1):85–104, 1977.
2. Z. Bouziane. *Algorithmes primitifs récursifs et problèmes* EXPSPACE-*complets pour les réseaux de Petri cycliques.* PhD thesis, LSV, École Normale Supérieure de Cachan, France, November 1996.
3. E. Cardoza, R. Lipton, and A. Meyer. Exponential space complete problems for Petri nets and commutative semigroups. In *Proc. of the* 8^{th} *annual ACM Symposium on theory of computing*, pages 50–54, May 1976.
4. A. Cheng, J. Esparza, and J. Palsberg. Complexity result for 1-safe nets. *TCS*, 147:117–136, 1995.
5. J. Desel and J. Esparza. *Free Choice Petri Nets.* Cambridge University Press, 1995.
6. C. Dufourd and A. Finkel. A polynomial λ-bisimilar normalization for Petri nets. Technical report, LIFAC, ENS de Cachan, July 1996. Presented at *AFL'96, Salgótarján, Hungary, 1996.*
7. J. Esparza and M. Nielsen. Decidability issues on Petri nets – a survey. *Bulletin of the EATCS*, 52:254–262, 1994.
8. M. Hack. *Decidability questions for Petri Nets.* PhD thesis, M.I.T., 1976.
9. J.E. Hopcroft and J.D. Ullman. *Introduction to automata theory, languages, and computation.* Addison-Wesley, 1979.
10. M. Jantzen. Complexity of Place/Transition nets. In *Petri nets: central models and their properties*, volume 254 of *LNCS*, pages 413–434. Springer-Verlag, 1986.
11. R.M. Karp and R.E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3:146–195, 1969.
12. R. Kosaraju. Decidability of reachability in vector addition systems. In *Proc. of the* 14^{th} *Annual ACM Symposium on Theory of Computing, San Francisco*, pages 267–281, May 1982.
13. R.J. Lipton. The reachability problem requires exponential space. Technical Report 62, Yale University, Department of computer science, January 1976.
14. E.W. Mayr. An algorithm for the general Petri net reachability problem. *SIAM Journal on Computing*, 13(3):441–460, 1984.
15. E.W. Mayr and R. Meyer. The complexity of the word problem for commutative semigroups and polynomial ideals. *Advances in Mathematics*, 46:305–329, 1982.
16. J.L. Peterson. *Petri Net Theory and the Modeling of Systems.* Prentice Hall, 1981.
17. C. Rackoff. The covering and boundedness problems for vector addition systems. *TCS*, 6(2), 1978.
18. R. Valk. Self-modifying nets, a natural extension of Petri nets. In *Proc. of ICALP'78*, volume 62 of *LNCS*, pages 464–476. Springer-Verlag, September 1978.
19. R. Valk. Generalizations of Petri nets. In *Proc. of the* 10^{th} *Symposium on Mathematical Fondations of Computer Science*, volume 118 of *LNCS*, pages 140–155. Springer-Verlag, 1981.
20. R. Valk and G. Vidal-Naquet. Petri nets and regular languages. *Journal of Computer and System Sciences*, 23(3):299–325, 1981.