Reduction Rules for Reset/Inhibitor Nets

H.M.W. Verbeek[1], M.T. Wynn[2], W.M.P. van der Aalst[1,2], A.H.M. ter Hofstede[2]

Department of Mathematics and Computer Science, Eindhoven University of Technology
PO Box 513, NL-5600 MB Eindhoven, The Netherlands.
{*h.m.w.verbeek,w.m.p.v.d.aalst*}*@tue.nl*
Business Process Management Group, Queensland University of Technology
GPO Box 2434, Brisbane QLD 4001, Australia.
{*m.wynn,a.terhofstede*}*@qut.edu.au*
Corresponding author: Dr. Moe T. Wynn, Phone: (617) 31389385, Fax: (617)31389390

Abstract. Reset/inhibitor nets are Petri nets extended with reset arcs and inhibitor arcs. A reset arc allows a transition to remove all tokens from a certain place when the transition fires. An inhibitor arc can stop a transition from being enabled if the place contains one or more tokens. While reset/inhibitor nets increase the expressive power of Petri nets, they also result in increased complexity of analysis techniques. One way of speeding up Petri net analysis is to apply reduction rules. Unfortunately, many of the rules defined for classical Petri nets do not hold in the presence of reset and/or inhibitor arcs. Moreover, new rules can be added. This is the first paper systematically presenting a comprehensive set of reduction rules for reset/inhibitor nets. These rules are liveness and boundedness preserving and are able to dramatically reduce models and their state spaces. Note that most of the modelling languages used in practice have features related to cancellation and blocking. Therefore, this work is highly relevant for all kinds of application areas where analysis is currently intractable.
Keywords: Reduction rules, Petri nets, reset arcs, inhibitor arcs, liveness, boundedness.

1 Introduction

Petri nets are a well-established formalism for modeling and analysing concurrent systems. Over time many extensions have been proposed in order to capture specific, possibly quite complex, behaviour in a more direct manner. These extensions include reset arcs and inhibitor arcs. Reset arcs provide a natural means of dealing with cancellation behaviour. For example, a customer may cancel a travel request which would result in certain activities terminating. A *reset arc* is a type of arc that goes from a place to a transition and its semantics is to remove all tokens from that place when the transition fires [5, 7, 8, 10, 11]. Inhibitor arcs provide a natural means of dealing with blocking behaviour. For example, an invoice should only be generated when the items ordered are ready for delivery and the order has not been cancelled. An *inhibitor arc* is a type of arc that goes from a place to a transition and its semantics is to prevent the transition from firing when the place contains one or more tokens [12, 14].

While these extensions increase the expressiveness of Petri nets, they can compromise analysis techniques and certain properties may even become undecidable. Examples of such properties are the reachability problem, which is undecidable for Petri nets with two inhibitor arcs and for Petri nets with reset arcs, and place invariants, which do not hold for Petri nets with reset arcs. Examples of such analysis techniques are reachability and coverability analysis, which can be used to detect structural and behavioural properties of Petri nets [13]. Coverability analysis has been extended in order to deal with Petri nets with reset arcs [10] and also in order to deal with Petri nets with inhibitor arcs [3]. Limiting the practical applicability of reachability and coverability analysis is the problem of *state explosion*, which occurs in nets where a very large number of markings need to be considered.

Reduction rules for Petri nets have been proposed to deal with the state explosion problem. Reduction rules can reduce the size of the net while preserving certain essential properties such as liveness. Their application therefore has the potential to significantly speed up the analysis process. A significant body of research exists that addresses the concept of reduction in the area of Petri nets (see e.g. [2, 13]) and its various subclasses (see e.g. [6]) and extensions (see e.g. [17]). However, as far as we know, the issue of reduction in Petri nets with both reset and inhibitor arcs has not been considered in the literature. Existing reduction rules are not directly applicable in the presence of both types of arcs.

Business process modeling languages used in practice, e.g. UML Activity Diagrams, the Business Process Modelling Notation (BPMN) and the Business Process Execution Language (BPEL), offer features which correspond to cancellation and blocking. In capturing their semantics, reset arcs and inhibitors arcs have a natural place. Hence the analysis of business process modeling languages mapped to Petri nets with reset and inhibitor arcs could benefit from reduction rules developed for such nets. Here it can be added that the application of general translations of modeling notations to Petri nets typically results in nets with many "dummy" transitions that are used to glue various parts of the model together. Reduction rules can then be quite effective in reducing the resulting nets.

In this paper a number of reduction rules for Petri nets with reset and inhibitor arcs are proposed. These are inspired by reduction rules provided for Petri nets in [2, 13] and for Free Choice Petri nets provided in [6]. Additional conditions are proposed to cater for the presence of reset and inhibitor arcs. The proposed rules are shown to preserve liveness and boundedness.

The contributions of the paper are as follows. (1) The paper aims to make a contribution to the body of theory in Petri nets with reset and inhibitor arcs by providing a set of liveness and boundedness preserving reduction rules. (2) In practical terms, the reduction rules presented in this paper can be used for an efficient analysis of business process models described using various business process modelling languages that support cancellation and blocking such as the Business Process Modelling Notation (BPMN), the Business Process Execution Langauge (BPEL) and the Unified Modelling Language (UML). The organisation of the remainder of this paper is as follows. Section 2 provides terminology, concepts, notations and formal definitions that are required in subsequent sections of the paper. In section 3 a set of liveness and boundedness preserving reduction rules for Petri nets with reset and inhibitor arcs are introduced. Section 4 discusses related work and section 5 concludes the paper.

2 Preliminaries

This section provides the formal foundation for Petri nets and reset/inhibitor nets as it is used throughout this paper. Readers familiar with Petri nets, reset arcs, and inhibitor arcs, may very well skip this section, although the particular notations used in the paper might still be of interest to them.

2.1 Petri nets

In its basic form, a Petri net consists of a set of places, a set of transitions, and a set of arcs that connect places to transitions and vice versa. Note that arcs do not connect places to places or transitions to transitions. For sake of simplicity, we assume that a Petri net contains at least one place.

Definition 1 (Petri net [13]). *A Petri net is a tuple (P, T, F) where P is a (non-empty finite) set of places, T is a set of transitions, $P \cap T = \varnothing$ and $F \subseteq (P \times T) \cup (T \times P)$ is the set of arcs.*

Let N be a Petri net (P, T, F), and let x be a node of N, that is, let $x \in P \cup T$. We use $\bullet x$ and $x\bullet$ to denote the set of input nodes and output nodes respectively. If the net involved cannot be understood from the context, we explicitly include net N in the notation and we write $\overset{N}{\bullet} x$ and $x \overset{N}{\bullet}$. Relation F implies a function and $F(x, y)$ evaluates to 1 if $(x, y) \in F$ and to 0 otherwise.

To every place of a Petri net $N = (P, T, F)$ a (non-negative) counter can be associated. The actual values of all these counters of all places of the net is called a *marking* of that net, and corresponds to a state of the net. Let $\mathbb{M}(N)$ denote the set of all possible markings of a net N, and let $M \in \mathbb{M}(N)$ be a marking of net N. Then $M \in (P \to \mathbb{N})$, and M can also be interpreted as a vector, function, or multiset over the set of places P. Typically, a marking $M \in \mathbb{M}(N)$ is visualized by putting $M(p)$ *tokens* (black dots) into every place p. Thus, the number of tokens in a place corresponds to the actual value of its counter.

A marking M contains another marking M', denoted $M \geq M'$, *iff* the counters of M are at least the counters of M', that is, for every $p \in P$: $M(p) \geq M'(p)$. Likewise, a marking M exceeds a marking M', denoted $M > M'$, *iff* $M \geq M'$ and $M \neq M'$. Markings M and M' can be added, denoted $M + M'$, in a straightforward way (for every $p \in P$:$(M + M')(p) = M(p) + M'(p)$). Furthermore, these markings can be subtracted, denoted $M - M'$, in a straightforward way (for every $p \in P$: $(M - M')(p) = M(p) - M'(p)$), provided that the former marking contains the latter ($M \geq M'$). In definitions to come, we use the fact that every set of places induces a marking in a straightforward way (by associating the value 1 to every place). As a result, we can add (subtract) a set of places to (from) a marking, and can compare (contains, exceeds) sets of places to markings. Finally, we use $\mathbf{0}$ to denote the empty marking, that is, $\mathbf{0}(p) = 0$ for every place p.

Places hold the current state of a Petri net, but transitions may change this current state by *firing*. However, before a transition fires, it should be *enabled*. A transition is enabled if all input places have tokens, that is, if all the counters of its input places exceed zero. If an enabled transition fires, it removes a token from every input place and adds a token to every output place, that is, it decreases the counter of its input places, and increments the counter of its output places. Note that because the transition is enabled, the counters of its input places will be at least 0 after the transition has fired.

Definition 2 (Enabling and firing a transition in a Petri net). *Let $N = (P, T, F)$ be a Petri net, $t \in T$ and $M, M' \in \mathbb{M}(N)$. Transition t is* enabled *at M, denoted as $M[t\rangle$,* iff *$M \geq \bullet t$. If transition t is enabled at M, then it may* fire*, which results in a marking M', where $M' = M - \bullet t + t\bullet$. This, we denote by $M \overset{N,t}{\rightarrow} M'$.*

If there can be no confusion regarding the net, the expression is abbreviated as $M \overset{t}{\rightarrow} M'$ and if the transition is not relevant, it is written as $M \rightarrow M'$. We write $M \overset{N,\sigma}{\rightarrow} M_n$ if $\sigma = t_1 t_2 ... t_n$ is an occurrence sequence leading from M to M_n i.e. $M \overset{t_1}{\rightarrow} M_1 \overset{t_2}{\rightarrow} ... \overset{t_n}{\rightarrow} M_n$. The empty occurrence sequence is denoted ϵ.

A Petri net $N = (P, T, F)$ together with a marking $M \in \mathbb{M}(P)$ is called a marked Petri net, denoted (N, M). Clearly, a marked Petri net induces a state space, where every state corresponds to a reachable marking. The set of all reachable markings is called the reachability set of the marked Petri net (N, M) and is denoted $N[M\rangle$. This reachability set is the minimal set that satisfies the following conditions:

- the initial marking is reachable, that is, $M \in N[M\rangle$, and
- if a marking is reachable which enables some transition, then the marking that results from firing this transition is also reachable, that is, if $M' \in N[M\rangle$ and $M'[t\rangle$ then $(M' - \bullet t + t\bullet) \in N[M\rangle$.

A marked Petri net (N, M) is called *live iff* every transition can get enabled from every reachable marking. As a result, in a live marked Petri net no transition can get shut down.

Definition 3 (Liveness [6]). *Let (N, M) be a marked Petri net with the initial marking M. (N, M) is live* iff *for every $M' \in N[M\rangle$ and every $t \in T$ there exists a $M'' \in N[M'\rangle$ such that $M''[t\rangle$.*

A marked Petri net (N, M) is called *bounded iff* every counter of every place has a maximal value. As a result, for a bounded marked Petri net, the number of reachable states is finite.

Definition 4 (Boundedness [6]). *Let (N, M) be a marked Petri net with the initial marking M. (N, M) is bounded* iff *there exists a natural number $b \in \mathbb{N}$ such that for every $M' \in N[M\rangle$ and $p \in P$ it holds that $M'(p) \leq b$.*

2.2 Reset/Inhibitor nets

A reset net [7] is a Petri net with special *reset arcs*, that can clear the tokens in selected places. Reset arcs are represented as doubled-headed arrows. An inhibitor net [12, 3] is a Petri net with inhibitor arcs. Inhibitor arcs are used to test for absence of tokens in a place. A transition t can only fire if all its inhibitor places are empty. Graphically, an inhibitor arc connects a place to a transition and the arc ends with an empty circle on the transition side.

Definition 5 (reset/inhibitor net). *A reset/inhibitor net is a tuple (P, T, F, R, I) where (P, T, F) is a Petri net, $R : T \rightarrow \mathbb{P}(P)$ provides the reset places for the transitions, and $I : T \rightarrow \mathbb{P}(P)$ provides the inhibitor places for the transitions.*

The notations $R(t)$ and $I(t)$ for a transition t return the (possibly empty) set of places that it resets and that inhibit it. We also write $R^{\leftharpoonup}(p)$ and $I^{\leftharpoonup}(p)$ for a place p, which returns the set of transitions that can reset p and that are inhibited by p. Furthermore, we introduce a notation to project a marking M onto a set of places P, denoted $M \upharpoonright P$: $(M \upharpoonright P)(p) = M(p)$ if $p \in P$ and $(M \upharpoonright P)(p) = 0$ otherwise.

The notions of inputs, outputs and markings defined for an ordinary Petri net also apply to reset/inhibitor nets. Clearly, inhibitor arcs affect whether transitions are enabled, whereas reset arcs affect the result of firing an enabled transition.

Definition 6 (Enabling and firing a transition in a reset/inhibitor net). *Let $N = (P, T, F, R, I)$ be a reset/inhibitor net, $t \in T$ and $M, M' \in \mathbb{M}(N)$. Transition t is* enabled *at M, denoted as $M[t\rangle$,* iff *$M \geq \bullet t$ and $M \upharpoonright I(t) = \mathbf{0}$. If a transition is enabled at M, it may* fire*, which results in a marking M', where $M' = (M - \bullet t) \upharpoonright (P \setminus R(t)) + t\bullet$.*

Mutatis mutandis, the definitions of liveness and boundedness for marked reset/inhibitor nets are the same as defined for marked Petri nets.

2.3 An example reset/inhibitor net

Figure 1 shows an example reset/inhibitor net. As usual, circles represent places, squares represent transitions, and black dots represent tokens. As mentioned before, the arc with the open dot at the end is an inhibitor arc, whereas the arc with the double-headed arrow is a reset arc. The dashed area represents a cluster of places that is being reset by the same set of transitions: ca, sp, mdc, and md. For sake of readability, we have replaced all reset arcs from these places to these transitions by one reset area.

The reset/inhibitor net shown in Figure 1 is loosely based on the description of the *visa application for general skilled migration to Australia*, which can be found on the Internet (see http://www.immi.gov.au). The process starts when a visa application is received (rva) and ends when the applicant cancel the request (ca), the processing is stopped due to non-responsiveness of the applicant (sp), or when the application is finalized in a proper way (fa). In the latter case, the visa can either be granted (gv) or denied (dv), in which case the applicant is notified (na). Typically, after the application has been received, a case officer opens a file for the applicant (oaf), processes application fees (paf), and performs an initial assessment (pia). It the application is complete (c), the officer continues with the main assessment (pma). Otherwise (nc), the officer s/he sends an acknowledgement letter to the applicant (sal) and requests further information (rfi). After having completed the main assessment, the case officer might request for more information (rmi), or s/he makes a decision (mdc or md). However, before making the decision, the officer first needs to check whether circumstances have changed (ccc). If the officer receives the requested additional information (rri), the main assessment is performed again. However, the applicant could wait too long to supply the office with

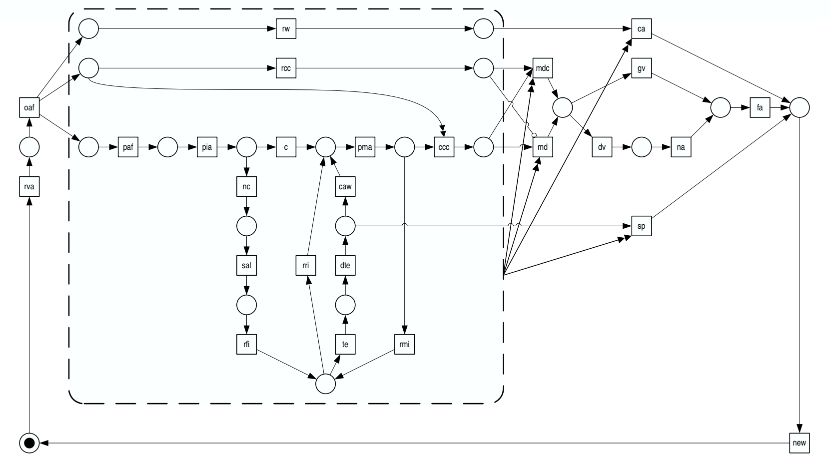

Fig. 1. An example reset/inhibitor net

the requested information (time expiry, te), in which case the officer needs to decide (dte) whether to stop processing the application (sp) or to continue anyway (caw).

While the application is being processed, but before the decision is made, two events might occur. First, an applicant may decide to withdraw his/her application (receive withdraw, rw); second, an applicant can notify the officer that his/her circumstances (for example, change of address) have changed (receive circumstance change, rcc). After the application has been handled, the applicant may decide to reapply for a visa (new).

3 Reduction rules

In this section, we present eight reduction rules for reset/inhibitor nets. The underlying rules for marked Petri nets presented in this section are based on existing reduction rules for Petri nets and free-choice nets [13, 6], and are therefore not original as such, rather the contribution is in the identification of the conditions under which they can be applied in the presence of reset and inhibitor arcs.

For sake of clarity, we decided to first present applicable conditions for marked Petri nets, before extending these rules for marked reset/inhibitor nets. We also show that these reduction rules preserve liveness and boundedness. The style of presentation is inspired by [6].

3.1 Fusion of Series Transitions

Using the Fusion of Series Transitions rule, we can reduce two transitions and a place to one transition. Thus, we can effectively remove a place and a transition. For the rule to be applicable, we need the two transitions and place to be in a series. The place acts as a kind of transient place for the output places of the series. Tokens from this transient place can be considered as being *ghost* tokens in these output places: These ghost tokens are not there yet, but they may arrive at any moment. If something happens to these ghost tokens, it should happen to the tokens in the transient place. For transitions that consume these ghost tokens, this means that the intermediate transition (the second one in the series) should fire first.

Definition 7 (Fusion of Series Transitions Rule for marked Petri nets: ϕ_{FST}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(S_1, S_2) \in \phi_{\mathrm{FST}}$ if there exists a place $p \in P_1$, two transitions $t, u \in T_1$, and a transition $v \in T_2 \setminus T_1$ such that:*

Conditions on S_1:

1. *$\bullet p = \{t\}$ (t is the only input of p)*
2. *$p\bullet = \{u\}$ (u is the only output of p)*
3. *$\bullet u = \{p\}$ (p is the only input of u)*
4. *$t\bullet \cap u\bullet = \varnothing$ (any output of t is not an output of u and vice versa)*

Construction of S_2:

5. *$P_2 = P_1 \setminus \{p\}$*
6. *$T_2 = (T_1 \setminus \{t, u\}) \cup \{v\}$*
7. *$F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2))) \cup (\overset{N_1}{\bullet} t \times \{v\}) \cup (\{v\} \times ((t \overset{N_1}{\bullet} \cup u \overset{N_1}{\bullet}) \setminus \{p\}))$*
8. *for all $x \in P_2$: $M_2(x) = \begin{cases} M_1(x) & \text{if} \quad x \notin u\bullet \\ M_1(x) + M_1(p) & \text{if} \quad x \in u\bullet \end{cases}$*

Theorem 1 (The ϕ_{FST} rule is boundedness and liveness preserving). *Let S_1 and S_2 be two marked Petri nets such that $(S_1, S_2) \in \phi_{\mathrm{FST}}$. Then S_1 is bounded* iff *S_2 is bounded, and S_1 is live* iff *S_2 is live.*

Proof The ϕ_{FST} rule is boundedness and liveness preserving [13].

Figure 2 shows both the ϕ_{FST} and the upcoming ϕ^{RI}_{FST} rule. As usual, transitions are visualized using squares and places by circles. The places and transitions that are relevant for the rule at hand are white inside, whereas the places and transitions in their allowed environment are grey inside. To visualize that this environment might include multiple places and/or transitions, we have stacked three places and/or transitions. Reset arcs are visualized using a double-headed arrow, whereas inhibitor arcs are visualized using an open dot at the end. Thus, in Figure 2, transition t may have additional output places, and transition u is not allowed to reset any place (as there is no reset place for u in the allowed environment) nor should it inhibit any place. For the ϕ_{FST} rule presented in Definition 7, we simply have to ignore every reset and inhibitor arc.

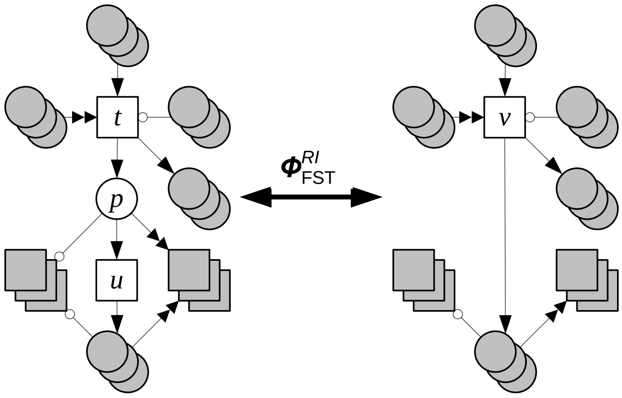

Fig. 2. Fusion of series transitions

As mentioned before, this rule holds as we can consider the tokens in place p to be matched by *ghost* tokens in the output places of transition u. These ghost tokens have not arrived yet, but they will arrive when needed by firing u. From this observation, the restrictions on reset arcs and inhibitor arcs follow in a straightforward way:

– Transition u should not be inhibited. As u needs to be enabled if p is marked, any inhibitor should be ineffective: If u is inhibited by some place x, then x should be empty when p is marked. In some cases this can be checked using only structural properties. However, it is not possible to formulate simple requirements. Therefore, we simply require that u has no inhibitor arcs.
– Transition u should not reset. We cannot tell exactly when u may fire. However, the effect of these resets should always be the same: If in some firing sequence u resets some place x by removing 2 tokens, then in any other firing sequence it should reset x by removing 2 tokens. As this too is hard to check using only structural properties, we do not allow u to have any reset arcs.
– Place p and the output places of transition u should inhibit the same set of transitions. Assume that place x is an output place of u and that x inhibits some transition y. As a result, transition y should be inhibited if x contains ghost tokens. Therefore, place p should inhibit y, and thus, every output place of u should inhibit y (as these places may contain ghost tokens of p as well).
– Place p and the output places of transition u should all be reset by the same set of transitions. Assume that place x is an output place of u and that x is being reset by some transition y. As y also resets the ghost tokens in x, it should also reset p, and thus, all other output places of u.

Definition 8 (Fusion of Series Transitions Rule for marked reset/inhibitor nets: ϕ^{RI}_{FST}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi^{RI}_{\mathrm{FST}}$ if there exists a place $p \in P_1$, two transitions $t, u \in T_1$, and a transition $v \in T_2 \setminus T_1$ such that:*

Extension of the ϕ_{FST} rule:

1. $(((P_1, T_1, F_1), M_1), ((P_2, T_2, F_2), M_2)) \in \phi_{\mathrm{FST}}$ *(Note that, by definition, the t, u, v, and p mentioned in this definition have to coincide with the t, u, v, and p as mentioned in the definition of ϕ_{FST}.)*

Conditions on R_1:

2. *for all $q \in u\bullet$: $R_1^{\leftarrow}(p) = R_1^{\leftarrow}(q)$ (p is being reset by the same transitions as every output place of u is)*
3. *$R_1(u) = \varnothing$ (u does not reset)*

Conditions on I_1:

4. *for all $q \in u\bullet$: $I_1^{\leftarrow}(p) = I_1^{\leftarrow}(q)$ (p inhibits the same transitions as every output place of u does)*
5. *$I_1(u) = \varnothing$ (u is not inhibited)*

Construction of R_2:

6. *for all $x \in T_2$: $R_2(x) = \begin{cases} R_1(x) \setminus \{p\} & \text{if} \quad x \neq v \\ R_1(t) \setminus \{p\} & \text{if} \quad x = v \end{cases}$*

Construction of I_2:

7. *for all $x \in T_2$: $I_2(x) = \begin{cases} I_1(x) \setminus \{p\} & \text{if} \quad x \neq v \\ I_1(t) \setminus \{p\} & \text{if} \quad x = v \end{cases}$*

We now present two lemmas that show that occurrence sequences in N_1 and N_2 correspond to another. These lemmas are then used to prove that the ϕ_{FST}^{RI} rule preserves liveness and boundedness.

Lemma 1. *[Under the ϕ_{FST}^{RI} rule, sequence in S_1 correspond to sequences in S_2] Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets such that $(S_1, S_2) \in \phi_{\mathrm{FST}}^{RI}$, let $\sigma_1 \in T_1^*$ and $M_1' \in \mathbb{M}(P_1)$ be such that $M_1 \stackrel{N_1,\sigma_1}{\longrightarrow} M_1'$, and let $\sigma_2 = \alpha(\sigma_1)$, where $\alpha \in T_1^* \to T_2^*$ is defined as follows:*

- *$\alpha(\epsilon) = \epsilon$,*
- *$\alpha(t\sigma) = v\alpha(\sigma)$,*
- *$\alpha(u\sigma) = \alpha(\sigma)$, and*
- *$\alpha(x\sigma) = x\alpha(\sigma)$, where $x \in T_1 \setminus \{t, u\}$.*

Thus, α removes every occurrence of u from the sequence, and replaces every occurrence of t with v. Then $M_2 \stackrel{N_2,\sigma_2}{\longrightarrow} M_2'$, where $M_2'(x) = M_1'(x) + M_1'(p)$ for every $x \in u\stackrel{N_1}{\bullet}$ and $M_2'(x) = M_1'(x)$ for every $x \notin u\stackrel{N_1}{\bullet}$.

Proof By induction on the length of σ_1.

Base Assume $\sigma_1 = \epsilon$. Clearly, $M_1 \stackrel{N_1,\sigma_1}{\longrightarrow} M_1$ and $M_2 \stackrel{N_2,\sigma_2}{\longrightarrow} M_2$. The where-clause holds, as ϕ_{FST}^{RI} implies ϕ_{FST}.

Step Assume the lemma holds for some σ_1, let M_1' be such that $M_1 \overset{N_1,\sigma_1}{\longrightarrow} M_1'$, and let M_2' be such that $M_2 \overset{N_2,\alpha(\sigma_1)}{\longrightarrow} M_2'$. We prove that it also holds if we extend σ_1 by one transition.

- First, assume that we extend σ_1 by t. As t and v have the same preset, we can extend $\alpha(\sigma_1)$ by v. t adds a token to place p, whereas v adds tokens to its postset, which does not violate the where-clause.
- Second, assume that we extend σ_1 by u. It is obvious that u does not violate the where-clause.
- Third, assume that we extend σ_1 by x, where $x \in P_1 \setminus \{t, u\}$. As all places in M_2' contain at least as many tokens as their counterparts in M_1' (the where-clause), we know that x is enabled in S_2 at M_2' as well, *provided* it is not inhibited by a place in the postset of u (as these places may contain more tokens in M_2' than in M_1'). However, due to the where-clause, a transition inhibited in S_2 at M_2' would have been inhibited in S_1 at M_1' as well.

Lemma 2. *[Under the ϕ^{RI}_{FST} rule, sequences in S_2 correspond to sequences in S_1] Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets such that $(S_1, S_2) \in \phi^{RI}_{\mathrm{FST}}$, let $\sigma_2 \in T_2^*$ and $M_2' \in \mathbb{M}(P_2)$ be such that $M_2 \overset{N_2,\sigma_2}{\longrightarrow} M_2'$, and let $\sigma_1 = \beta(\sigma_2)$, where $\beta \in T_2^* \to T_1^*$ is defined as follows:*

- $\beta(\epsilon) = \epsilon$,
- $\beta(v\sigma) = tu\beta(\sigma)$, *and*
- $\beta(x\sigma) = x\beta(\sigma)$, *where* $x \in T_2 \setminus \{v\}$.

Thus, β replaces every occurrence of v with tu. Then $M_1 \overset{N1,\sigma_1}{\longrightarrow} M_1'$, where $M_1'(p) = 0$ and $M_1'(x) = M'(x)$ for every $x \in P_1 \setminus \{p\}$.

Proof By induction on the length of σ_2.

Base Assume $\sigma_2 = \epsilon$. Clearly, $M_2 \overset{N_2,\sigma_2}{\longrightarrow} M_2$ and $M_1 \overset{N_1,\sigma_1}{\longrightarrow} M_1$. The where-clause holds, as ϕ^{RI}_{FST} implies ϕ_{FST}, which also implies $M_1(p) = 0$.

Step Assume the lemma holds for some σ_2, let M_2' be such that $M_2 \overset{N_2,\sigma_2}{\longrightarrow} M_2'$, and let M_1' be such that $M_1 \overset{N_1,\beta(\sigma_2)}{\longrightarrow} M_1'$. We prove that is also holds if we extend σ_2 by one transition.

- First, assume that we extend σ_2 by v. It is obvious that t is enabled in S_1 at M_1', and that u is enabled after having fired t. Furthermore, the combination of tu and v does not violate the where-clause.
- Second, assume that we extend σ_2 by x such that $x \in T_2 \setminus \{v\}$. Again, it is obvious that x is enabled in S_1 at M_1', and that x does no violate the where-clause.

From these lemmas, liveness and boundedness follow in a straightforward way.

Theorem 2 (The ϕ^{RI}_{FST} rule preserves liveness).

Proof Assume $(S_1, S_2) \in \phi^{RI}_{\mathrm{FST}}$ such that S_1 is live and S_2 is not live (as ϕ^{RI}_{FST} is symmetrical, we only need to consider this case). Thus, in S_2 we can reach a marking M'_2 from which some transition t cannot be enabled. Due to Lemma 2, we can reach a marking M'_1 in S_1 such that its where-clause holds. As S_1 is live, we can reach a marking M''_1 in S_1 through some occurrence sequence σ_1 such that t is enabled. Due to Lemma 1 we can thus reach a marking M''_2 in S_2 such that its where clause-holds. Obviously, t should be enabled in M''_2. Thus, S_2 has to be live as well.

Theorem 3 **(The ϕ^{RI}_{FST} rule preserves boundedness).**

Proof Assume $(S_1, S_2) \in \phi^{RI}_{\mathrm{FST}}$ such that S_1 is bounded and S_2 is not bounded (as ϕ^{RI}_{FST} is symmetrical, we only need to consider this case). Thus, for every $b \in \mathbb{N}$ we can reach a marking M'_2 in S_2 in which some place p contains more than b tokens. Due to Lemma 2, we can reach a marking M'_1 in S_1 such that $M'_1(p) = M'_2(p)$. Thus S_1 has to be unbounded as well.

The remaining reduction rules all preserve liveness and boundedness. For the Fusion of Series Places, the required proofs for this claim are similar to the proofs presented for the current, Fusion of Series Transitions, rule, whereas for the other rules the required proofs are simpler. As these proofs add little or nothing to the paper, we decided not to include them.

3.2 Fusion of Series Places

Using the Fusion of Series Places rule, we can reduce two places and one transition to one place. Thus, like the Fusion Series Transitions rule, this rule also effectively removes a transition and a place. However, the Fusion of Series Places may be applicable in situations where the Fusion of Series Transitions rule is not. Again like the Fusion of Series Transitions rule, this rule is applicable if the places and transitions are in a series, and again we can use the concept of ghost tokens to explain the rule. Tokens which reside in the first place of the series can be considered to be ghost tokens for the second place. If some transition needs to consume these ghost tokens, the intermediate transitions should fire first, removing the ghost tokens by real ones.

Definition 9 **(Fusion of Series Places Rule for marked Petri nets: ϕ_{FSP}).** *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(S_1, S_2) \in \phi_{\mathrm{FSP}}$ if there exist two places $p, q \in P_1$, a transition $t \in T_1$, and a place $r \in P_2 \setminus P_1$ such that:*

Conditions on S_1:

1. $\bullet t = \{p\}$ *(p is the only input of t)*
2. $t\bullet = \{q\}$ *(q is the only output of t)*
3. $p\bullet = \{t\}$ *(t is the only output of p)*
4. $\bullet p \cap \bullet q = \varnothing$ *(any input of p is not an input of q and vice versa)*

Construction of S_2:

5. $P_2 = (P_1 \setminus \{p, q\}) \cup \{r\}$
6. $T_2 = T_1 \setminus \{t\}$
7. $F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2))) \cup (((\overset{N_1}{\bullet} p \cup \overset{N_1}{\bullet} q) \setminus \{t\}) \times \{r\}) \cup (\{r\} \times q \overset{N_1}{\bullet})$
8. *for all* $x \in P_2$*:* $M_2(x) = \begin{cases} M_1(x) & \text{if} \quad x \neq r \\ M_1(p) + M_1(q) & \text{if} \quad x = r \end{cases}$

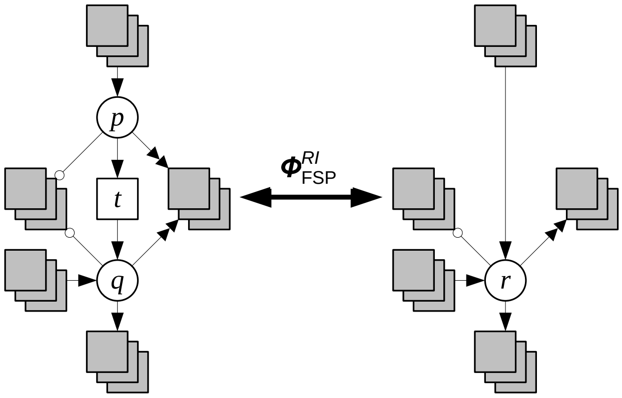

Fig. 3. Fusion of series places

Tokens in place p are matched by ghost tokens in place q. Again, these tokens have not arrived yet, but they will materialize if needed by firing transition t. Again, the restrictions on reset arcs and inhibitor arcs follow in a straightforward way from this observation:

- Transition t should not be inhibited. As it is hard to check on ineffective inhibitor arcs, we simply require that t has no inhibitor arcs.
- Transition t should not reset. As it is hard to check that every reset has the same effect, we simply require that t has no reset arcs.
- Place p should be inhibited by the same set of transitions as place q.
- Place p should be being reset by the same set of transitions as place q.

Definition 10 (Fusion of Series Places Rule for marked reset/inhibitor nets:ϕ^{RI}_{FSP}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi^{RI}_{\mathrm{FSP}}$ if there exist two places $p, q \in P_1$, a transition $t \in T_1$, and a place $r \in P_2 \setminus P_1$ such that:*

Extension of the ϕ_{FSP} rule:

1. $(((P_1, T_1, F_1), M_1), ((P_2, T_2, F_2), M_2)) \in \phi_{\mathrm{FSP}}$ *(Note that, by definition, the p, q, t, and r mentioned in this definition have to coincide with the p, q, t, and r as mentioned in the definition of ϕ_{FSP}.)*

Conditions on R_1:

2. $R_1(t) = \varnothing$ *(t does not reset)*
3. $R_1^{\leftarrow}(p) = R_1^{\leftarrow}(q)$ *(p and q are being reset by the same transitions)*

Conditions on I_1:

4. $I_1(t) = \varnothing$ *(t does not have inhibitor arcs)*
5. $I_1^{\leftarrow}(p) = I_1^{\leftarrow}(q)$ *(p and q have the same set of inhibitor arcs)*

Construction of R_2:

6. *for all* $x \in T_2$*:* $R_2(x) = \begin{cases} (R_1(x) \setminus \{p,q\}) \cup \{r\} & \textit{if} \quad \{p,q\} \cap R_1(x) \neq \varnothing \\ R_1(x) & \textit{if} \quad \{p,q\} \cap R_1(x) = \varnothing \end{cases}$

Construction of I_2:

7. *for all* $x \in T_2$*:* $I_2(x) = \begin{cases} (I_1(x) \setminus \{p,q\}) \cup \{r\} & \textit{if} \quad \{p,q\} \cap I_1(x) \neq \varnothing \\ I_1(x) & \textit{if} \quad \{p,q\} \cap I_1(x) = \varnothing \end{cases}$

3.3 Fusion of Parallel Transitions

Using the Fusion of Parallel Transitions rule, we can reduce a number of transitions to one transition. This rule is applicable if all transitions have the same set of input places and the same set of output places. Clearly, all transitions are enabled at the same time, and all have the same effect.

Definition 11 (Fusion of Parallel Transitions Rule for marked Petri nets: ϕ_{FPT}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(S_1, S_2) \in \phi_{\mathrm{FPT}}$ if there exists transitions $V \subseteq T_1$ where $|V| \geq 2$, an arbitrary transition $t \in V$, and a transition $v \in T_2 \setminus T_1$ such that:*
Conditions on S_1:

1. *for all* $x, y \in V : \bullet x = \bullet y$ *(input places for all transitions in V are identical)*
2. *for all* $x, y \in V : x\bullet = y\bullet$ *(output places for all transitions in V are identical)*

Construction of S_2:

3. $P_2 = P_1$
4. $T_2 = (T_1 \setminus V) \cup \{v\}$
5. $F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2))) \cup (\overset{N_1}{\bullet} t \times \{v\}) \cup (\{v\} \times t \overset{N_1}{\bullet})$
6. $M_2 = M_1$

As the transitions should be enabled at the same times, either all or none should be inhibited. As a check on ineffective inhibitor arcs is hard, we simply require the transitions to have the same set of inhibitors. Furthermore, their effects should be identical. As it is hard to check when the effect of a transition that resets some place is identical to the effect of a transition that does not reset that place, we simply require that every transition resets the same set of places.

Definition 12 (Fusion of Parallel Transitions Rule for marked reset/inhibitor nets: ϕ^{RI}_{FPT}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi^{RI}_{\mathrm{FPT}}$ if there exists transitions $V \subseteq T_1$ where $|V| \geq 2$, an arbitrary transition $t \in V$, and a transition $v \in T_2 \setminus T_1$ such that:*

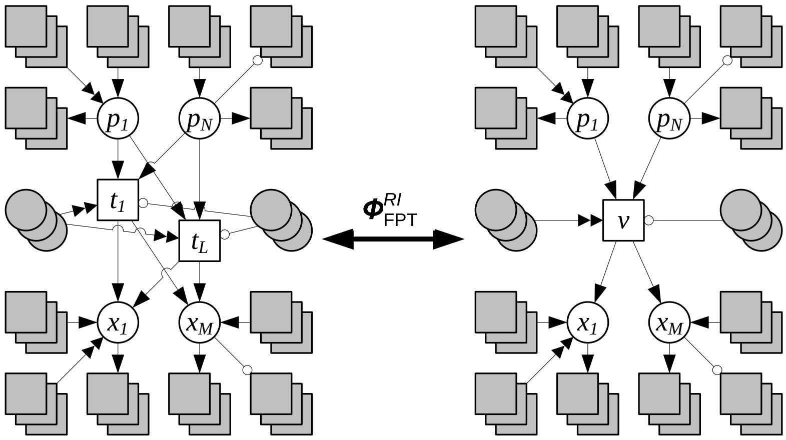

Fig. 4. Fusion of parallel transitions

Extension of the ϕ_{FPT} rule:

1. $(((P_1, T_1, F_1), M_1), ((P_2, T_2, F_2), M_2)) \in \phi_{\mathrm{FPT}}$ *(Note that, by definition, the V and v mentioned in this definition have to coincide with the V and v as mentioned in the definition of ϕ_{FPT}.)*

Condition on R_1:

2. *for all $x, y \in V : R_1(x) = R_1(y)$ (all transitions in V reset the same places)*

Condition on I_1:

3. *for all $x, y \in V : I_1(x) = I_1(y)$ (all transitions in V share the same set of inhibitor arcs)*

Construction of R_2:

4. *for all $x \in T_2$: $R_2(x) = \begin{cases} R_1(x) & \text{if} \quad x \neq v \\ R_1(t) & \text{if} \quad x = v \end{cases}$*

Construction of I_2:

5. *for all $x \in T_2$: $I_2(x) = \begin{cases} I_1(x) & \text{if} \quad x \neq v \\ I_1(t) & \text{if} \quad x = v \end{cases}$*

3.4 Fusion of Parallel Places

Using the Fusion of Parallel Places rule, we can reduce a number of places to one place. This rule is applicable if all places have the same set of input transitions and the same set of output transitions. Clearly, only a place among these places that initially contains the fewest tokens can become empty and can, hence, disable any transitions. Therefore, all other places are implicit and can be removed.

Definition 13 (Fusion of Parallel Places Rule for marked Petri nets: ϕ_{FPP}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(S_1, S_2) \in \phi_{\mathrm{FPP}}$ if there exists places $Q \subseteq P_1$ where $|Q| \geq 2$, an arbitrary place $p \in Q$ and a place $q \in P_2 \setminus P_1$ such that:*

Conditions on S_1:

1. *for all $x, y \in Q$: $\bullet x = \bullet y$ (input transitions for all places in Q are identical)*
2. *for all $x, y \in Q$: $x\bullet = y\bullet$ (output transitions for all places in Q are identical)*

Construction of S_2:

3. $P_2 = (P_1 \setminus Q) \cup \{q\}$
4. $T_2 = T_1$
5. $F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2))) \cup (\overset{N_1}{\bullet} p \times \{q\}) \cup (\{q\} \times p \overset{N_1}{\bullet})$
6. *for all $x \in P_2$:* $M_2(x) = \begin{cases} M_1(x) & \text{if} \quad x \neq q \\ \min_{y \in Q} M_1(y) & \text{if} \quad x = q \end{cases}$

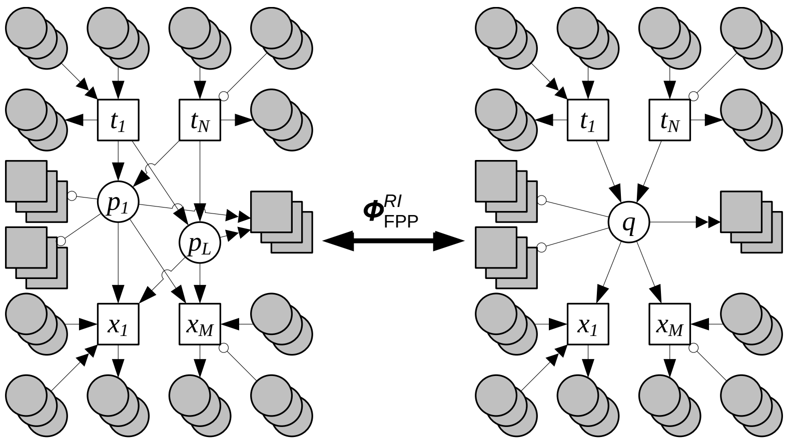

Fig. 5. Fusion of parallel places

When adding reset arcs and inhibitor arcs, we should guarantee that the other places remain implicit. Thus, these other places should always contain at least as many tokens as the place we keep. Therefore, any transition that resets any other place, should also reset the place we keep. However, we may not allow the other places to become unbounded if the place we keep is bounded. For this reason, we require that any transition that resets the one place, should also reset all other places. As a result, we require that all places in Q are being reset by the same set of transitions. However, for inhibitor arcs something similar *does not hold*. Figure 6 shows an example where the rightmost parallel place contains more tokens than the leftmost place. Note that we may not initialize the place in the right-hand net with two tokens, as this would allow for two firings of

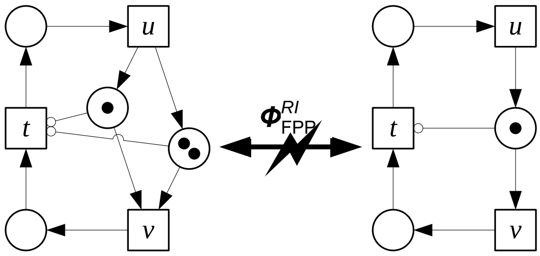

Fig. 6. Fusion of parallel places: Inhibitors explained

transition v. In both marked nets transition v can fire once, but the left-hand net is then dead, whereas the right-hand net is not. Clearly, this is caused by the fact that a token was left in the right-most parallel place (of the left-hand net), which inhibits transition t. For this reason, we do not allow a parallel place to inhibit a transition if initially it contains more tokens than its sibling places. Note that due to reset arcs both places may be drained from tokens, after which we could allow an inhibitor arc for both places. However, as there is no simple way to guarantee (using only structural information) that the parallel places have to be reset before they can inhibit, we do not use this insight.

Definition 14 (Fusion of Parallel Places Rule for marked reset/inhibitor nets: ϕ^{RI}_{FPP}). *Let $s_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two reset nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi^{RI}_{\mathrm{FPP}}$ if there exists places $Q \subseteq P_1$ where $|Q| \geq 2$ and a place $q \in P_2 \setminus P_1$ such that:*

Extension of the ϕ_{FPP} rule:

1. $(((P_1, T_1, F_1), M_1), ((P_2, T_2, F_2), M_2)) \in \phi_{\mathrm{FPP}}$ *(Note that, by definition, the Q and q mentioned in this definition have to coincide with the Q and q as mentioned in the definition of ϕ_{FPP}.)*

Condition on R_1:

2. *for all $x, y \in Q$: $R_1^{\leftarrow}(x) = R_1^{\leftarrow}(y)$ (all places in Q are being reset by the same transitions)*

Condition on I_1:

3. *for all $x \in Q$: if $M_1(x) > \min_{y \in Q} M_1(y)$ then $I_1^{\leftarrow}(x) = \varnothing$ (only places with a minimal initial marking may inhibit transitions)*

Construction of R_2:

4. *for all $x \in T_2$:* $R_2(x) = \begin{cases} (R_1(x) \setminus Q) \cup \{q\} & \textit{if} \quad R_1(x) \cap Q \neq \varnothing \\ R_1(x) & \textit{if} \quad R_1(x) \cap Q = \varnothing \end{cases}$

Construction of I_2:

5. *for all $x \in T_2$:* $I_2(x) = \begin{cases} (I_1(x) \setminus Q) \cup \{q\} & \textit{if} \quad I_1(x) \cap Q \neq \varnothing \\ I_1(x) & \textit{if} \quad I_1(x) \cap Q = \varnothing \end{cases}$

3.5 Elimination of Self-Loop Transitions

Using the Elimination of Self-Loop Transitions rule, we can remove a self-loop transition, that is, a transition that has only one input place and only one output place, and for which the input place and the output place are identical. Clearly, firing the transition does not have any effect. Thus, removing the transition does not effect boundedness. However, removing it could affect liveness, as it can be the only non-live transition. To prevent this, we require that the input/output place has at least one additional input transition.

Definition 15 (Elimination of Self-Loop Transitions Rule for marked Petri nets: ϕ_{ELT}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(S_1, S_2) \in \phi_{\mathrm{ELT}}$ if there exists a place $p \in P_1$, and a transition $t \in T_1$ such that:*
Conditions on S_1:

1. *$\bullet t = \{p\}$ (p is the only input place of t)*
2. *$t\bullet = \{p\}$ (p is the only output place of t)*
3. *$|\bullet p| \geq 2$ (p has at least one additional input transition)*

Construction of S_2:

4. *$P_2 = P_1$*
5. *$T_2 = T_1 \setminus \{t\}$*
6. *$F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2)))$*
7. *$M_2 = M_1$*

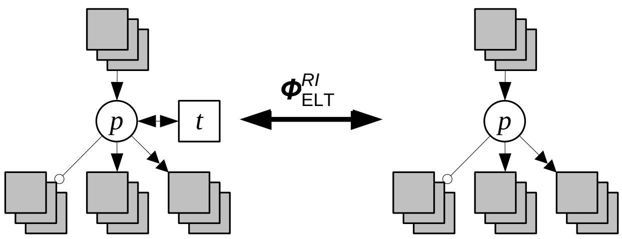

Fig. 7. Elimination of self-loop transitions

Clearly, after reset arcs and inhibitor arcs have been added, t should be enabled at some point in time, and its effect should not result in a new marking. Thus:

– any place that inhibits t should be emptiable while place p is marked, and
– t should not reset any place.

As the first requirement is hard to obtain from the structure of the marked net, we simply require that t is not inhibited at all.

Definition 16 (Elimination of Self-Loop Transitions Rule for marked reset/inhibitor nets: ϕ^{RI}_{ELT}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi^{RI}_{\mathrm{ELT}}$ if there exists a place $p \in P_1 \cap P_2$ and a transition $t \in T_1$ such that:*

Extension of the ϕ_{ELT} rule:

1. *$(((P_1, T_1, F_1), M_1), ((P_2, T_2, F_2), M_2)) \in \phi_{\mathrm{ELT}}$ (Note that, by definition, the t and p mentioned in this definition have to coincide with the t and p as mentioned in the definition of ϕ_{ELT}.)*

Condition on R_1:

2. *$R_1(t) = \varnothing$ (t does not reset)*

Condition on I_1:

3. *$I_1(t) = \varnothing$ (t does not have any inhibitor arcs)*

Construction of R_2:

4. *for all $x \in T_2$: $R_2(x) = R_1(x)$*

Construction of I_2:

5. *for all $x \in T_2$: $I_2(x) = I_1(x)$*

3.6 Elimination of Self-Loop Places

The Elimination of Self-Loop Places rule can be used to remove places that are always marked. As a result, these places never disable any output transition.

Definition 17 (Elimination of Self-Loop Places for marked Petri nets: ϕ_{ELP}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(S_1, S_2) \in \phi_{\mathrm{ELP}}$ if there exists a place $p \in P_1 \setminus P_2$ such that:*
Conditions on S_1:

1. *$p\bullet = \bullet p$ (the inputs of p are also its outputs)*
2. *$M_1(p) \geq 1$ (p is marked at M_1)*

Construction of S_2:

3. *$P_2 = P_1 \setminus \{p\}$*
4. *$T_2 = T_1$*
5. *$F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2)))$*
6. *for all $x \in P_2$: $M_2(x) = M_1(x)$*

Clearly, place p should not inhibit any transition. Furthermore, to ensure that the place is always marked, any transition that removes tokens from this place should put at least one token back. Thus, any transition that resets p should also put a token in p. However, a transition that resets p and puts a token in p does not need to consume a token using a normal input arc. Therefore, the ϕ^{RI}_{ELP} rule is not a simple extension of the ϕ_{ELP} rule. This is illustrated by the two sets of transitions in Figure 8.

Definition 18 (Elimination of Self-Loop Places Rule for marked reset/inhibitor nets: ϕ^{RI}_{ELP}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi^{RI}_{\mathrm{ELP}}$ if there exists a place $p \in P_1 \setminus P_2$ such that (note that the ϕ^{RI}_{ELP} rule is not a simple extension of the ϕ_{ELP} rule, as the first condition can be weakened):*

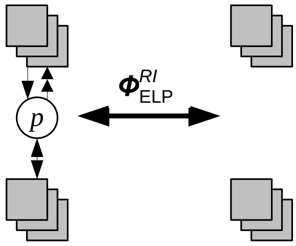

Fig. 8. Elimination of self-loop places

Conditions on S_1:

1. $p\bullet \subseteq \bullet p$ *(the outputs of p are also inputs)*
2. $M_1(p) \geq 1$ *(p is marked at M_1)*

Condition on R_1:

3. $R_1^{\leftarrow}(p) \cup p\bullet = \bullet p$ *(every reset transition or output transition should also be an input transition)*

Condition on I_1:

4. $I_1^{\leftarrow}(p) = \varnothing$ *(p does not inhibit any transition)*

Construction of S_2:

5. $P_2 = P_1 \setminus \{p\}$
6. $T_2 = T_1$
7. $F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2)))$
8. *for all* $x \in P_2$*:* $M_2(x) = M_1(x)$

Construction of R_2:

9. *for all* $x \in T_2$*:* $R_2(x) = R_1(x) \cap P_2$

Construction of I_2:

10. $I_2 = I_1$

3.7 Abstraction

Like the Fusion of Series Transitions rule and the Fusion of Series Places rule, using the Abstraction rule, we can remove a place and a transition. In fact, the Abstraction rule is in some way a mix of both fusion rules. Like both fusion rules, this rule can be understood using the concept of ghost tokens. Basically, we can replace a place-transition pair (where the place is the only input of the transition and the transition is the only output of the place) by a number of arcs connecting every input transition of the place to every output place of the transition, thus bypassing both. Any token in the place is matched by ghost tokens in the output places of the transition. If needed, these ghost tokens can materialize by firing the transition.

Definition 19 (Abstraction Rule for marked Petri nets: ϕ_{A}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked Petri nets, where $N_1 = (P_1, T_1, F_1)$ and $N_2 = (P_2, T_2, F_2)$. $(N_1, N_2) \in \phi_{\mathrm{A}}$ if there exists places $Q \subseteq P_1 \cap P_2$, a place $s \in P_1 \setminus Q$, transitions $U \subseteq T_1 \cap T_2$, and a transition $t \in T_1 \setminus U$ such that:*

Conditions on S_1:

1. *$\bullet t = \{s\}$ (s is the only input of t)*
2. *$s\bullet = \{t\}$ (t is the only output of s)*
3. *$\bullet s = U$ (transitions in U are input transitions for s)*
4. *$t\bullet = Q$ (places in Q are output places for t)*
5. *$(\bullet s \times t\bullet) \cap F = \varnothing$ (any input of s is not connected to an output of t and vice versa)*

Construction of S_2:

6. *$P_2 = P_1 \setminus \{s\}$*
7. *$T_2 = T_1 \setminus \{t\}$*
8. *$F_2 = (F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2))) \cup (\overset{N_1}{\bullet} s \times t \overset{N_1}{\bullet})$*
9. *for all $x \in P_2$: $M_2(x) = \begin{cases} M_1(x) & \text{if} \quad x \notin Q \\ M_1(x) + M_1(s) & \text{if} \quad x \in Q \end{cases}$*

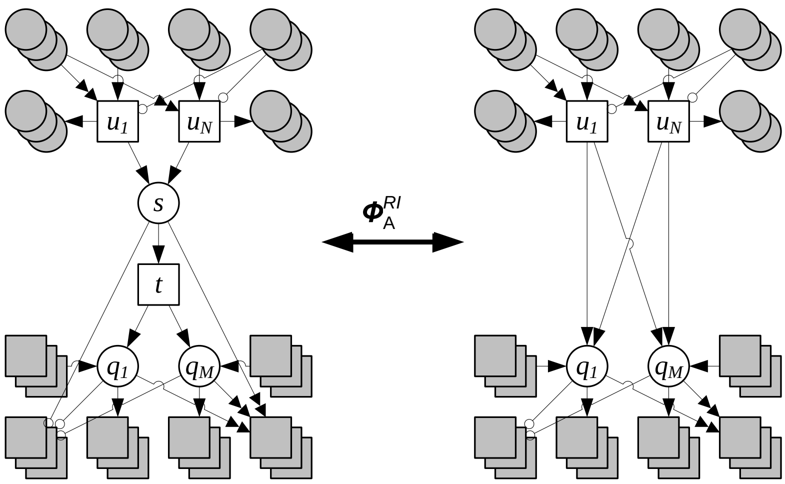

Fig. 9. Abstraction

Like with the fusion rules, transition t should not be inhibited, as this might disable the firing of t. Also likewise, t should not reset any place. Like with the Fusion of Series Transitions rule, place s should inhibit the same set of transitions as every output place of t does, and it is being reset by the same set of transitions that reset every output place of t.

Definition 20 (Abstraction Rule for marked reset/inhibitor nets: ϕ_{A}^{RI}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi_{\mathrm{A}}^{RI}$ if there exists places $Q \subseteq P_1 \cap P_2$, a place $s \in P_1 \setminus Q$, transitions $U \subseteq T_1 \cap T_2$, and a transition $t \in T_1 \setminus U$ such that:*

Extension of the ϕ_{A}^{R} rule:

1. *$(((P_1, T_1, F_1), M_1), ((P_2, T_2, F_2), M_2)) \in \phi_{\mathrm{A}}$ (Note that, by definition, the s, t, Q, and U mentioned in this definition have to coincide with s, t, Q, and U as mentioned in the definition of ϕ_{A}.)*

Conditions on R_1:

2. *$R_1^{\leftarrow}(s) = R_1^{\leftarrow}(q)$, for every $q \in Q$ (s is being reset by transitions that reset Q)*
3. *$R_1(t) = \varnothing$ (t does not reset)*

Conditions on I_1:

4. *$I_1^{\leftarrow}(s) = I_1^{\leftarrow}(q)$, for every $q \in Q$ (s inhibits the same transitions as every place from Q does)*
5. *$I_1(t) = \varnothing$ (t is not inhibited by any place)*

Construction of R_2:

6. *for all $x \in T_2$: $R_2(x) = R_1(x) \cap P_2$*

Construction of I_2:

7. *for all $x \in T_2$: $I_2(x) = I_1(x) \cap P_2$*

3.8 Reset reduction

If a transition u resets a place that inhibits it, then the reset arc is clearly redundant: The transition can only fire if the place is empty. Note that the place may optionally be an input place and/or output place of u as well (if it is an input place as well, u will be dead of course, but the rule still applies).

Definition 21 (Reset Reduction Rule for marked reset/inhibitor nets: ϕ_{R}^{RI}). *Let $S_1 = (N_1, M_1)$ and $S_2 = (N_2, M_2)$ be two marked reset/inhibitor nets, where $N_1 = (P_1, T_1, F_1, R_1, I_1)$ and $N_2 = (P_2, T_2, F_2, R_2, I_2)$. $(S_1, S_2) \in \phi_{\mathrm{R}}^{RI}$ if there exists a place $u \in P_1 \cap P_2$ and a transition $t \in T_1 \cap T_2$ such that:*

Conditions on S_1:

1. *$p \in R_1(u) \cap I_1(u)$*

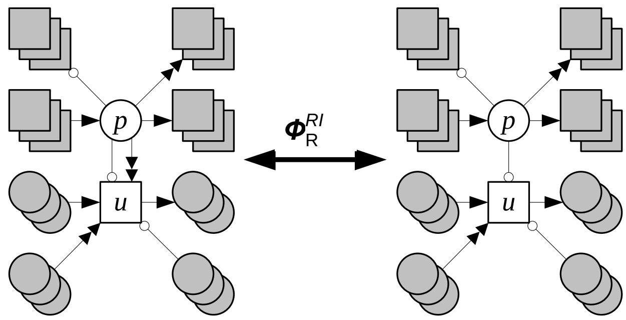

Fig. 10. Reset reduction

Construction of S_2:

2. $P_2 = P_1$
3. $T_2 = T_1$
4. $F_2 = F_1$
5. *for all* $x \in T_2$: $R_2(x) = \begin{cases} R_1(x) & \textit{if} \quad x \neq u \\ R_1(x) \setminus \{p\} & \textit{if} \quad x = u \end{cases}$
6. $I_2 = I_1$
7. $M_2 = M_1$

3.9 The visa example

To illustrate what we can achieve by reduction, we apply the rules defined earlier to the visa example shown in Figure 1. Figure 11 shows this example with a number of possible reductions highlighted. Figure 12 shows the visa example after all the reductions shown in Figure 11 have been applied. Clearly, some of the reduction rules can still be applied. Figure 13 shows the example after two rounds of reduction rules have been applied. As this figure shows, no more reduction rules can be applied now.

As a result of applying the reduction rules on the visa example, the number of places is reduced from 21 to 9, and the number of transitions drops from 26 to 13. As a result, the number of reachable states is reduced from 50 to 11. This will make it easier to determine both boundedness and liveness and other related properties. Note that the visa example only has a few states. Hence the reduction in states is not very spectacular. However, for more realistic examples the state space grows very rapidly. As shown in different studies (e.g., [1, 9]) reduction rules can reduce the state space dramatically. Given the generic character of the rules presented in this paper, we are confident that we can obtain similar results for Petri nets extended with reset arcs and inhibitor arcs.

4 Related work

In the general area of reset nets, Dufourd et al.'s work has provided valuable insights into the decidability of various properties of reset nets including reachability, boundedness and coverability [7, 8]. The use of backwards coverability techniques to analyse

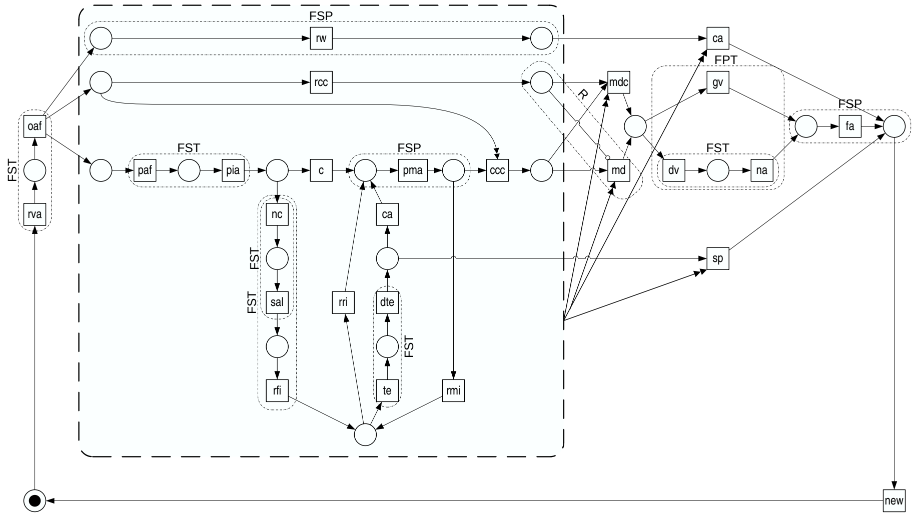

Fig. 11. The visa example with possible reductions

reset nets is discussed in [11, 10]. In [14, 15], the extension to Petri nets using inhibitor arcs is mentioned. The reachability problem of Petri nets with one inhibitor arc is studied in [16] and shown to be decidable. The reachability problem of Petri nets with at least two inhibitor arcs is shown to be undecidable [12]. In [3], the author focuses on expressiveness of inhibitor arcs and shows that an extension of coverability tree construction could be used as an analysis technique for Petri nets with inhibitor arcs. In [4], the authors propose an extension to coloured Petri nets with inhibitor arcs that supports both zero-testing inhibitors and threshold inhibitors.

A number of authors have investigated reduction rules for Petri nets and for various subclasses of Petri nets. Berthelot presents a set of reduction rules for general Petri nets, which includes transformations on places such as structurally redundant places, double places and equivalent places and fusion of transitions such as post-fusion, pre-fusion and lateral fusion [2]. These rules provided the inspiration for our work. In [13], six reduction rules are presented for Petri nets and this set of rules has been used as a starting point for the rules in this paper. In [6], a set of reduction rules was proposed for free-choice Petri nets while preserving well-formedness. In [17], the authors extend the reduction rules given by Berthelot for Time Petri nets. As part of our work on workflow verification, a set of soundness preserving reduction rules for YAWL models was presented in [18]. In this technical report similar ideas are applied to YAWL workflows with cancellation regions and OR-joins.

In [9] a comprehensive comparison of the different state-space reduction techniques is reported. Here, different reduction techniques are applied to both artificial and real-life examples. The study shows that the classical Petri net reduction rules (for nets without reset arcs and inhibitor arcs) perform very well and are able to reduce state-spaces dramatically. This illustrates the practical relevance of the results reported in this paper.

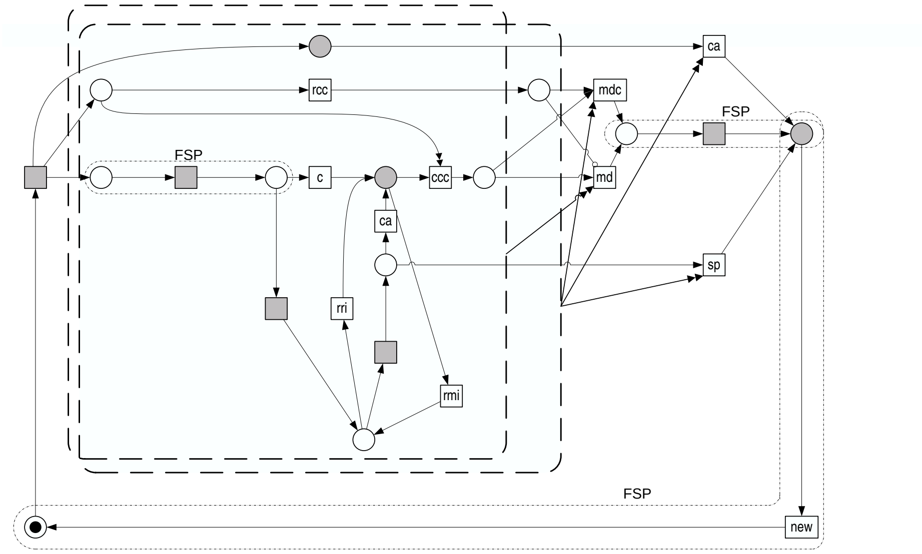

Fig. 12. The visa example after a first round of reductions

5 Conclusion

It is widely known that applying reduction rules to large Petri nets can dramatically reduce the time it takes to perform all kinds of analyses. Typically, a reduction rule will decrease the number of elements under consideration by removing certain transitions and/or places in the net while preserving some interesting properties. For Petri nets extended with reset arcs and inhibitor arcs, the existing Petri net reduction rules do not apply since each rule can be invalidated by the presence of reset arcs and/or inhibitor arcs.

In this paper, we have presented a set of eight reduction rules for reset/inhibitor nets that are liveness and boundedness preserving. These reduction rules are generic and easy to implement. We used an example to illustrate the applicability of our approach. The results allow for potentially spectacular reductions of the state space and, therefore, facilitate a more efficient analysis of reset/inhibitor nets.

In our view these results are highly relevant because real-life modelling languages such as UML, BPEL, BPMN, etc. have features such as cancellation and blocking that correspond directly to reset and inhibitor arcs. Moreover, model translations typically introduce lots of “dummy” transitions that do not correspond to real events. The results presented in this paper therefore potentially allow for a substantial speed-up of any form of Petri-net-based analysis using languages such as UML, BPEL and BPMN as a starting point.

References

1. W.M.P. van der Aalst, M. Dumas, C. Ouyang, A. Rozinat, and H.M.W. Verbeek. Chore-

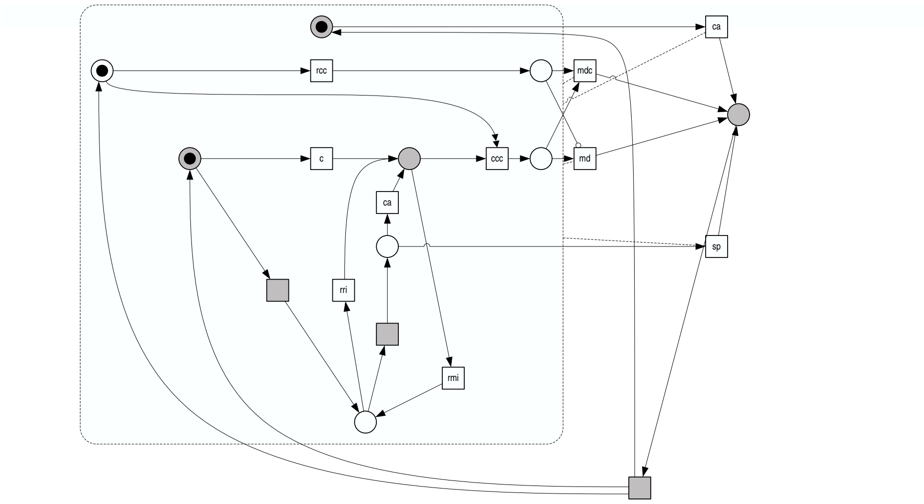

Fig. 13. The visa example after two rounds of reductions

ography Conformance Checking: An Approach based on BPEL and Petri Nets (extended version). BPM Center Report BPM-05-25, BPMcenter.org, 2005.

2. G. Berthelot. Transformations and Decompositions of Nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Petri Nets: Central Models and Their Properties, Advances in Petri Nets, Proceedings of an Advanced Course, Part 1*, volume 254 of *Lecture Notes in Computer Science*, pages 359–376, Bad Honnef, September 1986. Springer-Verlag.
3. N. Busi. Analysis Issues in Petri Nets with Inhibitor Arcs. *Theoretical Computer Science*, 275(1-2):127–177, 2002.
4. S. Christensen and N. Hansen. Coloured Petri Nets Extended with Place Capacities, Test Arcs and Inhibitor Arcs. In *Proceedings of the 14th International Conference on Application and Theory of Petri Nets*, pages 186–205, London, UK, 1993. Springer-Verlag.
5. P. Darondeau. Unbounded Petri net Synthesis. In J. Desel, W. Reisig, and G. Rozenberg, editors, *Lectures on Concurrency and Petri Nets, Advances in Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 413–428, Eichstätt, Germany, 2003. Springer-Verlag.
6. J. Desel and J. Esparza. *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, United Kingdom, 1995.
7. C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset Nets Between Decidability and Undecidability. In K. Larsen, S. Skyum, and G. Winskel, editors, *Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, volume 1443 of *Lecture Notes in Computer Science*, pages 103–115, Aalborg, Denmark, July 1998. Springer-Verlag.
8. C. Dufourd, P. Jančar, and Ph. Schnoebelen. Boundedness of Reset P/T Nets. In J. Wiedermann, P. van Emde Boas, and M. Nielsen, editors, *Lectures on Concurrency and Petri Nets*, volume 1644 of *Lecture Notes in Computer Science*, pages 301–310, Prague, Czech Republic, July 1999. Springer-Verlag.
9. S. Duri, U. Buy, R. Devarapalli, and S.M. Shatz. Application and Experimental Evaluation of State Space Reduction Methods for Deadlock Analysis in Ada. *ACM Transactions on Software Engineering Methodology*, 3(4):340–380, 1994.

10. A. Finkel, J.-F. Raskin, M. Samuelides, and L. van Begin. Monotonic Extensions of Petri Nets: Forward and Backward Search Revisited. *Electronic Notes in Theoretical Computer Science*, 68(6):1–22, 2002.
11. A. Finkel and Ph. Schnoebelen. Well-structured Transition Systems Everywhere! *Theoretical Computer Science*, 256(1–2):63–92, April 2001.
12. M. Hack. Petri Net Language. Technical report, Cambridge, MA, USA, 1976.
13. T. Murata. Petri nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.
14. J.L. Peterson. Petri nets. *ACM Computing Surveys*, 9(3):223–252, 1977.
15. J.L. Peterson. *Petri net theory and the modeling of systems*. Prentice-Hall, Englewood Cliffs, USA, 1981.
16. K. Reinhardt. Reachability in Petri Nets with Inhibitor Arcs. Technical report WSI-96-30, Wilhelm Schickard Institut für Informatik, Universitat Tübingen, 1996.
17. R.H. Sloan and U.A. Buy. Reduction Rules for Time Petri Nets. *Acta Informatica*, 33(7):687–706, 1996.
18. M.T. Wynn, H.M.W. Verbeek, W.M.P. van der Aalst, A.H.M. ter Hofstede, and D. Edmond. Reduction rules for Reset Workflow Nets. Technical report BPM-06-25, BPM Center (bpmcenter.org), 2006.