# Synthesis of Labelled Transition Systems into Equal-Conflict Petri Nets

Uli Schlachter* and Valentin Spreckels

Department of Computing Science, Carl von Ossietzky Universität
D-26111 Oldenburg, Germany
{schlachter,spreckels}@informatik.uni-oldenburg.de

**Abstract.** This paper introduces properties *preset-equality* for equal-conflict Petri nets and *enabling-equivalence* for their reachability graphs. It explores the relation between these properties and shows its use for synthesis of equal-conflict Petri nets from labelled transition systems.

## 1 Introduction

The task in Petri net synthesis is to construct a Petri net whose reachability graph is isomorphic to a given labelled transition system (lts). The theory behind Petri net synthesis is based on *regions* of the lts and goes back to [16,17]. It was developed for many classes of Petri nets [4,1,2,6,7,8] and was implemented in several tools, e.g. Petrify [14,13,15], GENET [12,11], Synet [1,3], and our own tool APT [10,9,20].

In this paper, we investigate the synthesis of equal-conflict Petri nets. Equal-conflictness is a combination of two other properties: All transitions consuming tokens from a place consume the same number of tokens (homogeneous) and transitions with non-disjoint presets have equal presets (weighted free-choice).

Our approach has two phases: First we identify structural constraints on the desired Petri net based on the lts. Then a predicate is formulated that restricts regions so that only equal-conflict solutions are found. This predicate can then be used in the framework of [20] to actually produce a Petri net. Combining this predicate with the existing predicate for *plain* yields a predicate for (plain) free-choice Petri nets.

Figure 1 shows that equal-conflict Petri nets are less expressive than general Petri nets. It shows an lts $A$ and a Petri net $N$ so that $\mathrm{RG}(N) \cong A$, but no equal-conflict Petri net can generate $A$, as will be shown later.

After the preliminaries in section 2, section 3 defines the predicate announced above and shows its correctness. Section 4 extends this result to plain and pure Petri nets and Section 5 concludes this paper.

* Supported by DFG (German Research Foundation) through grant Be 1267/15-1 ARS (Algorithms for Reengineering and Synthesis).

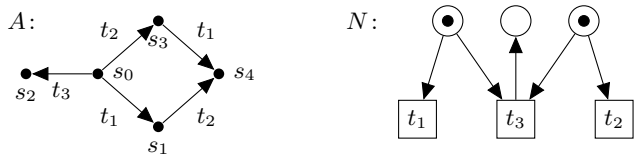


**Fig. 1.** An lts $A$ which is not isomorphic to the reachability graph of any equal-conflict Petri net and a Petri net $N$ with $\mathrm{RG}(N) \cong A$.

## 2 Petri Nets, Labelled Transition Systems and Regions

A (finite, initially marked, place-transition, arc-weighted, unlabelled) Petri net is a tuple $(P, T, F, M_0)$ such that $P$ is a finite set of *places*, $T$ is a finite set of *transitions*, with $P \cap T = \emptyset$, $F$ is a *flow* function $F\colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$, and $M_0$ is the *initial marking*, where a *marking* is a mapping $M\colon P \to \mathbb{N}$, indicating the number of *tokens* in each place. $F(p,t) = w > 0$ (resp. $F(t,p) = w > 0$) means that there is an *arc* from $p$ to $t$ (resp. from $t$ to $p$) with *arc weight* $w$. A transition $t \in T$ is *enabled by* a marking $M$, denoted by $M[t\rangle$, if for all places $p \in P$, $M(p) \geq F(p,t)$. If $t$ is enabled at $M$, then $t$ can *occur* (or *fire*) in $M$, leading to the marking $M'$ defined by $\forall p \in P\colon M'(p) = M(p) - F(p,t) + F(t,p)$ (notation: $M[t\rangle M'$). We denote by $[M_0\rangle$ the set of markings, which are reachable from $M_0$ by (repeated) firing. For a place $p$ of a Petri net $N = (P, T, F, M_0)$, let ${}^\bullet p = \{t \in T \mid F(t,p) > 0\}$ be its *preset*, and $p^\bullet = \{t \in T \mid F(p,t) > 0\}$ its *postset*. Analogously, for a transition $t$ define its *preset* as ${}^\bullet t = \{p \in P \mid F(p,t) > 0\}$, and its *postset* as $t^\bullet = \{p \in P \mid F(t,p) > 0\}$. $N$ is called *pure* or *side-condition free* if $p^\bullet \cap {}^\bullet p = \emptyset$ for all $p \in P$; *plain* (also known as *ordinary*) if $F(t,p) \leq 1$ and $F(p,t) \leq 1$ for all $p \in P$ and $t \in T$; *bounded* if $\exists k \in \mathbb{N}\colon \forall M \in [M_0\rangle\colon \forall p \in P\colon M(p) \leq k$ (i.e., a limit of the number of tokens on any place exists); *simply live* if $\forall t \in T\colon \exists M \in [M_0\rangle\colon M[t\rangle$; *homogeneous* if $\forall p \in P\colon \exists k \in \mathbb{N}\colon \forall t \in p^\bullet\colon F(p,t) = k$; *weighted free-choice* if $\forall t_1, t_2 \in T\colon {}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset \Rightarrow {}^\bullet t_1 = {}^\bullet t_2$; and *equal-conflict* if it is homogeneous and weighted free-choice.

An lts (labelled transition system with initial state) is a tuple $(S, \to, T, s_0)$, where $S$ is a set of *states*, $T$ is a set of *labels* with $S \cap T = \emptyset$, $\to \subseteq (S \times T \times S)$ is the *transition relation*, and $s_0 \in S$ is an *initial state*. A label $t$ is *enabled* in a state $s$ if there is some state $s'$ such that $(s,t,s') \in \to$. Two lts $A_1 = (S_1, \to_1, T, s_{01})$ and $A_2 = (S_2, \to_2, T, s_{02})$ over the same set of labels $T$ are *isomorphic* (notation: $A_1 \cong A_2$) if there is a bijection $\zeta\colon S_1 \to S_2$ with $\zeta(s_{01}) = s_{02}$ and $(s,t,s') \in \to_1 \iff (\zeta(s), t, \zeta(s')) \in \to_2$, for all $s, s' \in S_1$.

The *reachability graph of* $N$, with initial marking $M_0$ and transitions $T$, is the lts $\mathrm{RG}(N) = ([M_0\rangle, \{(M,t,M') \mid M, M' \in [M_0\rangle \wedge t \in T \wedge M[t\rangle M'\}, T, M_0)$. If an lts $A$ is isomorphic to the reachability graph of a Petri net $N$, then we will also say that $N$ is a *solution* of $A$.

A *region* of an lts $A = (S, \to, T, s_0)$ is a tuple $(\mathbb{R}, \mathbb{B}, \mathbb{F}) \in (\mathbb{N}^S, \mathbb{N}^T, \mathbb{N}^T)$ such that for all arcs $(s,t,s') \in \to$ in $A$, both $\mathbb{R}(s) \geq \mathbb{B}(t)$ and $\mathbb{R}(s') = \mathbb{R}(s) - \mathbb{B}(t) + \mathbb{F}(t)$ hold. Intuitively, this describes a possible place in a Petri net generating $A$

where $\mathbb{B}(t)$, resp. $\mathbb{F}(t)$, describes the number of tokens consumed, resp. produced, by a transition $t \in T$ and $\mathbb{R}(s)$ is the number of tokens on this place in state $s \in S$. The requirement above then describes an occurrence of transition $t$. Thus, a region corresponds to a Petri net with a single place. Similarly, a set of regions corresponds to a Petri net with multiple places. We call a region homogeneous if its corresponding Petri net is homogeneous and use the same convention for sets of regions and other properties, like equal-conflict and weighted free-choice.

## 3 Equal-Conflict Synthesis

This section investigates the following two properties: Two transitions $t_1, t_2 \in T$ of a Petri net $N$ are *preset-equal* if they have the same preset, ${}^\bullet t_1 = {}^\bullet t_2$. Two labels $t_1, t_2 \in T$ of an lts $A$ are *enabling-equivalent* if no state $s \in S$ only enables one of the two labels, i.e. $\forall s \in S\colon (\exists s' \in S\colon (s, t_1, s') \in \rightarrow) \Leftrightarrow (\exists s' \in S\colon (s, t_2, s') \in \rightarrow)$.

As an example for this definition, consider the equal-conflict Petri net $N$ from Figure 2. The transition $t_1$ is not preset-equal with any other transition, but $t_2$ and $t_3$ are preset-equal. A similar relationship can be seen in $\mathrm{RG}(N)$: There are states where only $t_1$ is enabled, so it is not enabling-equivalent with any other label, but $t_2$ and $t_3$ are enabling-equivalent.

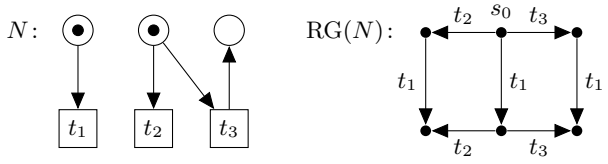


**Fig. 2.** An equal-conflict Petri net $N$ and its reachability graph $\mathrm{RG}(N)$.

The following theorems show that these two properties (almost) imply each other in equal-conflict Petri nets and their reachability graphs. This relationship is then used to define additional constraints on the regions found by Petri net synthesis, so that equal-conflict Petri nets are computed.

**Theorem 1.** *Let $N$ be a homogeneous Petri net. Then the preset-equal property on $N$ implies the enabling-equivalence property on $RG(N)$.*

*Proof.* Let $t_1, t_2 \in T$ be two preset-equal transitions. Since $N$ is homogeneous, we get for each $p \in P$ that $F(p, t_1) = F(p, t_2)$. By the definition of enabledness, any given marking enables either both transitions or neither. □

This theorem allows us to show that there is no equal-conflict Petri net $N$ with $\mathrm{RG}(N) \cong A$, where $A$ was shown in Figure 1. Assume that an equal-conflict Petri net $N$ exists. We can see that $t_1$ and $t_3$ are not enabling-equivalent, because in $s_3$

only $t_1$ is enabled. By the previous theorem, they are not preset-equal. Since we are desiring an equal-conflict solution, they must have disjoint presets. However, in the initial state both $t_1$ and $t_3$ are enabled, but firing one of them disables the other. This is only possible if they have non-disjoint presets in $N$, hence we get a contradiction. Thus, there is no equal-conflict Petri net that generates $A$.

For the opposite implication, namely deriving preset-equality from enabling-equivalence, we can only show a weaker result: There is a Petri net where enabling-equivalent transitions are preset-equal. Examples for this construction are provided in Figure 3.

**Theorem 2.** *Let $N$ be an equal-conflict Petri net. Then there is an equal-conflict Petri net $N'$ with $\mathrm{RG}(N) = \mathrm{RG}(N')$ so that enabling-equivalent transitions in $\mathrm{RG}(N')$ are preset-equal in $N'$.*

*Proof.* The following step can inductively be used to modify $N$ so that any $t_1, t_2 \in T$ which are enabling-equivalent, but not preset-equal, become preset-equal. The result of this construction will be the needed $N'$.

For any transition $t_3$ that is preset-equal with $t_2$ (including $t_2$ itself) and any place $p \in {}^\bullet t_1$, add a side-condition between $t_3$ and $p$ with weight $F(p, t_1)$. This means both $F(p, t_3)$ (previously zero by assumption) and $F(t_3, p)$ (possibly non-zero) are increased by $F(p, t_1)$. Also, do the analogous operation with $t_1$ and $t_2$ swapped.

The resulting net will still be equal-conflict by construction, but it will also satisfy ${}^\bullet t_1 = {}^\bullet t_2$. Also, its behaviour was not modified, because the effects of transitions were not modified and no transition becomes disabled in a marking that previously enabled it, because by enabling-equivalence enough tokens for the added flows are available. □

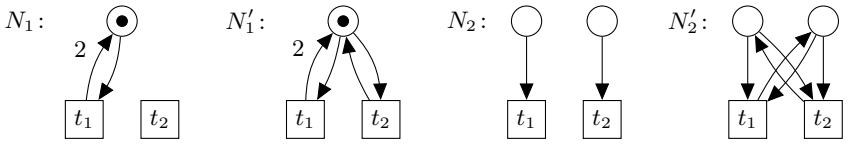


**Fig. 3.** Examples where $t_1$ and $t_2$ are enabling-equivalent in $\mathrm{RG}(N)$, but not preset-equal in $N$. The net $N'$ is constructed from $N$ via Theorem 2. In $N_1$ both transitions are always enabled while in $N_2$ both transitions can never fire.

With the results so far we can assume w.l.o.g. that in equal-conflict Petri nets, transitions are enabling-equivalent iff they are preset-equal. This can be used to define a predicate on regions, as follows.

Let an lts $A$ over the set of labels $T$ be given that should be solved by an equal-conflict Petri net $N$ with $A \cong \mathrm{RG}(N)$. First, the enabling-equivalent labels in $A$ are computed. This produces a partitioning $E \subseteq 2^T$ of the set of labels into

enabling-equivalent classes. Next, we say that a region $(\mathbb{R}, \mathbb{B}, \mathbb{F})$ is *compatible* with the desired equal-conflict Petri net $N = (P, T, F, M_0)$, if it is homogeneous, meaning that for all transitions $t_1, t_2 \in T$ if $\mathbb{B}(t_1) > 0 \wedge \mathbb{B}(t_2) > 0$ then $\mathbb{B}(t_1) = \mathbb{B}(t_2)$, and its postset is in $E \cup \{\emptyset\}$. These two conditions (homogeneous and postset in $E$) are expressed by the following predicate:

$$\bigvee_{e \in E \cup \{\emptyset\}} \left( \left( \bigwedge_{t_1, t_2 \in e} \mathbb{B}(t_1) = \mathbb{B}(t_2) \wedge \mathbb{B}(t_2) > 0 \right) \wedge \left( \bigwedge_{t \notin e} \mathbb{B}(t) = 0 \right) \right)$$

The algorithm from [20] formalises regions as an SMT problem [5] and then solves this problem. Adding this new predicate to the SMT problem yields an algorithm for the synthesis of equal-conflict Petri nets.

As an example of this, let us come back to the lts $A$ from Figure 1. We already argued that this lts cannot be solved by an equal-conflict Petri net and will now look more formally at this problem. Because no two labels are enabling-equivalent, we have $E = \{\{t_1\}, \{t_2\}, \{t_3\}\}$. Our predicate indicates that at most one transition can consume tokens from a place.

We will try to find a region $(\mathbb{R}, \mathbb{B}, \mathbb{F})$ that prevents[1] $t_1$ in state $s_2$, i.e. that satisfies $\mathbb{R}(s_2) < \mathbb{B}(t_1)$. From this formula we already conclude that $0 < \mathbb{B}(t_1)$, because $\mathbb{R}(s_2) \in \mathbb{N}$. By definition of a region, we have $\mathbb{R}(s_0) - \mathbb{B}(t_3) + \mathbb{F}(t_3) = \mathbb{R}(s_2) < \mathbb{B}(t_1)$. Because of the edge $(s_0, t_1, s_1)$ we know by definition of a region that $\mathbb{B}(t_1) \leq \mathbb{R}(s_0)$. Adding the last two inequalities produces $-\mathbb{B}(t_3) + \mathbb{F}(t_3) < 0$, which implies that $0 < \mathbb{B}(t_3)$. Thus, we have shown that $0 < \mathbb{B}(t_1)$ and $0 < \mathbb{B}(t_3)$, which is however not compatible with the predicate, since there is no $e \in E \cup \{\emptyset\}$ with $t_1, t_3 \in e$. This means that the assumption that $t_1$ can be prevented in state $s_2$ was wrong and $A$ cannot be solved by an equal-conflict Petri net.

**Theorem 3.** *Given a finite lts $A$, the synthesis procedure produces an equal-conflict Petri net $N_R$ with $\mathrm{RG}(N_R) \cong A$ iff there is an equal-conflict Petri net $N$ with $\mathrm{RG}(N) \cong A$.*

*Proof.* ($\Rightarrow$): If the algorithm finds a Petri net $N$, then each place of $N$ corresponds to a region satisfying the predicate. This means that $N$ is homogeneous and the postset of each place is in $E \cup \{\emptyset\}$. Since elements of $E$ are disjoint, $N$ is an equal-conflict Petri net.

($\Leftarrow$): To show that a solution is always found if one exists, assume a suitable Petri net $N$ exists. By Theorem 2 we can w.l.o.g. assume that enabling-equivalent transitions are preset-equal in $N$. The contraposition of Theorem 1 shows that non-enabling-equivalent transitions must not be preset-equal, which means that their presets must be disjoint by weighted free-choice. Thus, the regions corresponding to the places of $N$ satisfy the above predicate and are a possible result of the algorithm. □

[1] We are attempting to solve the ESSP instance $(s_2, t_1)$. For more details, see [20]

## 4 Incorporating Additional Properties

So far we can synthesise equal-conflict Petri nets via the new predicate and the algorithm from [20]. However, the later algorithm allows to combine multiple predicates so that, for example, pure equal-conflict Petri nets are found. Specifically, predicates for the following properties are given: plain, pure, conflict-free, homogeneous, $k$-bounded, generalised T-net, generalised marked graph, place-output-nonbranching, and distributed. In this section, we show that these predicates can be combined with the new predicate for equal-conflict synthesis.

Theorem 2 is not applicable for synthesising pure equal-conflict Petri nets, because it only shows the existence of an equivalent non-pure Petri net, as the example $N_2$ and $N_2'$ in Figure 3 demonstrates. Additionally, Figure 4 has an example $N_3$ and $N_3'$ showing that plainness is not preserved, too. All properties from the list above except for plain and pure are unaffected by the construction from Theorem 2, or are already more restrictive than equal-conflictness. The following Theorem 4 closes this gap.

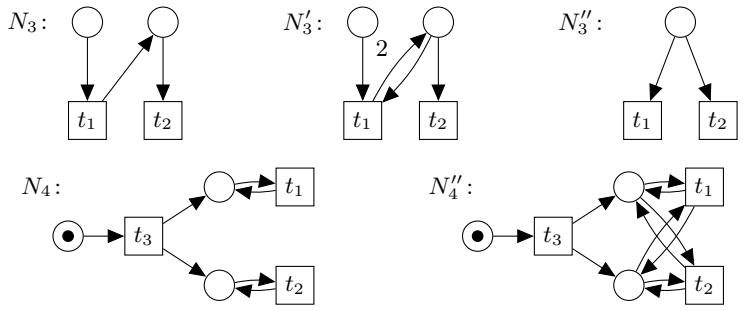


**Fig. 4.** An equal-conflict Petri net $N_3$ where $t_1$ and $t_2$ are enabling-equivalent in $\mathrm{RG}(N_3)$ (they can never fire), but not preset-equal in $N_3$. While $N_3$ is plain and pure, the construction from Theorem 2 would produce $N_3'$ which is neither plain nor pure. Instead, Theorem 4 preserves these properties and produces the Petri net $N_3''$. Similarly, $N_4$ is transformed into $N_4'$ which both are plain.

**Theorem 4.** *Let $N$ be an equal-conflict Petri net that is pure or both bounded and plain. Then there is an equal-conflict Petri net $N'$ with $\mathrm{RG}(N) \cong \mathrm{RG}(N')$ so that enabling-equivalent transitions in $\mathrm{RG}(N')$ are preset-equal in $N'$. Additionally, $N'$ is pure if $N$ is pure and plain if $N$ is plain.*

*Proof.* First we modify the transitions which can never occur, so that we can assume simple liveness for the remaining transitions. Let $D \subseteq T$ be the set of all such dead transitions. We construct $N'$ by removing all flows connected to a transition in $D$. Then, we add a new place $p$ with $M_0(p) = 0$ and add flows so that $\forall t \in D\colon F(p,t) = 1$. This place ensures that all transitions in $D$ cannot occur.

Because dead transitions do not influence the behaviour of the Petri net, we have $\mathrm{RG}(N) \cong \mathrm{RG}(N')$. Also, this construction preserves plainness and pureness.

Next, let $T' \subseteq T$ be a set of pairwise enabling-equivalent transitions containing $t_1, t_2 \in T'$ that are not preset-equal. By simple liveness, there are two cases: Either there are reachable markings $M, M'$ with $M'[t_1\rangle M$ and $\neg M[t_1\rangle$, or $t_1$ can be fired infinitely often once it is enabled.

In the first case, by enabling-equivalence we get $M'[t_2\rangle M$ and $\neg M[t_2\rangle$. Thus, firing $t_1$ disables $t_2$ and by the firing rule we deduce ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$. Because $N$ is weighted free-choice, this implies that ${}^\bullet t_1 = {}^\bullet t_2$. Since $t_2$ was arbitrary, we have that all transitions that are enabling-equivalent with $t_1$ are in fact preset-equal with it. Thus, in this case we have $N' = N$.

In the latter case, by the firing rule we have $\forall p \in P\colon F(t_1, p) \geq F(p, t_1)$. If $N$ is pure (and possibly plain), it follows that $F(p, t_1) = 0$ and so $t_1$ has an empty preset. By enabling-equivalence we get analogously ${}^\bullet t_2 = \emptyset$ and hence ${}^\bullet t_1 = \emptyset = {}^\bullet t_2$. We have again $N' = N$.

If $N$ is bounded and plain, then by boundedness we get $\forall p \in P\colon F(t_1, p) = F(p, t_1) \wedge F(t_2, p) = F(p, t_2)$. By plainness, the only option for $t_1$ and $t_2$ not to be preset-equal is if (w.l.o.g.) $F(t_1, p) = 1$ and $F(t_2, p) = 0$ for some place $p \in P$. Let $T'' \subseteq T'$ contain all transitions $t \in T'$ with $F(t, p) = 0$. We can add a side condition between each $t \in T''$ and $p$ of weight one without modifying the behaviour of the Petri net, because by enabling-equivalence with $t_1$ there is always a token in $p$ when $t \in T''$ is enabled. This produces the pure and equal-conflict Petri net $N'$ with $\mathrm{RG}(N') = \mathrm{RG}(N)$. □

With this theorem, the construction from the previous section also works when the new predicate for equal-conflict Petri net synthesis is combined with predicates for other properties.

## 5 Conclusion

In this paper we extended the family of Petri net classes which can be targeted for Petri net synthesis. The synthesis can now target equal-conflict Petri nets and, by combination with other predicates (c.f. [20]) also sub-classes, e.g. free-choice[2], which was identified as problematic in [20], because it was not clear how to identify transitions that should have the same presets. This problem was solved by introducing a preprocessing step that identifies which transitions necessarily have the same presets based on the structure of the lts to be synthesised. The result of this preprocessing step is then used to formulate a predicate that restricts the places found by Petri net synthesis.

A construction similar to Theorem 2 is called *equalisation* in [19], but used in a different context. In [15] the synthesis of free-choice Petri nets is handled by label splitting, which means that in the resulting Petri net different transitions might produce the same event, which our approach avoids. Another algorithm

[2] Combining *equal-conflict* with *plain* restricts to homogeneous free-choice Petri nets, but here *homogeneous* is not actually a restriction.

is sketched in [21]. This is a recursive approach based on integer linear programming (ILP) where the computation of a new place which would violate the free-choice condition is handled by discarding the already-found places. The authors point out that their approach negatively influences running times since places are computed repeatedly. In contrast to this, the present paper uses a pre-processing phase so that the free-choice condition is never violated. Also, we are using satisfiability modulo theories (SMT) problems, where only some solution is sought, while ILP problems are optimisation problems which we believe are harder to solve.

Besides implementing this new approach in our tool APT [10,9,20], we also want to extend Petri net synthesis to further classes of nets. For example, both weighted free-choice and asymmetric-choice[3] are generalisations of equal-conflictness which need further consideration, but might be tractable in a similar way.

**Acknowledgements:** The authors are grateful to the anonymous reviewers for their helpful remarks and to Eike Best for his help in preparing this paper.

## References

1. Badouel, E., Bernardinello, L., Darondeau, P.: Polynomial algorithms for the synthesis of bounded nets. In: Mosses, P.D., Nielsen, M., Schwartzbach, M.I. (eds.) TAPSOFT 1995. LNCS, vol. 915, pp. 364–378. Springer (1995), http://dx.doi.org/10.1007/3-540-59293-8_207
2. Badouel, E., Bernardinello, L., Darondeau, P.: Petri Net Synthesis. Springer (2015), http://dx.doi.org/10.1007/978-3-662-47967-4
3. Badouel, E., Caillaud, B., Darondeau, P.: Distributing finite automata through Petri net synthesis. Formal Asp. Comput. 13(6), 447–470 (2002), http://dx.doi.org/10.1007/s001650200022
4. Badouel, E., Darondeau, P.: Theory of regions. In: Reisig, W., Rozenberg, G. (eds.) Lectures on Petri Nets I: Basic Models, Advances in Petri Nets. LNCS, vol. 1491, pp. 529–586. Springer (1996), http://dx.doi.org/10.1007/3-540-65306-6_22
5. Barrett, C., Stump, A., Tinelli, C.: The SMT-LIB Standard: Version 2.0. In: Gupta, A., Kroening, D. (eds.) Proceedings of the 8th International Workshop on Satisfiability Modulo Theories (Edinburgh, UK) (2010)
6. Best, E., Devillers, R.R.: Characterisation of the state spaces of live and bounded marked graph Petri nets. In: Dediu, A.H., Martín-Vide, C., Sierra-Rodríguez, J.L., Truthe, B. (eds.) LATA 2014. LNCS, vol. 8370, pp. 161–172. Springer (2014), http://dx.doi.org/10.1007/978-3-319-04921-2_13
7. Best, E., Devillers, R.R.: State space axioms for t-systems. Acta Inf. 52(2-3), 133–152 (2015), http://dx.doi.org/10.1007/s00236-015-0219-0
8. Best, E., Devillers, R.R.: Synthesis of bounded choice-free Petri nets. In: Aceto, L., de Frutos-Escrig, D. (eds.) CONCUR 2015. LIPIcs, vol. 42, pp. 128–141. Schloss Dagstuhl (2015), http://dx.doi.org/10.4230/LIPIcs.CONCUR.2015.128
9. Best, E., Schlachter, U.: Analysis of Petri nets and transition systems. In: Knight, S., Lanese, I., Lluch-Lafuente, A., Vieira, H.T. (eds.) ICE 2015. EPTCS, vol. 189, pp. 53–67 (2015), http://dx.doi.org/10.4204/EPTCS.189.6

[3] If two places have non-disjoint postsets, then one of the postsets includes the other.

10. Borde, D., Dierkes, S., Ferrari, R., Gieseking, M., Göbel, V., Grunwald, R., von der Linde, B., Lückehe, D., Schlachter, U., Schierholz, C., Schwammberger, M., Spreckels, V.: APT: analysis of Petri nets and labeled transition systems, https://github.com/CvO-Theory/apt
11. Carmona, J., Cortadella, J., Kishinevsky, M.: New region-based algorithms for deriving bounded Petri nets. IEEE Trans. Computers 59(3), 371–384 (2010), http://dx.doi.org/10.1109/TC.2009.131
12. Carmona, J., Cortadella, J., Kishinevsky, M., Kondratyev, A., Lavagno, L., Yakovlev, A.: A symbolic algorithm for the synthesis of bounded Petri nets. In: van Hee and Valk [18], pp. 92–111, http://dx.doi.org/10.1007/978-3-540-68746-7_10
13. Cortadella, J., Kishinevsky, M., Kondratyev, A., Lavagno, L., Yakovlev, A.: Petrify: A tool for manipulating concurrent specifications and synthesis of asynchronous controllers (1996)
14. Cortadella, J., Kishinevsky, M., Lavagno, L., Yakovlev, A.: Synthesizing Petri nets from state-based models. In: Rudell, R.L. (ed.) ICCAD 1995. pp. 164–171. IEEE Computer Society / ACM (1995), http://dx.doi.org/10.1109/ICCAD.1995.480008
15. Cortadella, J., Kishinevsky, M., Lavagno, L., Yakovlev, A.: Deriving Petri nets for finite transition systems. IEEE Trans. Computers 47(8), 859–882 (1998), http://dx.doi.org/10.1109/12.707587
16. Ehrenfeucht, A., Rozenberg, G.: Partial (set) 2-structures. part I: basic notions and the representation problem. Acta Inf. 27(4), 315–342 (1990), http://dx.doi.org/10.1007/BF00264611
17. Ehrenfeucht, A., Rozenberg, G.: Partial (set) 2-structures. part II: state spaces of concurrent systems. Acta Inf. 27(4), 343–368 (1990), http://dx.doi.org/10.1007/BF00264612
18. van Hee, K.M., Valk, R. (eds.): ICATPN 2008, LNCS, vol. 5062. Springer (2008)
19. Recalde, L., Teruel, E., Silva, M.: Improving the decision power of rank theorems. IEEE Trans. Syst. Man, Cybern. B, Cybern. 4, 3768–3773 (1997), http://dx.doi.org/10.1109/ICSMC.1997.633256
20. Schlachter, U.: Petri net synthesis for restricted classes of nets. In: Kordon, F., Moldt, D. (eds.) ICATPN 2016. LNCS, vol. 9698, pp. 79–97. Springer (2016), http://dx.doi.org/10.1007/978-3-319-39086-4_6
21. van der Werf, J.M.E.M., van Dongen, B.F., Hurkens, C.A.J., Serebrenik, A.: Process discovery using integer linear programming. In: van Hee and Valk [18], pp. 368–387, http://dx.doi.org/10.1007/978-3-540-68746-7_24