Automated Polyhedral Abstraction Proving

Nicolas Amat, Silvano Dal Zilio, Didier Le Botlan

Automated Polyhedral Abstraction Proving

Nicolas Amat[1], Silvano Dal Zilio[1], and Didier Le Botlan[1]

[1]LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France

Abstract

We propose an automated procedure to prove polyhedral abstractions for Petri nets. Polyhedral abstraction is a new type of state-space equivalence based on the use of linear integer constraints. Our approach relies on an encoding into a set of SMT formulas whose satisfaction implies that the equivalence holds. The difficulty, in this context, arises from the fact that we need to handle infinite-state systems. For completeness, we exploit a connection with a class of Petri nets that have Presburger-definable reachability sets. We have implemented our procedure, and we illustrate its use on several examples.

Keywords— Automated reasoning; Abstraction techniques; Reachability problems; Petri nets.

1 Introduction

We describe a procedure to automatically prove *polyhedral abstractions* between pairs of parametric Petri nets. Polyhedral abstraction [2, 6] is a new type of equivalence that can be used to establish a "linear relation" between the reachable markings of two Petri nets. Basically, an abstraction is a triplet of the form (N_1, E, N_2), where E is a system of linear constraints between the places of two nets N_1 and N_2. The idea is to preserve enough information in E so that we can rebuild the reachable markings of N_1 knowing only the ones of N_2, and vice versa.

In this context, we use the term *parametric* to stress the fact that we manipulate semilinear sets of markings, meaning sets that can be defined using a Presburger arithmetic formula C. In particular, we reason about parametric nets (N, C), instead of marked nets (N, m_0), with the intended meaning that all markings satisfying C are potential initial markings of N. We also define an extended notion of polyhedral equivalence between parametric nets, denoted $(N_1, C_1) \cong_E (N_2, C_2)$, whereas our original definition [1] was between marked nets only (see Definition 2.1).

We show that given a valid equivalence statement $(N_1, C_1) \cong_E (N_2, C_2)$, it is possible to derive a Presburger formula, in a constructive way, whose satisfaction implies that the equivalence holds. We implemented this procedure on top of an SMT-solver for Linear Integer Arithmetic (LIA) and show that our approach is applicable in practice (Sect. 7). Our method is only a semi-procedure though, since there are two possible outcomes when the equivalence does not hold: either we can generate a formula that is unsound, or our procedure does not terminate,

and each of these outcomes provide useful information on why the equivalence does not hold.

This decidability result is not surprising, since most equivalence problems on Petri nets are undecidable [15, 16]. If anything, it makes the fact that we may often translate our problem into Presburger arithmetic quite remarkable. Indeed, polyhedral abstraction is by essence related with the *marking equivalence* problem, which amounts to decide if two Petri nets with the same set of places have the same reachable markings; a problem proved undecidable by Hack [17]. Also, polyhedral equivalence entails trace equivalence, another well-known undecidable equivalence problem when we consider general Petri nets [17, 18].

Description of our Approach and Related Works.

We introduced the concept of polyhedral abstraction as a way to solve reachability problems more efficiently. We applied this approach to several problems: originally for model-counting, that is to count the number of reachable markings of a net [12, 13]; then to check reachability formulas and to find inductive invariants [1, 2]; and finally to speed-up the computation of concurrent places (places that can be marked simultaneously in a reachable marking) [5, 6]. We implemented our approach in two symbolic model-checkers developed by our team: Tedd, a tool based on Hierarchical Set Decision Diagrams (SDD) [25], part of the `Tina` toolbox [22]; and `SMPT` [21, 3], an SMT-based model-checker focused on reachability problems [4].

In each case our approach can be summarized as follows. We start from an initial net (N_1, C_1) and derive a polyhedral abstraction $(N_1, C_1) \cong_E (N_2, C_2)$ by applying a set of *abstraction laws* in an iterative and compositional way. Finally, we solve a reachability problem about N_1 by transforming it into a reachability problem on net N_2, which should hopefully be easier to check. A large number of the laws we implement in our tools derive from structural reduction rules [11], or are based on the elimination of redundant places and transitions, with the goal to obtain a "reduced" net N_2 that is smaller than N_1.

We also implement several other kinds of abstraction rules—often subtler to use and harder to prove correct—which explains why we want machine checkable proofs of equivalence. For instance, some of our rules are based on the identification of Petri nets subclasses in which the set of reachable markings equals the set of potentially reachable ones, a property we call the PR-R equality in [19, 20]. We use this kind of rules in the example of the "SwimmingPool" model of Fig. 8, a classical example of Petri net often used in case studies (see e.g. [10]).

We give an example of a basic abstraction law in Fig. 1, with an instance of rule (CONCAT) that allows us to fuse two places connected by a direct, silent transition. We give another example with (MAGIC), in Fig. 2, which illustrates a more complex agglomeration rule, and refer to other examples in Sect. 7.

The parametric net (N_1, C_1) (left of Fig. 1) has a condition which entails that place y_2 should be empty initially ($y_2 = 0$), whereas net (N_2, C_2) has a trivial constraint, which can be interpreted as simply $x \geqslant 0$. We can show (see Sect. 3) that nets N_1 and N_2 are E-equivalent, which amounts to prove that any marking $(y_1 : k_1, y_2 : k_2)$ of N_1, reachable by firing a transition sequence σ, can be associated with the marking $(x : k_1 + k_2)$ of N_2, also reachable by the same firing sequence. Actually, we prove that this equivalence is sound when no

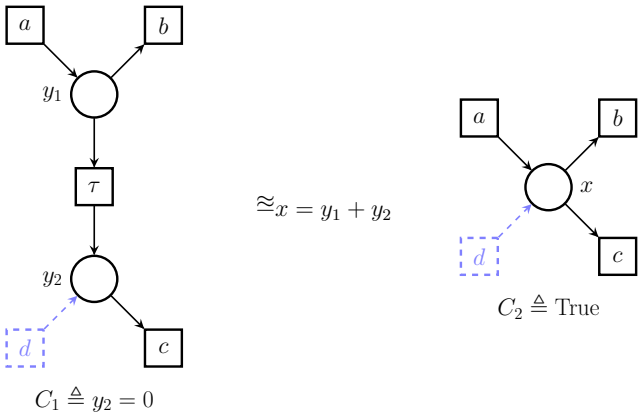


Figure 1: Equivalence rule (CONCAT), $(N_1, C_1) \cong_E (N_2, C_2)$, between nets N_1 (left) and N_2 (right), for the relation $E \triangleq (x = y_1 + y_2)$.

transition can input a token directly into place y_2 of N_1. This means that the rule is correct in the absence of the dashed transition (with label d), but that our procedure should flag the rule as unsound when transition d is present.

The results presented in this paper provide an automated technique for proving the correctness of polyhedral abstraction laws. This helps us gain more confidence on the correctness of our tools and is also useful if we want to add new abstraction rules. Indeed, up until now, all our rules where proven using "manual theorem proving", which can be tedious and error-prone. Incidentally,

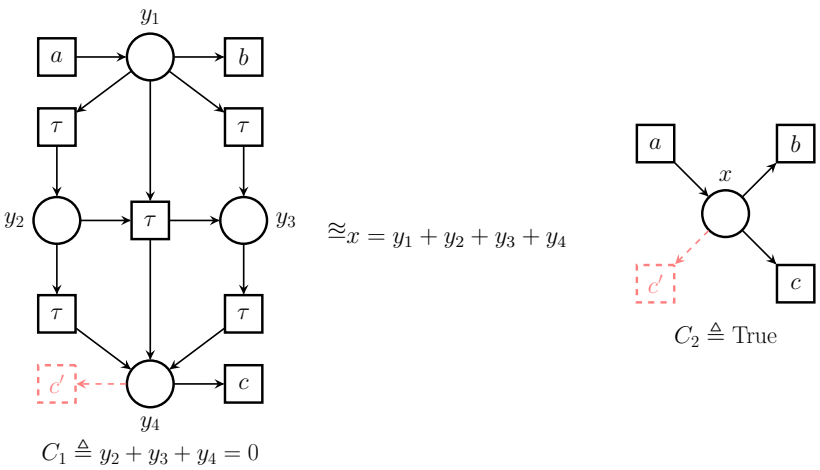


Figure 2: Equivalence rule (MAGIC).

the theory we developed for this paper also helped us gain a better understanding of the constraints necessary when designing new abstraction laws. A critical part of our approach relies on the ability, given a Presburger predicate C, to encode the set of markings reachable from C by firing only silent transitions, that we denote $\tau^\star_C$ in the following. Our approach draws a connection with previous works [7, 8, 24] that study the class of Petri nets that have Presburger-

definable reachability sets; also called *flat nets*. We should also make use of a tool implemented by the same authors, called `FAST`, which provides a method for representing the reachable set of flat nets. Basically, we gain the insight that polyhedral abstraction provides a way to abstract away (or collapse) the subparts of a net that are flattable. Note that our approach may work even though the reachability set of the whole net is not semilinear, since only the part that is abstracted must be flattable. We also prove that when $(N_1, C_1) \approxeq_E (N_2, C_2)$ then necessarily the sets $\tau^\star_{C_1}$ and $\tau^\star_{C_2}$ are semilinear.

Outline and Contributions.

The paper is organized as follows. We define our central notion of *parametric polyhedral abstraction* in Sect. 3 and prove several of its properties in Sect. 6. In particular, we prove that polyhedral abstraction is a congruence, and that it is preserved when "duplicating labeled transitions". These properties mean that every abstraction law we prove can be safely applied in every context, and that each law can be used as a "rule schema". Our definition relies on a former notion of polyhedral equivalence, that we recall in Sect. 2, together with a quick overview of our notations. We describe our proof procedure in Sect. 4, which is defined as the construction of a set of four *core requirements*, each expressed as separate quantified LIA formulas. A key ingredient in this translation is to build a predicate, $\tau^\star_C$, which encodes the markings reachable by firing only the silent transitions of a net. We defer the definition of this predicate until Sect. 5, where we show how it can be obtained using the output of the `FAST` tool. We also describe a method for automatically certifying that the resulting predicate is sound, which means that we do not have to trust the soundness of any outside software component, except SMT solvers. We conclude by presenting the results obtained with a new tool implementing our approach, called `Reductron`, on some concrete examples.

2 Petri Nets and Polyhedral Abstraction

A Petri net is a tuple $(P, T, \mathrm{Pre}, \mathrm{Post})$, where $P = p_1, \ldots, p_n$ is a finite set of places, $T = t_1, \ldots, t_k$ is a finite set of transitions (disjoint from P), and $\mathrm{Pre} : T \to (P \to \mathbb{N})$ and $\mathrm{Post} : T \to (P \to \mathbb{N})$ are the pre- and post-condition functions (also known as the flow functions of the net). A state of a net, also called a marking, is a mapping $m : P \to \mathbb{N}$ (also denoted $\mathbb{N}^P$) that assigns a number of tokens, $m(p)$, to each place p in P. A marked net (N, m_0) is a pair consisting of a net, N, and an initial marking, m_0. In the following, we will often consider that each transition is labeled with a symbol from an alphabet Σ. In this case, we assume that a net is associated with a labeling function $l : T \to \Sigma \cup \{\tau\}$, where τ is a special symbol for the silent action. Every net has a default labeling function, l_N, such that $\Sigma = T$ and $l_N(t) = t$ for every transition $t \in T$.

A transition $t \in T$ is enabled at a marking $m \in \mathbb{N}^P$ if $m(p) \geqslant \mathrm{Pre}(t, p)$ for all places $p \in P$, which we also write $m \geqslant \mathrm{Pre}(t)$, where $\geqslant$ represents component-wise comparison of the markings. A marking $m' \in \mathbb{N}^P$ is reachable from a marking $m \in \mathbb{N}^P$ by firing transition t, denoted $(N, m) \xrightarrow{t} (N, m')$ or simply $m \xrightarrow{t} m'$ when N is obvious from the context, if: (1) transition t is enabled at

m, and (2) $m' = m - \mathrm{Pre}(t) + \mathrm{Post}(t)$. A firing sequence $\varrho = t_1, \ldots, t_n \in T^*$ can be fired from m, denoted $(N, m) \overset{\varrho}{\Rightarrow} (N, m')$ or simply $m \overset{\varrho}{\Rightarrow} m'$, if there exist markings $m_0, \ldots, m_n$ such that $m = m_0$, $m' = m_n$, and $m_i \xrightarrow{t_{i+1}} m_{i+1}$ for all $i < n$. We denote $R(N, m_0)$ the set of markings reachable from m_0 in N.

We can lift any labeling function $l : T \to \Sigma \cup \{\tau\}$ to a mapping of sequences from T^* to Σ^*. Specifically, we define inductively $l(\varrho.t) = l(\varrho)$ if $l(t) = \tau$ and $l(\varrho.t) = l(\varrho).l(t)$ otherwise, where . is the concatenation operator, and $l(\epsilon) = \epsilon$, where ϵ is the empty sequence, verifying $\epsilon.\sigma = \sigma.\epsilon = \sigma$ for any $\sigma \in \Sigma^*$. Given a sequence of labels $\sigma \in \Sigma^*$, we write $(N, m) \overset{\sigma}{\Rightarrow} (N, m')$ if there exists a firing sequence $\varrho \in T^*$ such that $(N, m) \overset{\varrho}{\Rightarrow} (N, m')$ and $\sigma = l(\varrho)$. In this case, σ is referred to as an *observable sequence* of the marked net (N, m). In some cases, we have to consider firing sequences that must not finish with τ transitions. Hence, we define a relation $(N, m) \overset{\sigma\rangle}{\Longrightarrow} (N, m')$, written simply $m \overset{\sigma\rangle}{\Longrightarrow} m'$, as follows:

- $(N, m) \overset{\epsilon\rangle}{\Longrightarrow} (N, m)$ holds for all marking m.
- $(N, m) \overset{\sigma.a\rangle}{\Longrightarrow} (N, m')$ holds for any markings m, m' and $a, \sigma \in \Sigma \times \Sigma^*$, if there exists a marking m'' and a transition t such that $l(t) = a$ and $(N, m) \overset{\sigma}{\Rightarrow} (N, m'') \xrightarrow{t} (N, m')$.

It is immediate that $m \overset{\sigma\rangle}{\Longrightarrow} m'$ implies $m \overset{\sigma}{\Rightarrow} m'$. Note the difference between $m \overset{\epsilon}{\Rightarrow} m'$, which stands for any sequence of τ transitions, and $m \overset{\epsilon\rangle}{\Longrightarrow} m'$, which implies $m = m'$ (the sequence is empty).

We use the standard graphical notation for nets, where places are depicted as circles and transitions as squares such as the nets displayed in Fig. 1.

Polyhedral Abstraction.

We define an equivalence relation that can be used to describe a linear dependence between the markings of two different nets, N_1 and N_2. Assume V is a set of places $p_1, \ldots, p_n$, considered as variables, and let m be a mapping in $V \to \mathbb{N}$. We define $\underline{m}$ as a linear formula, whose unique model in $\mathbb{N}^V$ is m, defined as $\underline{m} \triangleq \bigwedge\{x = m(x) \mid x \in V\}$. By extension, given a Presburger formula E, we say that m is a (partial) solution of E if the formula $E \wedge \underline{m}$ is consistent. Equivalently, we can view $\underline{m}$ as a substitution, where each variable $x \in V$ is substituted by $m(x)$. Indeed, the formula $F\{m\}$ (the substitution $\underline{m}$ applied to F) and $F \wedge \underline{m}$ admit the same models. Given two mappings $m_1 \in \mathbb{N}^{V_1}$ and $m_2 \in \mathbb{N}^{V_2}$, we say that m_1 and m_2 are *compatible* when they have equal values on their shared domain: $m_1(x) = m_2(x)$ for all x in $V_1 \cap V_2$. This is a necessary and sufficient condition for the system $\underline{m_1} \wedge \underline{m_2}$ to be consistent. Finally, if V is the set of free variables of $\underline{m_1}$, $\underline{m_2}$, and the free variables of E are included in V, we say that m_1 and m_2 are related up-to E, denoted $m_1 \equiv_E m_2$, when $E \wedge \underline{m_1} \wedge \underline{m_2}$ is consistent.

$$m_1 \equiv_E m_2 \quad \Leftrightarrow \quad \exists m \in \mathbb{N}^V \ . \ m \models E \wedge \underline{m_1} \wedge \underline{m_2} \tag{1}$$

This relation defines an equivalence between markings of two different nets ($\equiv_E \ \subseteq \mathbb{N}^{P_1} \times \mathbb{N}^{P_2}$) and, by extension, can be used to define an equivalence between nets themselves, that is called *polyhedral equivalence* in [2, 5], where all reachable markings of N_1 are related to reachable markings of N_2 (and conversely), as explained next.

Definition 2.1 (E-abstraction)**.** *Assume* $N_1 = (P_1, T_1, \mathrm{Pre}_1, \mathrm{Post}_1)$ *and* $N_2 = (P_2, T_2, \mathrm{Pre}_2, \mathrm{Post}_2)$ *are two Petri nets, and* E *a Presburger formula whose free variables are included in* $P_1 \cup P_2$*. We say that the marked net* (N_2, m_2) *is an* E*-abstraction of* (N_1, m_1)*, denoted* $(N_1, m_1) \sqsubseteq_E (N_2, m_2)$*, if and only if:*

(A1) *The initial markings are compatible with* E*, meaning* $m_1 \equiv_E m_2$*.*

(A2) *For all observable sequences* $(N_1, m_1) \overset{\sigma}{\Rightarrow} (N_1, m_1')$ *in* N_1*, there is at least one marking* m_2' *over* P_2 *such that* $m_1' \equiv_E m_2'$*, and for all markings* m_2' *over* P_2 *such that* $m_1' \equiv_E m_2'$ *we have* $(N_2, m_2) \overset{\sigma}{\Rightarrow} (N_2, m_2')$*.*

We say that (N_1, m_1) *is* E*-equivalent to* (N_2, m_2)*, denoted* $(N_1, m_1) \equiv_E (N_2, m_2)$*, when we have both* $(N_1, m_1) \sqsubseteq_E (N_2, m_2)$ *and* $(N_2, m_2) \sqsubseteq_E (N_1, m_1)$*.*

By definition, given an equivalence statement $(N_1, m_1) \equiv_E (N_2, m_2)$, then for every marking m_2' reachable in N_2, the set of markings of N_1 consistent with $E \wedge \underline{m_2'}$ is non-empty (condition (A2)). This defines a partition of the reachable markings of (N_1, m_1) into a union of "convex sets"—hence the name polyhedral abstraction—each associated to one (at least) reachable marking in N_2.

Although E-abstraction looks like a simulation, it is not, since the pair of reachable markings m_1', m_2' from the definition does not satisfy $(N_1, m_1') \sqsubseteq_E (N_2, m_2')$ in general. This relation $\sqsubseteq_E$ is therefore broader than a simulation, but suffices for our primary goal, that is Petri net reduction. Of course, $\equiv_E$ is not a bisimulation either. It is also quite simple to show that checking E-abstraction equivalence is undecidable in general.

Theorem 2.1 (Undecidability of E-equivalence)**.** *The problem of checking whether a statement* $(N_1, m_1) \equiv_E (N_2, m_2)$ *is valid is undecidable.*

Proof. Assume that N_1 and N_2 are two nets with the same set of places, such that all transitions are silent. Then $(N_1, m_1) \equiv_{\mathrm{True}} (N_2, m_2)$, an E-abstraction for the trivial constraint $E \triangleq \mathrm{True}$, entails that (N_1, m_1) and (N_2, m_2) must have the same reachability set. This property is known as *marking equivalence* and is undecidable [17]. □

3 Parametric Reduction Rules and Equivalence

E-abstraction is defined on marked nets (Definition 2.1), thus the reduction rules defined in [1, 2], which are E-abstraction equivalences, mention marked nets as well. Their soundness was proven manually, using constrained parameters for initial markings. Such constraints on markings are called *coherency constraints*.

Coherency Constraints.

We define a notion of *coherency constraint*, C, that must hold not only in the initial state, but also in a sufficiently large subset of reachable markings. We have already seen an example with the constraint $C_1 \triangleq y_2 = 0$ used in rule (CONCAT). Without the use of C_1, rule (CONCAT) would be unsound since net N_2 (right of Fig. 1) could fire transition b more often than its counterpart, N_1.

Since C is a predicate on markings, we equivalently consider it as a subset of markings or as a logic formula, so that we may equivalently write $m \models C$ or $m \in C$ to indicate that $C(m)$ is true.

Definition 3.1 (Coherent Net)**.** *Given a Petri net N and a predicate C on markings, we say that N satisfies the coherency constraint C, or equivalently, that (N, C) is a coherent net, if and only if for all firing sequences $m \stackrel{\sigma}{\Rightarrow} m'$ with $m \in C$, we have*

$$\exists m'' \in C \;.\; m \stackrel{\sigma\rangle}{\Longrightarrow} m'' \wedge m'' \stackrel{\epsilon}{\Rightarrow} m'$$

Intuitively, if we consider that all τ transitions are irreversible choices, then we can define a partial order on markings with $m < m'$ whenever $m \stackrel{\tau}{\rightarrow} m'$ holds. Then, markings satisfying the coherency constraint C must be minimal with respect to this partial order.

In this paper, we wish to prove automatically the soundness of a given reduction rule. A reduction rule basically consists of two nets with their coherency constraints, and a Presburger relation between markings.

Definition 3.2 (Parametric Reduction Rule)**.** *A parametric reduction rule is written $(N_1, C_1) >_E (N_2, C_2)$, where (N_1, C_1) and (N_2, C_2) are both coherent nets, and C_1, C_2, and E are Presburger formulas whose free variables are in $P_1 \cup P_2$.*

A given reduction rule $(N_1, C_1) >_E (N_2, C_2)$ is a candidate, which we will analyze to prove its soundness: is it an E-abstraction equivalence?

Our analysis relies on a richer definition of E-abstraction, namely parametric E-abstraction (Definition 3.3, next), which includes the coherency constraints C_1, C_2. Parametric E-abstraction entails E-abstraction for each instance of its parameters (Theorem 3.1, below). Essentially, for any sequence $m_1 \stackrel{\sigma}{\Rightarrow} m_1'$ with $m_1 \in C_1$, there exists a marking m_2' such that $m_1' \equiv_E m_2'$; and for every marking $m_2 \in C_2$ compatible with m_1, i.e., $m_1 \equiv_E m_2$, all markings m_2' compatible with m_1' (i.e., $m_1' \equiv_E m_2'$) can be reached from m_2 by the same observable sequence σ.

To ease the presentation, we define the notation

$$m_1 \langle C_1 E C_2 \rangle m_2 \triangleq m_1 \models C_1 \wedge m_1 \equiv_E m_2 \wedge m_2 \models C_2 \tag{2}$$

Definition 3.3 (Parametric E-abstraction)**.** *Assume $(N_1, C_1) >_E (N_2, C_2)$ is a parametric reduction rule. We say that (N_2, C_2) is a parametric E-abstraction of (N_1, C_1), denoted $(N_1, C_1) \preceq_E (N_2, C_2)$ if and only if:*

(S1) *For all markings m_1 satisfying C_1 there exists a marking m_2 such that $m_1 \langle C_1 E C_2 \rangle m_2$.*

(S2) *For all firing sequences $m_1 \stackrel{\epsilon}{\Rightarrow} m_1'$ and all markings m_2, we have $m_1 \equiv_E m_2$ implies $m_1' \equiv_E m_2$.*

(S3) *For all firing sequences $m_1 \stackrel{\sigma}{\Rightarrow} m_1'$ and all marking pairs m_2, m_2', if $m_1 \langle C_1 E C_2 \rangle m_2$ and $m_1' \equiv_E m_2'$ then we have $m_2 \stackrel{\sigma}{\Rightarrow} m_2'$.*

We say that (N_1, C_1) and (N_2, C_2) are in parametric E-equivalence, denoted $(N_1, C_1) \approx_E (N_2, C_2)$, when we have both $(N_1, C_1) \preceq_E (N_2, C_2)$ and $(N_2, C_2) \preceq_E (N_1, C_1)$.

Condition (S1) corresponds to the solvability of the Presburger formula E with respect to the marking predicates C_1 and C_2. Condition (S2) ensures that silent transitions of N_1 are abstracted away by the formula E, and are therefore

invisible to N_2. Condition (S3) follows closely condition (A2) of the standard E-abstraction equivalence.

Note that equivalence $\approxeq$ is not a bisimulation, in the same way that $\equiv$ from Definition 2.1. It is defined only for observable sequences starting from states satisfying the coherency constraint C_1 of N_1 or C_2 of N_2, and so this relation is usually not true on every pair of equivalent markings $m_1 \equiv_E m_2$.

Instantiation Law.

Parametric E-abstraction implies E-abstraction for every instance pair satisfying the coherency constraints C_1, C_2.

Theorem 3.1 (Parametric E-abstraction Instantiation)**.** *Assume* $(N_1, C_1) \preceq_E (N_2, C_2)$ *is a parametric* E*-abstraction. Then for every pair of markings* m_1, m_2*,* $m_1 \langle C_1 E C_2 \rangle m_2$ *implies* $(N_1, m_1) \sqsubseteq_E (N_2, m_2)$*.*

Proof. Consider $(N_1, C_1) \preceq_E (N_2, C_2)$, a parametric E-abstraction, and m_1, m_2 such that $m_1 \langle C_1 E C_2 \rangle m_2$ holds. By definition of $m_1 \langle C_1 E C_2 \rangle m_2$, see Equation (2), condition (A1) of Definition 2.1 is immediately satisfied. We show (A2) by considering an observable sequence $(N_1, m_1) \stackrel{\sigma}{\Rightarrow} (N_1, m_1')$. Since m_1 satisfies the coherency constraint C_1, we get from Definition 3.1 a marking $m_1'' \in C_1$ such that $m_1 \stackrel{\sigma\rangle}{\Longrightarrow} m_1'' \stackrel{\epsilon}{\Rightarrow} m_1'$ holds. By applying (S1) to m_1'', we get a marking m_2' such that $m_1'' \langle C_1 E C_2 \rangle m_2'$ holds, which implies $m_1'' \equiv_E m_2'$. Then, by applying (S2) to $m_1'' \stackrel{\epsilon}{\Rightarrow} m_1'$, we obtain the expected result $m_1' \equiv_E m_2'$. Finally, for all markings m_2' such that $m_1' \equiv_E m_2'$, we conclude $m_2 \stackrel{\sigma}{\Rightarrow} m_2'$ from (S3). Condition (A2) is proved, hence $(N_1, m_1) \sqsubseteq_E (N_2, m_2)$ holds. □

4 Automated Proof Procedure

Our automated proof procedure receives a candidate reduction rule (Definition 3.2) as input, and has three possible outcomes: (i) the candidate is proven sound, congratulations you have established a new parametric E-abstraction equivalence; (ii) the candidate is proven unsound, try to understand why and fix it; or (iii) we cannot conclude, because part of our procedure relies on a semi-algorithm (see Sect. 5) for expressing the set of reachable markings of a flat subnet as a linear constraint.

Given the candidate reduction rule, the procedure generates SMT queries, which we call *core requirements* (defined in Sect. 4.2) that are solvable if and only if the candidate is a parametric E-abstraction (Theorems 4.8 and 4.9, Sect. 4.3). We express these constraints into Presburger predicates, so it is enough to use solvers for the theory of formulas on Linear Integer Arithmetic, what is known as LIA in SMT-LIB [9]. We illustrate the results given in this section using a diagram (Fig. 3) that describe the dependency relations between conditions (S1), (S2), (S3) and their encoding as core requirements.

4.1 Presburger Encoding of Petri Net Semantics

We start by defining a few formulas that ease the subsequent expression of core requirements. This will help with the most delicate point of our encoding, which relies on how to encode sequences of transitions. Note that the coherency constraints of reduction rules are already defined as such.

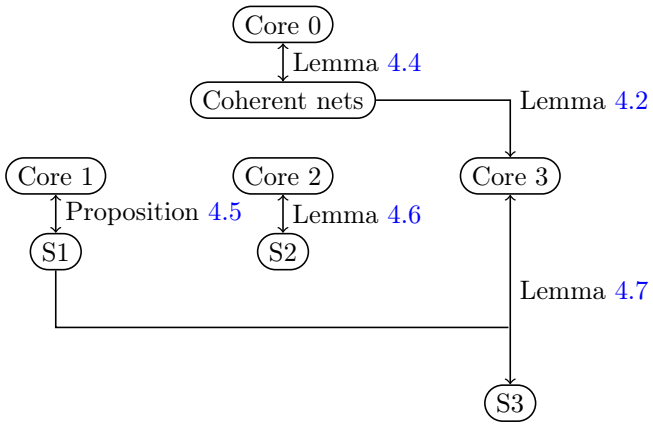


Figure 3: Detailed dependency relations.

In the following, we use $\boldsymbol{x}$ for the vector of variables $(x_1, \ldots, x_n)$, corresponding to the places $p_1, \ldots, p_n$ of P, and $F(\boldsymbol{x})$ for a formula whose variables are included in $\boldsymbol{x}$. We say that a mapping m of $\mathbb{N}^P$ is a *model* of F, denoted $m \models F$, if the ground formula $F(m) = F(m(p_1), \ldots, m(p_n))$ is true. Hence, we can also interpret F as a predicate over markings. Finally, we define the semantics of F as the set $[\![F]\!] = \{m \in \mathbb{N}^P \mid m \models F\}$. As usual, we say that a predicate F is *valid*, denoted $\models F$, when all its interpretations are true ($[\![F]\!] = \mathbb{N}^P$). In order to keep track of fired transitions in our encoding, and without any loss of generality we assume that our alphabet of labels Σ is a subset of the natural numbers ($\Sigma \subset \mathbb{N}^*$), except 0 that is reserved for τ.

We define next a few Presburger formulas that express properties on markings of a net N. For instance, Equation (3) below defines the predicate ENBL_t, for a given transition t, which corresponds exactly to the markings that enable t. We also define a linear predicate $\mathrm{T}(\boldsymbol{x}, \boldsymbol{x'}, a)$ that describes the relation between the markings before ($\boldsymbol{x}$) and after ($\boldsymbol{x'}$) firing a transition with label a. With this convention, formula $\mathrm{T}(m, m', a)$ holds if and only if $m \xrightarrow{t} m'$ holds for some transition t such that $l(t) = a$ (which implies $a \neq 0$).

$$\mathrm{ENBL}_t(\boldsymbol{x}) \triangleq \textstyle\bigwedge_{i \in 1..n}(x_i \geqslant \mathrm{Pre}(t, p_i)) \tag{3}$$

$$\Delta_t(\boldsymbol{x}, \boldsymbol{x'}) \triangleq \textstyle\bigwedge_{i \in 1..n}(x'_i = x_i + \mathrm{Post}(t, p_i) - \mathrm{Pre}(t, p_i)) \tag{4}$$

$$\mathrm{T}(\boldsymbol{x}, \boldsymbol{x'}, a) \triangleq \textstyle\bigvee_{t \in T}(\mathrm{ENBL}_t(\boldsymbol{x}) \wedge \Delta_t(\boldsymbol{x}, \boldsymbol{x'}) \wedge a = l(t)) \tag{5}$$

We admit the following, for all markings m, m' and label a:

$$\models T(m, m', a) \iff \exists t \,.\, m \xrightarrow{t} m' \wedge l(t) = a \tag{6}$$

In order to define the core requirements, we additionally require a predicate $\tau^*_C(\boldsymbol{x}, \boldsymbol{x'})$ encoding the markings reachable by firing any sequence of silent transitions from a state satisfying the coherency constraint C. And so, the following constraint must hold:

$$\models m \in C \implies (\tau^*_C(m, m') \iff m \xRightarrow{\epsilon} m') \tag{7}$$

Since $m \overset{\epsilon}{\Rightarrow} m'$ may fire an arbitrary number of silent transitions τ, the predicate τ_C is not guaranteed to be expressible as a Presburger formula in the general case. Yet, in Sect. 5, we characterize the Petri nets for which τ_C can be expressed in Presburger logic, which include all the polyhedral reductions that we meet in practice (we explain why).

Thanks to this predicate, we define the formula $\acute{T}_C(\boldsymbol{x}, \boldsymbol{x'}, a)$ encoding the reachable markings from a marking satisfying the coherency constraint C, by firing any number of silent transitions, followed by a transition labeled with a. Then, we define $\hat{T}$ which extends $\acute{T}$ with any number of silent transitions after a and also allows for only silent transitions (no transition a).

$$\acute{T}_C(\boldsymbol{x}, \boldsymbol{x'}, a) \triangleq \exists \boldsymbol{x''} \ . \ \tau_C^*(\boldsymbol{x}, \boldsymbol{x''}) \wedge T(\boldsymbol{x''}, \boldsymbol{x'}, a) \tag{8}$$

$$\hat{T}_C(\boldsymbol{x}, \boldsymbol{x'}, a) \triangleq \left(\exists \boldsymbol{x_1} \ . \ \acute{T}_C(\boldsymbol{x}, \boldsymbol{x_1}, a) \wedge C(\boldsymbol{x_1}) \wedge \tau_C^*(\boldsymbol{x_1}, \boldsymbol{x'})\right) \tag{9}$$

$$\vee \ (a = 0 \wedge \tau_C^*(\boldsymbol{x}, \boldsymbol{x'})) \tag{10}$$

Lemma 4.1. *For any markings m, m' and label a such that $m \in C$, we have $\models \acute{T}_C(m, m', a)$ if and only if $m \overset{a\rangle}{\Longrightarrow} m'$ holds.*

Proof. We show both directions separately.

- Assume $m \overset{a\rangle}{\Longrightarrow} m'$. By definition, this implies that there exists m'' and a transition t such that $l(t) = a$ and $m \overset{\epsilon}{\Rightarrow} m'' \overset{t}{\rightarrow} m'$. Therefore, $\tau_C^*(m, m'')$ is valid by (7), and $T(m'', m', a)$ is valid by (6), hence the expected result $\models \acute{T}_C(m, m', a)$.
- Conversely, assume $\acute{T}_C(m, m', a)$ is valid. Then, by (8) there exists a marking m'' such that both $\tau_C^*(m, m'')$ and $T(m'', m', a)$ are valid. From (7), we get $m \overset{\epsilon}{\Rightarrow} m''$, and (6) implies $\exists t \ , \ m'' \overset{t}{\rightarrow} m' \wedge l(t) = a$. Thus, $m \overset{\epsilon}{\Rightarrow} m'' \overset{t}{\rightarrow} m'$, that is the expected result $m \overset{a\rangle}{\Longrightarrow} m'$.

□

Lemma 4.2. *Given a coherent net (N, C), for any markings m, m' such that $m \in C$ and $a \in \Sigma \cup \{0\}$, we have $\models \hat{T}_C(m, m', a)$ if and only if either $m \overset{\epsilon}{\Rightarrow} m'$ and $a = 0$, or $m \overset{a}{\Rightarrow} m'$.*

Proof. We show both directions separately.

- Assume $m \overset{\epsilon}{\Rightarrow} m'$ and $a = 0$, then $\tau_C^*(m, m')$ is valid by (7), hence the expected result $\models \hat{T}_C(m, m', a)$ from (10).
- Assume $m \overset{a}{\Rightarrow} m'$. From Definition 3.1 (coherent net), there exists $m'' \in C$ such that $m \overset{a\rangle}{\Longrightarrow} m'' \overset{\epsilon}{\Rightarrow} m'$. Then, we get $\models \acute{T}_C(m, m'', a)$ from Lemma 4.1, and $\models \tau_C^*(m'', m')$ from (7). Consequently, $\hat{T}_C(m, m', a)$ is valid from (9).
- Conversely, assume $\hat{T}_C(m, m', a)$ holds by (10), then $a = 0$ and $\models \tau_C^*(m, m')$, which implies $m \overset{\epsilon}{\Rightarrow} m'$ by (7). This is the expected result.
- Finally, assume $\hat{T}_C(m, m', a)$ holds by (9), then there exists a marking $m'' \in C$ such that $\models \acute{T}_C(m, m'', a)$ and $\models \tau_C^*(m'', m')$. This implies $m \overset{a\rangle}{\Longrightarrow} m'' \overset{\epsilon}{\Rightarrow} m'$ from Lemma 4.1 and (7). This implies the expected result $m \overset{a}{\Rightarrow} m'$.

□

Finally, we denote $\tilde{E}(\boldsymbol{x}, \boldsymbol{y})$ the formula obtained from E where free variables are substituted as follows: place names in N_1 are replaced with variables in $\boldsymbol{x}$, and place names in N_2 are replaced with variables in $\boldsymbol{y}$ (making sure that bound variables of E are renamed to avoid interference). When the same place occurs in both nets, say $p_i^1 = p_j^2$, we also add the equality constraint $(x_i = y_j)$ to $\tilde{E}$ in order to preserve this equality constraint.

4.2 Core Requirements: Parametric E-abstraction Encoding

In order to check conditions (S1)–(S3) of parametric E-abstraction (Definition 3.3), we define a set of Presburger formulas, called *core requirements*, to be verified using an external SMT solver ((Core 1) to (Core 3)). You will find an illustration of these requirements in Figs. 4–7. The satisfaction of these requirements entail the parametric E-abstraction relation. We have deliberately stressed the notations to prove that (N_2, C_2) is a parametric E-abstraction of (N_1, C_1). Of course, each constraint must be checked in both directions to obtain the equivalence. Also, to not overload the notations, we assume that the transition relations are clear in the context if they belong to N_1 or N_2.

Verifying that a Net is Coherent.

The first step consists in verifying that both nets N_1 and N_2 satisfy their coherency constraints C_1 and C_2 (the coherency constraint is depicted in Figure 4). We recall Definition 3.1:

Definition (Coherent Net)**.** *For all firing sequence $m \stackrel{\sigma}{\Rightarrow} m'$ with $m \in C$, there exists a marking m'' satisfying C such that $m \stackrel{\sigma\rangle}{\Longrightarrow} m''$ and $m'' \stackrel{\epsilon}{\Rightarrow} m'$.*

We encode a simpler relation, below, with sequences σ of size 1. This relies on the following result:

Lemma 4.3. *(N, C) is coherent if and only if for all firing sequence $m \stackrel{a\rangle}{\Longrightarrow} m'$ with $m \in C$ and $a \in \Sigma$, we have $\exists m'' \in C \;.\; m \stackrel{a\rangle}{\Longrightarrow} m'' \wedge m'' \stackrel{\epsilon}{\Rightarrow} m'$.*

We deliberately consider a firing sequence $m \stackrel{a\rangle}{\Longrightarrow} m'$ (and not $m \stackrel{a}{\Rightarrow} m'$), since the encoding relies only on $\acute{T}_C$ (that is, $\stackrel{a\rangle}{\Longrightarrow}$), not on $\hat{T}_C$ (that is, $\stackrel{a}{\Rightarrow}$).

Proof. The "only if" part is immediate, as a particular case of Definition 3.1 and noting that $m \stackrel{a\rangle}{\Longrightarrow} m'$ implies $m \stackrel{a}{\Rightarrow} m'$. Conversely, assume the property stated in the lemma is true. Then, we show by induction on the size of σ, that Definition 3.1 holds for any σ. Note that the base case $\sigma = \epsilon$ always holds, for any net, by taking $m'' = m$. Now, consider a non-empty sequence $\sigma = \sigma'.a$ and $m \stackrel{\sigma'.a}{\Longrightarrow} m'$ with $m \in C$. By definition, there exists m_1 and m_2 such that $m \stackrel{\sigma'}{\Longrightarrow} m_1 \stackrel{a\rangle}{\Longrightarrow} m_2 \stackrel{\epsilon}{\Rightarrow} m'$. By induction hypothesis, on $m \stackrel{\sigma'}{\Longrightarrow} m_1$, there exists $m_3 \in C$ such that $m \stackrel{\sigma'\rangle}{\Longrightarrow} m_3 \stackrel{\epsilon}{\Rightarrow} m_1$. Therefore, we have $m \stackrel{\sigma'\rangle}{\Longrightarrow} m_3 \stackrel{\epsilon}{\Rightarrow} m_1 \stackrel{a\rangle}{\Longrightarrow} m_2 \stackrel{\epsilon}{\Rightarrow} m'$, which can simply be written $m \stackrel{\sigma'\rangle}{\Longrightarrow} m_3 \stackrel{a\rangle}{\Longrightarrow} m_2 \stackrel{\epsilon}{\Rightarrow} m'$. Using the property stated in the lemma on $m_3 \stackrel{a\rangle}{\Longrightarrow} m_2$, we get a marking $m_4 \in C$ such that $m_3 \stackrel{a\rangle}{\Longrightarrow} m_4 \stackrel{\epsilon}{\Rightarrow} m_2$. Hence, $m \stackrel{\sigma'\rangle}{\Longrightarrow} m_3 \stackrel{a\rangle}{\Longrightarrow} m_4 \stackrel{\epsilon}{\Rightarrow} m_2 \stackrel{\epsilon}{\Rightarrow} m'$ holds, which can be simplified as $m \stackrel{\sigma'.a\rangle}{\Longrightarrow} m_4 \stackrel{\epsilon}{\Rightarrow} m'$. This is the expected result. □

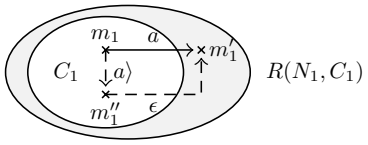


Figure 4: Illustration of (Core 0).

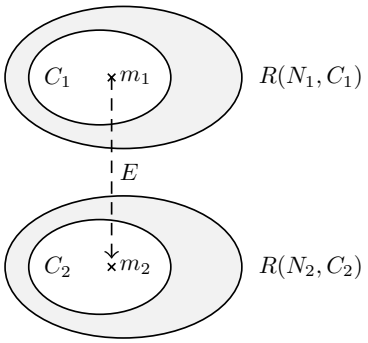


Figure 5: Illustration of (Core 1).

Therefore, we can encode Definition 3.1 using the following formula:

$$\forall \boldsymbol{p}, \boldsymbol{p'}, a \;.\; C(\boldsymbol{p}) \wedge \acute{T}_C(\boldsymbol{p}, \boldsymbol{p'}, a) \implies \exists \boldsymbol{p''} \;.\; C(\boldsymbol{p''}) \wedge \acute{T}_C(\boldsymbol{p}, \boldsymbol{p''}, a) \wedge \tau_C^*(\boldsymbol{p''}, \boldsymbol{p'}) \tag{Core 0}$$

Lemma 4.4. *Given a Petri net N, the constraint (Core 0) is valid if and only if the net satisfies the coherency constraint C.*

Proof. Constraint (Core 0) is an immediate translation of the property stated in Lemma 4.3. □

Given a net N, a constraint C expressed as a Presburger formula, and a formula τ_C^* that captures $\stackrel{\epsilon}{\Rightarrow}$ transitions (as obtained in Sect. 5), we are now able to check automatically that a net (N, C) is coherent. Thus, from now on, we assume that the considered nets (N_1, C_1) and (N_2, C_2) are indeed coherent.

Coherent Solvability.

The first requirement of the parametric E-abstraction relates to the solvability of formula E with regard to the coherency constraint C_1, and is encoded by (Core 1). This requirement ensures that every marking of N_1 satisfying C_1 can be associated to at least one marking of N_2 satisfying C_2. Let us recall (S1), taken from Definition 3.3:

Definition (S1)**.** *For all markings m_1 satisfying C_1 there exists a marking m_2 such that $m_1 \langle C_1 E C_2 \rangle m_2$.*

Condition (S1) is depicted in Figure 5. We propose to encode it by the following Presburger formula:

$$\forall \boldsymbol{x} \;.\; C_1(\boldsymbol{x}) \implies \exists \boldsymbol{y} \;.\; \tilde{E}(\boldsymbol{x}, \boldsymbol{y}) \wedge C_2(\boldsymbol{y}) \tag{Core 1}$$

Since the encoding is immediate, we admit this proposition:

Proposition 4.5. *The constraint (Core 1) is valid if and only if (S1) holds.*

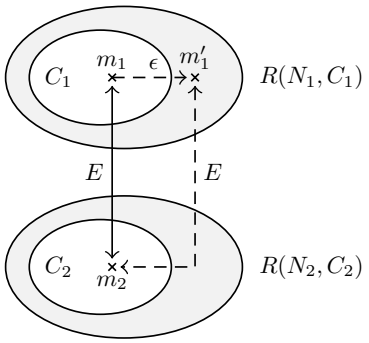


Figure 6: Illustration of (Core 2).

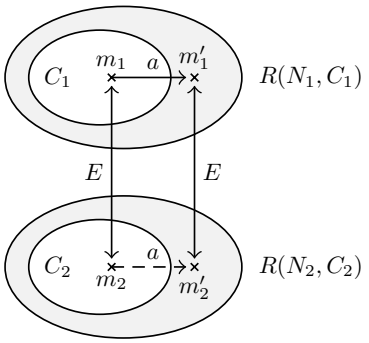


Figure 7: Illustration of (Core 3).

Silent Constraints.

So far, we have focused on the specific case of coherent nets, which refers to intermediate coherent markings. Another notable feature of parametric E-abstractions is the ability to fire any number of silent transitions without altering the solutions of E. In other words, if two markings, m_1 and m_2, are solutions of E, then firing any silent sequence from m_1 (or m_2) will always lead to a solution of $E \wedge m_2$ (or $E \wedge m_1$). This means that silent transitions must be invisible to the other net.

Let us recall (S2), taken from Definition 3.3:

Definition (S2). *For all firing sequences $m_1 \stackrel{\epsilon}{\Rightarrow} m'_1$ and all markings m_2, we have $m_1 \equiv_E m_2$ implies $m'_1 \equiv_E m_2$.*

It actually suffices to show the result for each silent transition $t \in T_1$ taken separately:

Lemma 4.6. *Condition (S2) holds if and only if, for all markings m_1, m_2 such that $m_1 \equiv_E m_2$, and for all $t_1 \in T_1$ such that $l_1(t_1) = \tau$, we have $m_1 \xrightarrow{t_1} m'_1 \implies m'_1 \equiv_E m_2$.*

Proof. The "only if" way is only a particular case of (S2) with a single silent transition t_1. For the "if" way, (S2) is shown from the given property by transitivity. □

Thanks to this result, we encode (S2) by the following core requirement:

$$\forall \boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p'_1} \,.\, \tilde{E}(\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge \tau(\boldsymbol{p_1}, \boldsymbol{p'_1}) \implies \tilde{E}(\boldsymbol{p'_1}, \boldsymbol{p_2}) \qquad \text{(Core 2)}$$

where $\tau(\boldsymbol{x}, \boldsymbol{x'})$ is defined as $\tau(\boldsymbol{x}, \boldsymbol{x'}) \triangleq \bigvee_{t \in T | l(t) = \tau} (\mathrm{ENBL}_t(\boldsymbol{x}) \wedge \Delta_t(\boldsymbol{x}, \boldsymbol{x'}))$

Reachability.

Let us recall the definition of (S3), taken from Definition 3.3:

Definition (S3). *For all firing sequences $m_1 \stackrel{\sigma}{\Rightarrow} m'_1$ and all marking pairs m_2, m'_2, if $m_1 \langle C_1 E C_2 \rangle m_2$ and $m'_1 \equiv_E m'_2$ then we have $m_2 \stackrel{\sigma}{\Rightarrow} m'_2$.*

Condition (S3) mentions sequences σ of arbitrary length. We encode it with a formula dealing only with sequences of length at most 1, thanks to the following result:

Lemma 4.7. *Given a parametric reduction rule* $(N_1, C_1) >_E (N_2, C_2)$ *which satisfies condition (S1), then condition (S3) holds if and only if for all firing sequence* $m_1 \stackrel{\sigma}{\Rightarrow} m_1'$ *with* $\sigma = \epsilon$ *or* $\sigma = a$ *with* $a \in \Sigma$*, and all markings* m_2, m_2'*, we have* $m_1 \langle C_1 E C_2 \rangle m_2 \wedge m_1' \equiv_E m_2' \implies m_2 \stackrel{\sigma}{\Rightarrow} m_2'$.

Proof. The given property is necessary as a particular case of (S3) taking $\sigma = a$ or $\sigma = \epsilon$. Conversely, assume the given property holds. We show by induction on the size of σ that (S3) holds for any sequence σ. The base cases $\sigma = a$ and $\sigma = \epsilon$ are ensured by hypothesis. Now, consider a non-empty sequence $\sigma = \sigma'.a$, and $m_1 \stackrel{\sigma}{\Rightarrow} m_1'$ (i), as well as markings m_2, m_2' such that $m_1 \langle C_1 E C_2 \rangle m_2$ and $m_1' \equiv_E m_2'$ holds. We have to show $m_2 \stackrel{\sigma}{\Rightarrow} m_2'$. From (i), we have $m_1 \stackrel{\sigma'.a}{\Longrightarrow} m_1'$, that is, there exists a marking u_1 such that $m_1 \stackrel{\sigma'}{\Longrightarrow} u_1 \stackrel{a}{\Rightarrow} m_1'$ (ii). By Definition 3.1, there exists $u_1' \in C_1$ such that $m_1 \stackrel{\sigma'}{\Longrightarrow} u_1' \stackrel{\epsilon}{\Rightarrow} u_1$ (iii). Also, by condition (S1), there exists a marking u_2' of N_2 such that $u_1' \langle C_1 E C_2 \rangle u_2'$, which implies $u_1' \equiv_E u_2'$ (iv). Hence, by induction hypothesis on $m_1 \stackrel{\sigma'}{\Longrightarrow} u_1'$, we have $m_2 \stackrel{\sigma'}{\Longrightarrow} u_2'$ (α) From (iii) and (ii), we get $u_1' \stackrel{a}{\Rightarrow} m_1'$ (v). Applying the property of the lemma on (iv) and (v), we get $u_2' \stackrel{a}{\Rightarrow} m_2'$ (β). Combining (α) and (β) leads to $m_2 \stackrel{\sigma'.a}{\Longrightarrow} m_2'$, that is the expected result $m_2 \stackrel{\sigma}{\Rightarrow} m_2'$. □

Thanks to Lemma 4.7, we can encode (S3) by the following formula:

$$\forall \boldsymbol{p_1}, \boldsymbol{p_2}, a, \boldsymbol{p_1'}, \boldsymbol{p_2'} \;.\; \langle C_1 E C_2 \rangle (\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge \hat{T}_{C_1}(\boldsymbol{p_1}, \boldsymbol{p_1'}) \wedge \tilde{E}(\boldsymbol{p_1'}, \boldsymbol{p_2'}) \implies \hat{T}_{C_2}(\boldsymbol{p_2}, \boldsymbol{p_2'}) \qquad \text{(Core 3)}$$

4.3 Global Procedure

In this section, we consider the full process for proving parametric E-abstraction. We demonstrate that verifying requirements (Core 0) to (Core 3) is sufficient for obtaining a sound abstraction (Th. 4.8). We also prove that these conditions are necessary (Th. 4.9).

Theorem 4.8 (Soundness)**.** *Given two nets* N_1, N_2 *and constraints* C_1, C_2 *expressed as Presburger formulas, if core requirement (Core 0) holds for both* (N_1, C_1) *and* (N_2, C_2)*, and if core requirements (Core 1), (Core 2), and (Core 3) are valid, then the rule is a parametric E-abstraction:* $(N_1, C_1) \preceq_E (N_2, C_2)$.

Proof. If (Core 0) holds for (N_1, C_1), then (N_1, C_1) is a coherent net by Lemma 4.4. Similarly for (N_2, C_2). Hence, $(N_1, C_1) >_E (N_2, C_2)$ is a parametric reduction rule. By Proposition 4.5, and since (Core 1) is valid, we get (S1) from Definition 3.3. Similarly, by Lemma 4.6, and since (Core 2) is valid, we get (S2). Finally, (S3) holds by Lemma 4.7 since (Core 3) is valid and since (S1) is known to hold. (S1), (S2), (S3) entail $(N_1, C_1) \preceq_E (N_2, C_2)$ by Definition 3.3. □

The converse also holds:

Theorem 4.9 (Completeness)**.** *Given a parametric E-abstraction* $(N_1, C_1) \preceq_E (N_2, C_2)$*, then core requirements (Core 1), (Core 2), and (Core 3) are valid, and (Core 0) holds for both* (N_1, C_1) *and* (N_2, C_2).

Proof. By hypothesis, conditions (S1), (S2) and (S3) hold and (N_1, C_1) and (N_2, C_2) are coherent nets. Then, Lemma 4.4 implies that (Core 0) holds for both nets. Besides, Proposition 4.5 and Lemmas 4.6 and 4.7 ensure that (Core 1), (Core 2), and (Core 3) are valid. □

Consequently, checking E-abstraction equivalence, i.e., $(N_1, C_1) \cong_E (N_2, C_2)$, amounts to check that SMT formulas (Core 0)-(Core 3) are valid on both nets.

Our approach relies on our ability to express (arbitrarily long) sequences $m \overset{\epsilon}{\Rightarrow} m'$ thanks to a formula $\tau_C^*(\boldsymbol{x}, \boldsymbol{x'})$. This is addressed in the next section.

5 Silent Transition Relation Acceleration

The previous results, including Theorems 4.8 and 4.9, rely on our ability to express the reachability set of silent transitions as a Presburger predicate, denoted τ_C^*. Finding a finite formula τ_C^* that captures an infinite state-space is not granted, since τ-sequences may be of arbitrary length. However, we now show that, since τ transitions must be abstracted away by E in order to define a valid parametric E-equivalence (condition (S2)), and since E is itself a Presburger formula, this implies that τ_C^* corresponds to the reachability set of a *flattable* subnet [24], which is expressible as a Presburger formula too.

We define the *silent reachability set* of a net N from a coherent constraint C as $R_\tau(N, C) \triangleq \{m' \mid m \models C \wedge m \overset{\epsilon}{\Rightarrow} m'\}$. We now want to find a predicate $\tau_C^*(\boldsymbol{x}, \boldsymbol{x'})$ that satisfies the relation:

$$R_\tau(N, C) = \{m' \mid m' \models \exists \boldsymbol{x} \,.\, C(\boldsymbol{x}) \wedge \tau_C^*(\boldsymbol{x}, \boldsymbol{x'})\} \tag{7}$$

In order to express the formula τ_C^*, we first use the tool `FAST` [7], designed for the analysis of infinite systems, and that permits to compute the reachability set of a given Vector Addition System with States (VASS). Note that a Petri net can be transformed to an equivalent VASS with the same reachability set, so the formal presentation of VASS can be skipped. The algorithm implemented in `FAST` is a semi-procedure, for which we have some termination guarantees whenever the net is flattable [8], i.e. its corresponding VASS can be unfolded into a VASS without nested cycles, called a flat VASS. Equivalently, a net N is flattable for some coherent constraint C if its language is flat, that is, there exists some finite sequence $\varrho_1 \dots \varrho_k \in T^*$ such that for every initial marking $m \models C$ and reachable marking m' there is a sequence $\varrho \in \varrho_1^* \dots \varrho_k^*$ such that $m \overset{\varrho}{\Rightarrow} m'$. In short, all reachable markings can be reached by simple sequences, belonging to the language: $\varrho_1^* \dots \varrho_k^*$. Last but not least, the authors stated in Theorem 5.1 from [24] that a net is flattable if and only if its reachability set is Presburger-definable:

Theorem 5.1 ([24])**.** *For every VASS V, for every Presburger set C_{in} of configurations, the reachability set* $\mathrm{ReachV}(C_{in})$ *is Presburger if, and only if, V is flattable from C_{in}.*

As a consequence, `FAST`'s algorithm terminates when its input is Presburger-definable. We show in Theorem 5.2 that given a parametric E-abstraction equivalence $(N_1, C_1) \cong_E (N_2, C_2)$, the silent reachability sets for both nets N_1 and N_2 with their coherency constraints C_1 and C_2 are indeed Presburger-definable – we can even provide the expected formulas. Yet, our computation is complete only if the candidate reduction rule is a parametric E-abstraction equivalence (then, we are able to compute the τ_C^* relation), otherwise `FAST`, and therefore our procedure too, may not terminate.

Theorem 5.2. *Given a parametric E-abstraction equivalence $(N_1, C_1) \approxeq_E (N_2, C_2)$, the silent reachability set $R_\tau(N_1, C_1)$ is Presburger-definable.*

Proof. We prove only the result for (N_1, C_1), the proof for (N_2, C_2) is similar since $\approxeq$ is a symmetric relation. We first propose an expression that computes $R_\tau(N_1, m_1)$ for any marking m_1 satisfying C_1. Consider an initial marking m_1 in C_1. From condition (S1) (solvability of E), there exists a compatible marking m_2 satisfying C_2, meaning $m_1 \langle C_1 E C_2 \rangle m_2$ holds. Now, take a silent sequence $m_1 \overset{\epsilon}{\Rightarrow} m_1'$. From condition (S2) (silent stability), we have $m_1' \equiv_E m_2$. Hence, $R_\tau(N_1, m_1) \subseteq \{m_1' \mid \exists m_2 \,.\, \tilde{E}(m_1, m_2) \wedge \tilde{E}(m_1', m_2)\}$. Conversely, we show that all m_1' solution of $\tilde{E}(m_1', m_2)$ are reachable from m_1. Take m_1' such that $m_1' \equiv_E m_2$. Since we have $m_2 \overset{\epsilon}{\Rightarrow} m_2$, by condition (S3) we must have $m_1 \overset{\epsilon}{\Rightarrow} m_1'$. And finally we obtain $R_\tau(N_1, m_1) = \{m_1' \mid m_1' \models \exists \boldsymbol{p_1}, \boldsymbol{p_2} \,.\, \underline{m_1}(\boldsymbol{p1}) \wedge \tilde{E}(\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge \tilde{E}(\boldsymbol{p_1'}, \boldsymbol{p_2})\}$.

We can generalize this reachability set for all coherent markings satisfying C_1. We first recall its definition, $R_\tau(N_1, C_1) = \{m_1' \mid \exists m_1 \,.\, m_1 \models C_1 \wedge m_1 \overset{\epsilon}{\Rightarrow} m_1'\}$. From condition (S1), we can rewrite this set as $\{m_1' \mid \exists m_1, m_2 \,.\, m_1 \langle C_1 E C_2 \rangle m_2 \wedge m_1 \overset{\epsilon}{\Rightarrow} m_1'\}$ without losing any marking. Finally, thanks to the previous result we get $R_\tau(N_1, C_1) = \{m_1' \mid m_1' \models P\}$ with $P = \exists \boldsymbol{p_1}, \boldsymbol{p_2} \,.\, \langle C_1 E C_2 \rangle(\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge \tilde{E}(\boldsymbol{p_1'}, \boldsymbol{p_2})$ a Presburger formula. Because of the E-abstraction equivalence, (S1) holds in both directions, which gives $\forall \boldsymbol{p_2} \,.\, C_2(\boldsymbol{p_2}) \implies \exists \boldsymbol{p_1} \,.\, \tilde{E}(\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge C_1(\boldsymbol{p_1})$. Hence, P can be simplified into $\exists \boldsymbol{p_2} \,.\, C_2(\boldsymbol{p_2}) \wedge \tilde{E}(\boldsymbol{p_1'}, \boldsymbol{p_2})$.

Note that this expression of $R_\tau(N, C)$ relies on the fact that the equivalence $(N_1, C_1) \approxeq_E (N_2, C_2)$ already holds. Thus, we cannot conclude that a candidate rule is an E-abstraction equivalence by using this formula at once, without the extra validation of `FAST`. □

Verifying FAST Results.

We have shown that `FAST` terminates in case of a correct parametric E-abstraction. We now show that it is possible to check that the predicates $\tau^*_{C_1}$ and $\tau^*_{C_2}$, computed from the result of `FAST` (see Th. 5.2) are indeed correct.

Assume τ^*_C is, according to `FAST`, equivalent to the language $\varrho_1^* \dots \varrho_n^*$ with $\varrho_i \in T^*$. We encode this language with the following Presburger predicate (similar to the one presented in [4]), which uses the formulas $H(\sigma^{k_i})$ and $\Delta(\sigma^{k_i})$ defined later:

$$\tau^*_C(\boldsymbol{p^1}, \boldsymbol{p^{n+1}}) \triangleq \begin{array}{l} \exists k_1 \dots k_n, \boldsymbol{p^2} \dots \boldsymbol{p^{n-1}} \,. \\ \quad \bigwedge_{i \in 1..n} \left((\boldsymbol{p^i} \geqslant H(\sigma^{k_i})) \wedge \Delta(\sigma^{k_i})(\boldsymbol{p^i}, \boldsymbol{p^{i+1}}) \right) \end{array} \tag{11}$$

This definition introduces acceleration variables k_i, encoding the number of times we fire the sequence ϱ_i. The hurdle and delta of the sequence of transitions ϱ_i^k, which depends on k, are written $H(\sigma^{k_i})$ and $\Delta(\sigma^{k_i})$, respectively. Their formulas are given in equations (14) and (15) below. Let us explain how we obtain them.

First, we define the notion of hurdle $H(\varrho)$ and delta $\Delta(\varrho)$ of an arbitrary sequence ϱ, such that $m \overset{\varrho}{\Rightarrow} m'$ holds if and only if (1) $m \geqslant H(\varrho)$ (the sequence ϱ is fireable), and (2) $m' = m + \Delta(\varrho)$. This is an extension of the hurdle and delta of a single transition t, already used in formulas (3) and (4). The definition of

H and Δ is inductive:

$$H(\epsilon) = \mathbf{0},\ H(t) = \mathrm{Pre}(t) \ \text{ and } \ H(\varrho_1.\varrho_2) = \max\left(H(\varrho_1), H(\varrho_2) - \Delta(\varrho_1)\right) \quad (12)$$

$$\Delta(\epsilon) = \mathbf{0},\ \Delta(t) = \mathrm{Post}(t) - \mathrm{Pre}(t) \ \text{ and } \ \Delta(\varrho_1.\varrho_2) = \Delta(\varrho_1) + \Delta(\varrho_2) \quad (13)$$

where max is the component-wise max operator. The careful reader will check by herself that the definitions of $H(\varrho_1.\varrho_2)$ and $\Delta(\varrho_1.\varrho_2)$ do not depend on the way the sequence $\varrho_1.\varrho_2$ is split.

From these, we are able to characterize a necessary and sufficient condition for firing the sequence ϱ^k, meaning firing the same sequence k times. Given $\Delta(\varrho)$, a place p with a negative displacement (say $-d$) means that d tokens are consumed each time we fire ϱ. Hence, we should budget d tokens in p for each new iteration, and this suffices to enable the $k-1$ more iterations following the first transition ϱ. Therefore, we have $m \xrightarrow{\varrho^k} m'$ if and only if (1) $m \models m \geqslant \mathbb{1}_{>0}(k) \times (H(\varrho) + (k-1) \times \max(\mathbf{0}, -\Delta(\varrho)))$, with $\mathbb{1}_{>0}(k) = 1$ if and only if $k > 0$, and 0 otherwise, and (2) $m' = m + k \times \Delta(\varrho)$. Concerning the token displacement of this sequence ϱ^k, it is k times the one of the non-accelerated sequence ϱ. Equivalently, if we denote by m^+ the "positive" part of a mapping m, such that $m^+(p) = 0$ when $m(p) \leqslant 0$ and $m^+(p) = m(p)$ when $m(p) > 0$, we get:

$$H(\varrho^k) = \mathbb{1}_{>0}(k) \times (H(\varrho) + (k-1) \times (-\Delta(\varrho))^+) \quad (14)$$

$$\Delta(\varrho^k) = k \times \Delta(\varrho) \quad (15)$$

Finally, given a parametric rule $(N_1, C_1) >_E (N_2, C_2)$ we can now check that the reachability expression $\tau^*_{C_1}$ provided by `FAST`, and encoded as explained above, corresponds to the solutions of $\exists \boldsymbol{p_2}\ .\ \tilde{E}(\boldsymbol{p_1}, \boldsymbol{p_2})$ using the following additional SMT query:

$$\forall \boldsymbol{p_1}, \boldsymbol{p'_1}\ .\ C_1(\boldsymbol{p_1}) \implies (\exists \boldsymbol{p_2}\ .\ \tilde{E}(\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge \tilde{E}(\boldsymbol{p'_1}, \boldsymbol{p_2}) \iff \tau^*_{C_1}(\boldsymbol{p_1}, \boldsymbol{p'_1})) \quad (16)$$

(and similarly for $\tau^*_{C_2}$).

Once the equivalence (16) above has been validated by a solver, it is in practice way more efficient to use the formula $(\exists \boldsymbol{p_2}\ .\ \tilde{E}(\boldsymbol{p_1}, \boldsymbol{p_2}) \wedge \tilde{E}(\boldsymbol{p'_1}, \boldsymbol{p_2}))$ inside the core requirements, rather than the formula $\tau^*_{C_1}(\boldsymbol{p_1}, \boldsymbol{p'_1})$ given by `FAST`, since the latter introduces many new acceleration variables.

6 Generalizing Equivalence Rules

Before looking at our implementation, we discuss some results related with the *genericity* and *generalisability* of our abstraction rules. We consider several "dimensions" in which a rule can be generalized. A first dimension is related with the parametricity of the initial marking, which is taken into account by our use of a parametric equivalence, $\cong$ instead of $\equiv$, see Th. 3.1. Next, we show that we can infer an infinite number of equivalences from a single abstraction rule using compositionality, transitivity, and structural modifications involving labels. Therefore, each abstraction law can be interpreted as a schema for several equivalence rules.

Definition 6.1 (Transition Operations)**.** *Given a Petri net* $N = (P, T, \mathrm{Pre}, \mathrm{Post})$ *and its labeling function* $l : T \to \Sigma \cup \{\tau\}$*, we define two operations:* T^-*, for removing, and* T^+*, for duplicating transitions. Let* a *and* b *be labels in* Σ*.*

- $T^-(a)$ *is a net* $(P, T', \mathrm{Pre}', \mathrm{Post}')$*, where* $T' \triangleq T \setminus l^{-1}(a)$*, and* Pre' *(resp.* Post'*) is the projection of* Pre *(resp.* Post*) to the domain* T'*.*
- $T^+(a, b)$ *is a net* $(P, T', \mathrm{Pre}', \mathrm{Post}')$*, where* T' *is a subset of* $T \times \{0, 1\}$ *defined by* $T' \triangleq T \times \{0\} \cup l^{-1}(a) \times \{1\}$*. Additionally, we define* $\mathrm{Pre}'(t, i) \triangleq \mathrm{Pre}(t)$ *and* $\mathrm{Post}'(t, i) \triangleq \mathrm{Post}(t)$ *for all* $t \in T$ *and* $i \in \{0, 1\}$*. Finally, the labeling function* l' *is defined with* $l'(t, 0) \triangleq l(t)$ *and* $l'(t, 1) = b$ *for all* $t \in T$*.*

The operation $T^-(a)$ removes transitions labeled by a, while $T^+(a, b)$ duplicates all transitions labeled by a and labels the copies with b. We illustrated T^+ in the nets of rule (MAGIC), in Fig. 2, where the "dashed" transition c' can be interpreted has the result of applying operation $T^+(c, c')$. Note that these operations only involve labeled transitions. Silent transitions are kept untouched—up-to some injection.

Theorem 6.1 (Preservation by Transition Operations)**.** *Assume we have a parametric* E*-abstraction equivalence* $(N_1, C_1) \approx_E (N_2, C_2)$*,* a *and* b *are labels in* Σ*. Then,*

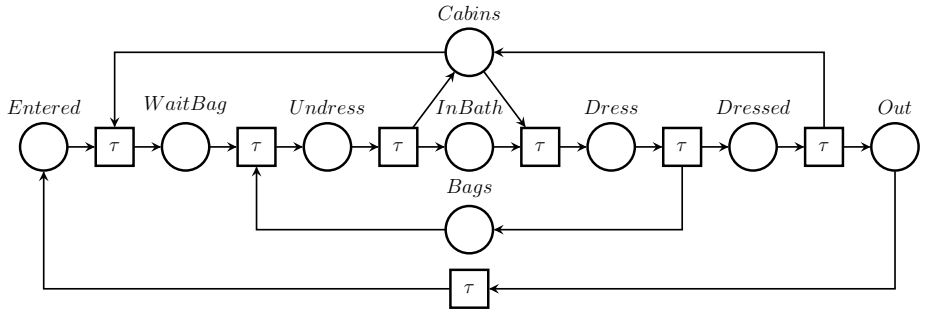
- $T_i^-(a)$ *and* $T_i^+(a, b)$ *satisfy the coherency constraint* C_i*, for* $i = 1, 2$*.*
- $(T_1^-(a), C_1) \approx_E (T_2^-(a), C_2)$*.*
- $(T_1^+(a, b), C_1) \approx_E (T_2^+(a, b), C_2)$*.*

where T_i^-*,* T_i^+ *is (respectively) the operation* T^-*,* T^+ *on* N_i*.*

Finally, we recall a previous result from [1, 2] (Theorem 6.2), which states that equivalence rules can be combined together using synchronous composition, relabeling, and chaining. Note that, in order to avoid inconsistencies that could emerge if we inadvertently reuse the same variable in different reduction equations (variable escaping its scope), we require that conditions can be safely composed: the equivalence statements $(N_1, m_1) \equiv_E (N_2, m_2)$ and $(N_2, m_2) \equiv_{E'} (N_3, m_3)$ are *compatible* if and only if $P_1 \cap P_3 = P_2 \cap P_3$. We also rely on classical operations for relabeling a net, and for synchronous product, $N_1 \parallel N_2$, which are defined in [2] for instance.

Theorem 6.2 (E-equivalence is a Congruence [1, 2])**.** *Assume we have two compatible equivalence statements* $(N_1, m_1) \equiv_E (N_2, m_2)$ *and* $(N_2, m_2) \equiv_{E'} (N_3, m_3)$*, and that* M *is a Petri net such that* $N_1 \parallel M$ *and* $N_2 \parallel M$ *are defined, then*

- $(N_1, m_1) \parallel (M, m) \equiv_E (N_2, m_2) \parallel (M, m)$*.*
- $(N_1, m_1) \equiv_{E,E'} (N_3, m_3)$*.*
- $(N_1[a/b], m_1) \equiv_E (N_2[a/b], m_2)$ *for any* $a \in \Sigma$ *and* $b \in \Sigma \cup \{\tau\}$*.*



N_1

$$C_1 \triangleq Cabins = 10 \wedge Out = 20 \wedge Bags = 15 \wedge$$
$$Entered + WaitingBag + Undress + Dresse + Inbath + Dressed = 0$$

$$E \triangleq \begin{cases} Cabins + Dress + Dressed + Undress + WaitBag = 10 \\ Dress + Dressed + Entered + InBath + Out + Undress + WaitBag = 20 \\ Bags + Dress + InBath + Undress = 15 \end{cases}$$

Figure 8: A Petri net modeling users in a swimming pool, see e.g. [10].

7 Validation and Conclusion

We have implemented our automated procedure in a new tool called `Reductron`. The tool is open-source, under the GPLv3 license, and is freely available on GitHub [23]. The repository contains a subdirectory, `rules`, that provides examples of equivalence rules that can be checked using our approach. Each test contains two Petri nets, one for N_1 (called `initial.net`) and another for N_2 (called `reduced.net`), defined using the syntax of `Tina`. These nets also include declarations for constraints, C_1 and C_2, and for the equation system E. Our list contains examples of laws that are implemented in `Tedd` and `SMPT`, such as rule (CONCAT) depicted in Fig. 1, but also some examples of unsound equivalences rules. For instance, we provide example (FAKE_CONCAT), which corresponds to the example of Fig. 1 with transition d added.

An interesting feature of `Reductron`, when a rule is unsound, is to return which core requirement failed. For instance, with (FAKE_CONCAT), we learn that (N_1, C_1) is not coherent because of d (we cannot reach a coherent marking after firing d using only silent transitions). We can also detect many cases in which there is an error in the specification of either C or E.

We performed some experimentation using `z3` [14] (version 4.8) as our target SMT solver, and `FAST` (version 2.1). All the examples given in our repository can be solved in a few seconds. Although we focus on the automatic verification of abstraction laws, we have also tested our tool on moderate-sized nets, such as the swimming pool example given in Fig. 8. In this context, we use the fact that an equivalence of the form $(N, C) \cong_E (\emptyset, \text{True})$, between N and a net containing an empty set of places, entails that the reachability set of (N, C) must be equal to the solution set of E. In this case, also, results are almost immediate.

These very good results depend largely on the continuous improvements made by SMT solvers. Indeed, we generate very large LIA formulas, with sometimes hundreds of quantified variables, and a moderate amount of quantifier

alternation (formulas of the form $\forall\exists\forall$). For instance, experiments performed with older versions of `z3` (such as 4.4.1, October 2015) exhibit significantly degraded performances. We also rely on the very good performances exhibited by the tool `FAST`, which is essential in the implementation of `Reductron`.

Acknowledgements

We would like to thanks Jérôme Leroux for his support during our experimentation with `FAST`.

References

[1] N. Amat, B. Berthomieu, and S. Dal Zilio. On the combination of polyhedral abstraction and SMT-based model checking for Petri nets. In *Application and Theory of Petri Nets and Concurrency (Petri Nets)*, LNCS. Springer, 2021.

[2] N. Amat, B. Berthomieu, and S. Dal Zilio. A polyhedral abstraction for Petri nets and its application to SMT-based model checking. *Fundamenta Informaticae*, 187(2-4), 2022.

[3] N. Amat and S. Dal Zilio. SMPT: A testbed for reachabilty methods in generalized Petri nets. In *Formal Methods (FM)*, LNCS. Springer, 2023.

[4] N. Amat, S. Dal Zilio, and T. Hujsa. Property Directed Reachability for Generalized Petri Nets. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, LNCS. Springer, 2022.

[5] N. Amat, S. Dal Zilio, and D. Le Botlan. Accelerating the computation of dead and concurrent places using reductions. In *Model Checking Software (SPIN)*, LNCS. Springer, 2021.

[6] N. Amat, S. Dal Zilio, and D. Le Botlan. Leveraging polyhedral reductions for solving Petri net reachability problems. *International Journal on Software Tools for Technology Transfer*, Dec. 2022.

[7] S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. Fast: Fast acceleration of symbolic transition systems. In *Computer Aided Verification (CAV)*, LNCS. Springer, 2003.

[8] S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. FAST: acceleration from theory to practice. *International Journal on Software Tools for Technology Transfer*, 10(5), 2008.

[9] C. Barrett, P. Fontaine, and C. Tinelli. The SMT-LIB Standard: Version 2.6. Technical report, Department of Computer Science, The University of Iowa, 2017. Available at http://www.smt-lib.org/.

[10] B. Bérard and L. Fribourg. Reachability analysis of (timed) Petri nets using real arithmetic. In *Concurrency Theory (CONCUR)*, LNCS. Springer, 1999.

[11] G. Berthelot. Transformations and Decompositions of Nets. In *Petri Nets: Central Models and their Properties (ACPN)*, LNCS. Springer, 1987.

[12] B. Berthomieu, D. Le Botlan, and S. Dal Zilio. Petri net reductions for counting markings. In *Model Checking Software (SPIN)*, LNCS. Springer, 2018.

[13] B. Berthomieu, D. Le Botlan, and S. Dal Zilio. Counting Petri net markings from reduction equations. *International Journal on Software Tools for Technology Transfer*, 2019.

[14] L. De Moura and N. Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, LNCS. Springer, 2008.

[15] J. Esparza. Decidability and complexity of Petri net problems–an introduction. In *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets (ACPN)*, LNCS. Springer, 1998.

[16] J. Esparza and M. Nielsen. Decidability issues for Petri nets. *BRICS Report Series*, 1(8), 1994.

[17] M. H. T. Hack. *Decidability questions for Petri Nets.* PhD thesis, Massachusetts Institute of Technology, 1976.

[18] Y. Hirshfeld. Petri nets and the equivalence problem. In *7th Workshop on Computer Science Logic (CSL)*, LNCS. Springer, 1994.

[19] T. Hujsa, B. Berthomieu, S. Dal Zilio, and D. Le Botlan. Checking marking reachability with the state equation in Petri net subclasses. *CoRR*, abs/2006.05600, 2020.

[20] T. Hujsa, B. Berthomieu, S. Dal Zilio, and D. Le Botlan. On the Petri nets with a single shared place and beyond. *CoRR*, abs/2005.04818, 2020.

[21] LAAS-CNRS. SMPT. https://github.com/nicolasAmat/SMPT/, 2020.

[22] LAAS-CNRS. Tina Toolbox. http://projects.laas.fr/tina, 2020.

[23] LAAS-CNRS. Reductron. https://github.com/nicolasAmat/Reductron/, 2023.

[24] J. Leroux. Presburger vector addition systems. In *2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, 2013.

[25] Y. Thierry-Mieg, D. Poitrenaud, A. Hamez, and F. Kordon. Hierarchical Set Decision Diagrams and regular models. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, LNCS. Springer, 2009.