Bounded Petri Net Synthesis from Modal Transition Systems is Undecidable*

Uli Schlachter

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

Abstract

In this paper, the synthesis of bounded Petri nets from deterministic modal transition systems is shown to be undecidable. The proof is built from three components. First, it is shown that the problem of synthesising bounded Petri nets satisfying a given formula of the conjunctive nu-calculus (a suitable fragment of the mu-calculus) is undecidable. Then, an equivalence between deterministic modal transition systems and a language-based formalism called modal specifications is developed. Finally, the claim follows from a known equivalence between the conjunctive nu-calculus and modal specifications.

1998 ACM Subject Classification F.4.1 Mathematical Logic and Formal Languages

Keywords and phrases Petri net synthesis, conjunctive nu-Calculus, modal transition systems

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2016.15

1 Introduction

Modal transition systems are a well-known and useful method for specifying systems [18, 1, 8, 17]. Petri net synthesis, or more precisely, the problem of finding an unlabelled Petri net implementing a given labelled transition system, has also been investigated since many years [3]. Petri net synthesis not only yields implementations which are correct by design, but it also allows to extract concurrency and distributability information from a sequential specification [4, 7, 21]. Since modal transition systems are extensions of labelled transition systems, it has been suggested to extend Petri net synthesis to cover modal transition systems, e.g., in [9]. However, some questions that are settled in the basic net synthesis theory are still open for such an extension. For example, the decidability status of checking whether a bounded Petri net implementing a given modal transition system exists, has been stated as unknown in [3].

In this paper, we give a negative answer to this question, by proving that it is undecidable whether a given deterministic modal transition system can be implemented by a bounded Petri net. This is done in several steps, two reductions and an equivalence. First, a counter machine, also known as Minsky machine [20], is encoded in a specification that is interpreted on a special class of Petri nets. The technique used in this step resembles constructions known from other papers, for instance [15] (but in the context of labelled Petri nets). In this first step, a variation of the model checking problem is shown to be undecidable. In a second step, the class of Petri nets used in the first step is encoded in a second specification, allowing the Petri net synthesis problem to be shown undecidable. In principle, the two steps

* This work was supported by the German Research Foundation (DFG) project ARS (Algorithms for Reengineering and Synthesis), reference number Be 1267/15-1.

would suffice for our proof, because both specifications used in them can be represented as deterministic modal transition systems. However, these representations are complex and hard to understand or reason about. Therefore, in order to simplify the representations, we add a third step using the conjunctive nu-calculus [11, 12, 13], a fragment of the modal mu-calculus [16, 2]. The first two steps are done with the conjunctive nu-calculus and in this third step we show the equivalence of the nu-calculus, modal specifications and deterministic modal transition systems to derive our main result.

Section 2 introduces the bounded execution problem for two-counter machines. In section 3, we define a class of Petri nets $N_{\text{sim}}(b_0, b_1)$ which can simulate two-counter machines whose counters stay below the bounds $(b_0, b_1) \in \mathbb{N}^2$. Section 4 characterises the reachability graphs of $N_{\text{sim}}(b_0, b_1)$ in a specification that is independent of b_0 and b_1 and derives the undecidability of the bounded Petri net synthesis problem for the conjunctive nu-calculus. We then obtain the main result, the undecidability of the bounded Petri net synthesis problem for deterministic modal transition systems, in section 5 by introducing an effective translation between deterministic modal transition systems and modal specifications, which are known to be equivalent to the conjunctive nu-calculus [11, 13].

The nu-calculus, used in all but the last section, does not allow to express, e.g., nondeterminism or unimplementable specifications. It turns out to be well-suited for research into Petri net synthesis from modal transition systems and was (to the author's knowledge) invented in [12] for the investigation of its Petri net synthesis problem. In fact, the pure and unbounded Petri net synthesis problem for the conjunctive nu-calculus was shown to be undecidable in [12]. Unfortunately, the approach described in [12] cannot be lifted to impure or bounded Petri nets because it encodes the effect of transitions on individual places via auxiliary transitions that have to be able to surpass any bounds. The effect of a transition only characterises the transition in pure Petri nets, but not in impure nets. Instead, a new approach for encoding Petri nets in the nu-calculus has been developed for the purpose of our proof. In fact, the results of [12] and the present paper are incomparable: neither implies the other.

The limitation of the expressivity of the nu-calculus tends to make it incomparable to other fragments of the mu-calculus. For example, the mu-calculus fragment considered in [10] has an undecidable model checking problem, while the nu-calculus is weaker and has a decidable model checking problem, likely even in polynomial time, because of its equivalence with deterministic modal transition systems, where model checking is possible in polynomial time [6].

2 Preliminaries

▶ **Definition 1.** A (finite, initial) *labelled transition system (lts)* is a structure $A = (Q, \Sigma, \to, q_0)$ where Q is a finite set of *states*, Σ is an alphabet, $\to \subseteq Q \times \Sigma \times Q$ is the *transition relation* and q_0 is the *initial state*. An *arc* $(q, a, q') \in \to$ is written as $q \xrightarrow{a} q'$. This is extended to words $w \in \Sigma^*$ via $q \xrightarrow{\varepsilon} q$ and $q \xrightarrow{w} q' \xrightarrow{a} q'' \Rightarrow q \xrightarrow{wa} q''$. If some $q' \in Q$ with $q \xrightarrow{w} q'$ exists, we write $q \xrightarrow{w}$. If no such q' exists, write $q \not\xrightarrow{w}$. The language of A is $L(A) = \{w \in \Sigma^* \mid q_0 \xrightarrow{w}\}$. A is *deterministic* if $\forall q, q', q'', a \colon q \xrightarrow{a} q' \wedge q \xrightarrow{a} q'' \Rightarrow q' = q''$. Two lts A_1, A_2 with $A_i = (Q_i, \Sigma, \to_i, q_{0i})$ are *isomorphic* if there exists a bijection $\xi \colon Q_1 \to Q_2$ so that $\xi(q_{01}) = q_{02}$ and for all $q, q' \in Q_1, a \in \Sigma \colon q \xrightarrow{a}_1 q' \iff \xi(q) \xrightarrow{a}_2 \xi(q')$.

▶ **Definition 2.** A (place-transition, initially marked, injectively labelled, arc-weighted) *Petri net* is a tuple $N = (P, \Sigma, F, M_0)$ where P, Σ are disjoint and finite sets of *places*, respectively *transitions*, $F \colon ((P \times \Sigma) \cup (\Sigma \times P)) \to \mathbb{N}$ is the *flow relation* and M_0 is the

initial marking where a *marking* M is a mapping $M\colon P \to \mathbb{N}$. We call $M(p)$ the number of *tokens* on place p in M. A Petri net is *pure* if it satisfies $\forall p \in P, t \in \Sigma\colon F(p,t) = 0 \vee F(t,p) = 0$, otherwise it is *impure*. The *effect* $E(t) \in \mathbb{Z}^p$ of a transition $t \in \Sigma$ is defined by $E(t)(p) = F(t,p) - F(p,t)$. A transition $t \in \Sigma$ *has no effect* if $E(t)(p) = 0$ for all p. A transition $t \in \Sigma$ is *enabled* in marking M, written $M[t\rangle$, if $\forall p \in P\colon M(p) \geq F(p,t)$. In this situation t can *fire* and leads to marking M', written $M[t\rangle M'$, if $M' = M + E(t)$. This is extended to sequences via $M[\varepsilon\rangle M$, and if $M[u\rangle M'[v\rangle M''$ then $M[uv\rangle M''$. The set of all markings reachable from M_0 is $\mathcal{E}(N) = \{M \mid \exists w \in \Sigma^*\colon M_0[w\rangle M\}$. The *reachability graph* of M is the lts $\mathrm{RG}(N) = (\mathcal{E}(N), \Sigma, \to, M_0)$ where $\to = \{(M,t,M') \mid M[t\rangle M'\}$. We call N *k-bounded* if $\forall M \in \mathcal{E}(N), p \in P\colon M(p) \leq k$ and *bounded* if such a k exists. The *language* of a Petri net is $L(N) = L(\mathrm{RG}(N))$.

The conjunctive nu-calculus is a syntactic fragment of the modal mu-calculus [16, 2]. For example the disjunction, the negation and the least fixed point $\mu X.\beta$ from mu-calculus are missing. Usually, the semantics of the mu-calculus is defined as sets of states. The language-based version used here simplifies some of the proofs in the rest of the paper. A deterministic lts satisfies a formula $\beta \in L_\nu$ in the language-based semantics if and only if it does in state-based semantics [12]. The nu-calculus has operations similar to the modal process logic [19] except that the nu-calculus cannot express non-determinism.

▶ **Definition 3** ([11, 12, 13])**.** Given a set of variables $\mathsf{Var} = \{X_1, X_2, \dots\}$ and an alphabet Σ, the set of all formulas of the *conjunctive nu-calculus* is called L_ν and is defined recursively as the least set containing $\mathtt{true}, X, \to^a, \not\to^a$ and if $\beta_1, \beta_2 \in L_\nu$, then $[a]\beta_1, \beta_1 \wedge \beta_2, \nu X.\beta_1 \in L_\nu$, where $a \in \Sigma$ and $X \in \mathsf{Var}$. A variable $X \in \mathsf{Var}$ is *free in* $\beta \in L_\nu$ if it is not under the scope of any νX, which is the *greatest fixed point operator*. The interpretation $[\![\beta]\!]^{\mathsf{val}}_L \subseteq L$ of a formula $\beta \in L_\nu$ is based on a prefix-closed language $L \subseteq \Sigma^*$ and a valuation $\mathsf{val}\colon \mathsf{Var} \to 2^L$ which assigns subsets of L to variables. $[\![\beta]\!]^{\mathsf{val}}_L$ is defined inductively over the structure of β:

$$
\begin{array}{lcllcl}
[\![\mathtt{true}]\!]^{\mathsf{val}}_L & = & L & [\![X]\!]^{\mathsf{val}}_L & = & \mathsf{val}(X) \\
[\![\to^a]\!]^{\mathsf{val}}_L & = & \{w \in L \mid wa \in L\} & [\![\not\to^a]\!]^{\mathsf{val}}_L & = & \{w \in L \mid wa \notin L\} \\
[\![[a]\beta]\!]^{\mathsf{val}}_L & = & \{w \in L \mid wa \in [\![\beta]\!]^{\mathsf{val}}_L \vee wa \notin L\} & [\![\beta_1 \wedge \beta_2]\!]^{\mathsf{val}}_L & = & [\![\beta_1]\!]^{\mathsf{val}}_L \cap [\![\beta_2]\!]^{\mathsf{val}}_L \\
[\![\nu X.\beta]\!]^{\mathsf{val}}_L & = & \bigcup\{V \subseteq L \mid [\![\beta]\!]^{\mathsf{val}(V/X)}_L \supseteq V\} & & &
\end{array}
$$

The valuation $\mathsf{val}(V/X)$ is defined by $\mathsf{val}(V/X)(X_1) = V$ for $X = X_1$ and $\mathsf{val}(V/X)(X_1) = \mathsf{val}(X_1)$ otherwise. We write $\langle a\rangle\beta$ for the formula $\to^a \wedge [a]\beta$ and by definition this means $[\![\langle a\rangle\beta]\!]^{\mathsf{val}}_L = \{w \in L \mid wa \in [\![\beta]\!]^{\mathsf{val}}_L\}$. For a word $w = a_1 \dots a_n \in \Sigma^+$ we define both $\langle w\rangle\beta = \langle a_1\rangle \dots \langle a_n\rangle\beta$ and $[w]\beta = [a_1] \dots [a_n]\beta$. Via this we can define $[V]\beta = [v_1]\beta \wedge \dots \wedge [v_n]\beta$ for a *finite* set $V \subseteq \Sigma^+$. Because the semantics of a formula β without any free variables does not depend on the valuation val, we simply write $[\![\beta]\!]_L$. A language L satisfies such a formula, written $L \models \beta$, if and only if $\varepsilon \in [\![\beta]\!]_L$ (understand this as the state reached via ε, i.e. the initial state, satisfying the formula). For a Petri net N we define $N \models \beta$ as $L(N) \models \beta$.

The operator $[a]\beta$ can be interpreted as a kind of disjunction as its meaning is $\langle a\rangle\beta \vee \not\to^a$, so it allows a continuation of a word, but does not require it. This is the analogue of a may arc in modal transition systems [18].

▶ **Example 4.** We consider the formula $[a]\to^a$ for the languages $L_1 = \{\varepsilon\}$, $L_2 = \{\varepsilon, a\}$ and $L_3 = \{\varepsilon, a, aa\}$. The interpretation is $[\![[a]\to^a]\!]_{L_i} = \{w \in L_i \mid wa \in [\![\to^a]\!]_{L_i} \vee wa \notin L_i\}$. First, we evaluate $[\![\to^a]\!]_{L_1} = \emptyset$, $[\![\to^a]\!]_{L_2} = \{\varepsilon\}$, and $[\![\to^a]\!]_{L_3} = \{\varepsilon, a\}$. Then, we derive $[\![[a]\to^a]\!]_{L_1} = \emptyset \cup \{\varepsilon\} = \{\varepsilon\}$, $[\![[a]\to^a]\!]_{L_2} = \emptyset \cup \{a\} = \{a\}$, and $[\![[a]\to^a]\!]_{L_3} = \{\varepsilon\} \cup \{aa\} = \{\varepsilon, aa\}$. The results are $L_1 \models [a]\to^a$, $L_2 \not\models [a]\to^a$ and $L_3 \models [a]\to^a$. An intuitive understanding of

$[a]{\rightarrow^a}$ is: If $a \in L$, then $aa \in L$. However, this formula only checks the beginning of words and is not influenced by a elsewhere, since, for example, $\{\varepsilon, b, ba\} \models [a]{\rightarrow^a}$ for the same reasons as $L_1 \models [a]{\rightarrow^a}$.

For an example exhibiting a fixed point operator, consider $\nu X_1.{\rightarrow^a} \wedge [a]X_1$. We begin evaluating the subformula ${\rightarrow^a} \wedge [a]X_1$. The first part requires $a \in L$. The second part states that if $a \in L$ (which is true by the first part), then $a \in \mathsf{val}(X_1)$ where X_1 is bound by the fixed point operator. For an intuitive understanding, we can substitute ${\rightarrow^a} \wedge [a]X_1$ for X_1 and get the formula ${\rightarrow^a} \wedge [a]({\rightarrow^a} \wedge [a]X_1)$. Now we see that also $aa \in L$ is required. Substituting X_1 many times results in the 'infinite formula' ${\rightarrow^a} \wedge [a]({\rightarrow^a} \wedge [a]({\rightarrow^a} \wedge [a](\dots)))$. An example for a language which satisfies this formula is produced by the regular expression a^*. Our formula is equivalent to $\nu X_1.\langle a\rangle X_1$ by the definition of $\langle a\rangle\beta$.

As another example, consider $\beta = \nu X_1.{\rightarrow^b} \wedge \langle a\rangle X_1$ and the regular language $L = L(a^*(b+\varepsilon))$. In this setting, $L \models \beta$. This language is prefix-closed, as required for $[\![\beta]\!]_L$ to be defined.

▶ **Definition 5.** A *two-counter machine* [20] is a tuple $\mathcal{C} = (\ell, \gamma)$ where $\ell \in \mathbb{N}$ is the number of states and $\gamma\colon \{1, \dots, \ell\} \to \Gamma_\ell$ maps states to instructions. Every instruction $\gamma(k) \in \Gamma_\ell$ has one of three possible forms; $\gamma(k) = \texttt{INC}_i\ k'$, $\gamma(k) = \texttt{DEC}_i\ k'\ \texttt{ELSE}\ k''$ or $\gamma(k) = \texttt{HALT}$ where $i \in \{0, 1\}$ and $k', k'' \in \{1, \dots, \ell\}$. The first instruction increments counter i and switches to state k'. The second instruction decrements counter i if possible and then switches to state k'. Otherwise, it switches to state k''. A *configuration* is a tuple $c = (k, (j_0, j_1))$ where $k \in \{1, \dots, \ell\}$ is the current state and $j_0, j_1 \in \mathbb{N}$ are the values of the counters. An *execution of* $\mathcal{C}$ is a sequence of configurations which begins with $(1, (0, 0))$ and a configuration $c = (k, (j_0, j_1))$ is followed by $c' = (k', (j_0', j_1'))$ if $\gamma(k) \neq \texttt{HALT}$ and:

- If $\gamma(k) = \texttt{INC}_i\ k''$, then $k' = k''$, $j_i' = j_i + 1$ and $j_{1-i}' = j_{1-i}$.
- If $\gamma(k) = \texttt{DEC}_i\ k''\ \texttt{ELSE}\ k'''$, then either $j_i = 0$ and $c' = (k''', (j_0, j_1))$ or $j_i \neq 0$, $k' = k''$, $j_i' = j_i - 1$ and $j_{1-i}' = j_{1-i}$.

The execution of $\mathcal{C}$ is unique, hence two-counter machines are deterministic. We say that $\mathcal{C}$ *halts* if a configuration with an instruction $\texttt{HALT}$ is reached. An execution is called *bounded by* (b_0, b_1) if all of its configurations $(k, (j_0, j_1))$ satisfy $j_0 \leq b_0$ and $j_1 \leq b_1$. It is called *bounded* if such (b_0, b_1) exist.

The *bounded execution problem for* $\mathcal{C}$ is to decide if the execution of the two-counter machine $\mathcal{C}$ is bounded. This problem is undecidable, because two-counter machines have the same computational power as Turing machines and solving this problem would allow to decide the halting problem.

3 Simulating Two-Counter Machines with Petri nets

This section introduces a family of bounded Petri nets $N_{\text{sim}}(b_0, b_1)$ with transitions $\Sigma = \{\oplus, \ominus, \odot, \boxplus, \boxminus, \boxdot\}$ and parameters $b_0, b_1 \in \mathbb{N}$. They are defined such that a two-counter machine $\mathcal{C}$ can be simulated on the reachability graph of such a net via a formula $\Phi_\mathcal{C}$ of the conjunctive nu-calculus if and only if its execution is bounded (lemma 10).

A prototypical member of the class $N_{\text{sim}}(b_0, b_1)$ is depicted in Figure 1. The Petri nets from this class can simulate two bounded counters. The values of the counters are the number of tokens on p_0, respectively p_1. Each counter has an initial value of zero, a capacity of b_0, resp. b_1 (this is because the complement places $\overline{p_i}$ have b_i tokens initially), and can be incremented, decremented and tested for zero via transitions $\oplus$, $\ominus$ and $\odot$ (resp. $\boxplus$, $\boxminus$ and $\boxdot$). Every reachable marking M satisfies, by the structure of the net, $M(p_0) + M(\overline{p_0}) = b_0$ and $M(p_1) + M(\overline{p_1}) = b_1$. As an example of the behaviour of these Petri nets, the reachability graph $\mathrm{RG}(N_{\text{sim}}(2, 3))$ is shown in Figure 1.

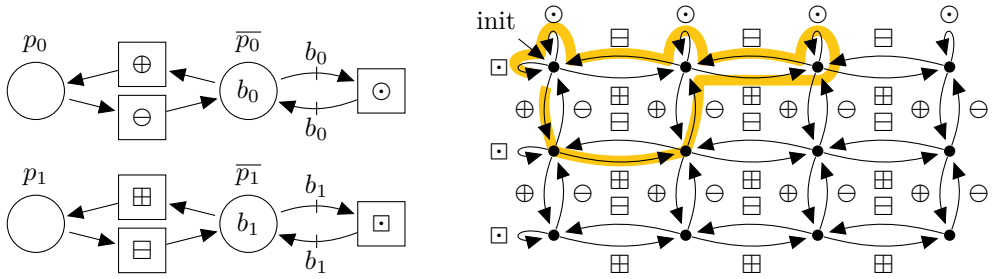

Figure 1 The family of nets $N_{\text{sim}}(b_0, b_1)$ with $b_0, b_1 \in \mathbb{N}$ is depicted on the left. On the right, the reachability graph of $N_{\text{sim}}(2, 3)$ is shown and the firing sequence $\oplus\boxplus\ominus\boxplus\odot\boxminus\odot\boxdot$ is highlighted.

▶ **Lemma 6.** *For $t \in \Sigma = \{\oplus, \ominus, \odot, \boxplus, \boxminus, \boxdot\}$ and $M \in \mathcal{E}(N_{sim}(b_0, b_1))$: $M[t\rangle M'$ iff:*

- *For $t = \oplus$ we have $M'(p_0) = M(p_0) + 1 \leq b_0$ and $M'(p_1) = M(p_1)$.*
- *For $t = \ominus$ we have $M'(p_0) = M(p_0) - 1 \geq 0$ and $M'(p_1) = M(p_1)$.*
- *For $t = \odot$ we have $M' = M$ and $M(p_0) = 0$.*
- *Analogously for $t \in \{\boxplus, \boxminus, \boxdot\}$ (the second counter).*

Proof. This lemma follows from the structure and behaviour of each net $N_{\text{sim}}(b_0, b_1)$. ◀

We define a formula $\Phi_{\mathcal{C}}$ describing a two-counter machine $\mathcal{C}$. By exploiting the structure of $N_{\text{sim}}(b_0, b_1)$, such a formula is satisfied on $N_{\text{sim}}(b_0, b_1)$ iff the execution of $\mathcal{C}$ is bounded.

▶ **Definition 7.** Given a two-counter machine $\mathcal{C} = (\ell, \gamma)$, the formula Φ_k for a state $k \in \{1, \ldots, \ell\}$ in the variables X_1 to X_ℓ is defined by:

$$\Phi_k = \begin{cases} \langle\oplus\rangle X_{k'} & \text{if } \gamma(k) = \texttt{INC}_0\ k' \\ \langle\boxplus\rangle X_{k'} & \text{if } \gamma(k) = \texttt{INC}_1\ k' \\ [\ominus]X_{k'} \wedge [\odot]X_{k''} & \text{if } \gamma(k) = \texttt{DEC}_0\ k'\ \texttt{ELSE}\ k'' \\ [\boxminus]X_{k'} \wedge [\boxdot]X_{k''} & \text{if } \gamma(k) = \texttt{DEC}_1\ k'\ \texttt{ELSE}\ k'' \\ \texttt{true} & \text{if } \gamma(k) = \texttt{HALT} \end{cases}$$

Consider the equation system $X_i = \Phi_i(X_1, \ldots, X_\ell)$ for $i \in \{1, \ldots, \ell\}$ where the $\Phi_k = \Phi_k(X_1, \ldots, X_\ell)$ are as defined above and the syntax $\beta(X)$ is used to clarify free variables. The Gaussian elimination principle (see, e.g., [2]), constructs for each k a formula Ψ_k representing a greatest fixed point solution of X_k in this system. For example, a variable X_k currently defined as $X_k = \beta(X_k)$ is eliminated by replacing $X_k = \beta(X_k)$ with $\Psi_k = \nu X_k.\beta(X_k)$ and substituting X_k in all other formulas with $\nu X_k.\beta(X_k)$. In this way, all variables are eliminated. Since the starting state of $\mathcal{C}$ is state 1, define $\Phi_{\mathcal{C}} := \Psi_1$.

Intuitively we can understand that Φ_k is fulfilled if the behaviour of state k can be simulated. The free variables are used to connect the formulas with each other. For the increment operation, the corresponding event has to be possible and afterwards the following state should be simulated. The decrement operation is more complicated, because there are two possibilities for the following state. To implement this, the structure of $N_{\text{sim}}(b_0, b_1)$ is exploited. In every reachable marking, exactly one of the transitions $\ominus$ and $\odot$ (resp. $\boxminus$ and $\boxdot$) is enabled. Thus, the $[a]$-operator can be used to express this choice. The increment operation uses the $\langle a\rangle$-operator instead, which will make the simulation fail if a counter needs to be incremented beyond the bound b_0 or b_1 of $N_{\text{sim}}(b_0, b_1)$ (because this transition is disabled).

The following examples show the construction and the Gaussian elimination principle:

▶ **Example 8.** We will construct $\Phi_{\mathcal{C}}$ for the two-counter machine $\mathcal{C} = (5, \gamma)$ with:

$\gamma(1) = \texttt{INC}_0\ 2$ $\quad$ $\gamma(3) = \texttt{DEC}_0\ 2\ \texttt{ELSE}\ 4$ $\quad$ $\gamma(5) = \texttt{HALT}$
$\gamma(2) = \texttt{INC}_1\ 3$ $\quad$ $\gamma(4) = \texttt{DEC}_1\ 3\ \texttt{ELSE}\ 5$

This machine begins by incrementing its first counter once (instruction 1). In a loop, it then increments the second counter and decrements the first (instructions 2–3). When the first counter reaches zero, another loop is done decrementing the second counter (instructions 3–4). Its execution and the corresponding operations are:

$$(1,(0,0)) \xrightarrow{\oplus} (2,(1,0)) \xrightarrow{\boxplus} (3,(1,1)) \xrightarrow{\ominus} (2,(0,1)) \xrightarrow{\boxplus} (3,(0,2)) \xrightarrow{\odot} (4,(0,2)) \xrightarrow{\boxminus}$$
$$(3,(0,1)) \xrightarrow{\odot} (4,(0,1)) \xrightarrow{\boxminus} (3,(0,0)) \xrightarrow{\odot} (4,(0,0)) \xrightarrow{\boxdot} (5,(0,0))$$

This execution is bounded by $(1,2)$, but also by $(2,3)$. The formulas for each state are shown in the following system, where a solution for X_1 is needed for $\Phi_{\mathcal{C}}$:

$$X_1 = \langle\oplus\rangle X_2 \quad X_2 = \langle\boxplus\rangle X_3 \quad X_3 = [\ominus]X_2 \wedge [\odot]X_4 \quad X_4 = [\boxminus]X_3 \wedge [\boxdot]X_5 \quad X_5 = \texttt{true}$$

The Gaussian elimination principle now eliminates variables. We begin by substituting the only uses of X_5 and X_4 with their definitions. This produces $X_4 = [\boxminus]X_3 \wedge [\boxdot]\texttt{true}$ and $X_3 = [\ominus]X_2 \wedge [\odot]([\boxminus]X_3 \wedge [\boxdot]\texttt{true})$. The variable X_3 is eliminated by substituting $\nu X_3.[\ominus]X_2 \wedge [\odot]([\boxminus]X_3 \wedge [\boxdot]\texttt{true})$ (note the added νX_3 in front of the value of X_3). This yields $X_2 = \langle\boxplus\rangle(\nu X_3.[\ominus]X_2 \wedge [\odot]([\boxminus]X_3 \wedge [\boxdot]\texttt{true}))$. Continuing by substituting X_2 and eliminating X_1 produces the result:

$$\Phi_{\mathcal{C}} = \Psi_1 = \langle\oplus\rangle(\nu X_2.\langle\boxplus\rangle(\nu X_3.[\ominus]X_2 \wedge [\odot]([\boxminus]X_3 \wedge [\boxdot]\texttt{true})))$$

As we saw above, the execution of $\mathcal{C}$ is bounded by $(2,3)$. The reader may verify that $N_{\text{sim}}(2,3) \models \Phi_{\mathcal{C}}$ due to $\oplus\boxplus\ominus\boxplus\odot\boxminus\odot\boxminus\odot\boxdot$ being in $L(N_{\text{sim}}(2,3))$ (compare Figure 1), which is the word representing the correct execution of $\mathcal{C}$.

▶ **Example 9.** An example for a machine which has a bounded execution, but does not halt, is the machine $\mathcal{C}' = (1, \gamma')$ with $\gamma'(1) = \texttt{DEC}_0\ 1\ \texttt{ELSE}\ 1$. This machine loops in its initial configuration. Its formula is $\Phi_{\mathcal{C}'} = \nu X_1.[\ominus]X_1 \wedge [\odot]X_1$. Here we have $N_{\text{sim}}(0,0) \models \Phi_{\mathcal{C}'}$ and the 'infinite word' representing a correct simulation is $\odot^\omega$. If we modify this machine by setting $\gamma'(1) = \texttt{INC}_0\ 1$, we get $\Phi_{\mathcal{C}'} = \nu X_1.\langle\oplus\rangle X_1$. This machine's unbounded execution is represented by $\oplus^\omega$. No instance of $N_{\text{sim}}(b_0, b_1)$ allows an infinite sequence of increments.

▶ **Lemma 10.** *The execution of a two-counter machine $\mathcal{C}$ is bounded by $(b_0, b_1) \in \mathbb{N} \times \mathbb{N}$ if and only if $N_{sim}(b_0, b_1) \models \Phi_{\mathcal{C}}$.*

Proof sketch. An analogous result was shown in [12]. The main difference is in the definitions of $N_{\text{sim}}(b_0, b_1)$ and Φ_k, so that we can incorporate impure Petri nets.

We can define words $w_i \in \Sigma^*$ that contain the operations done to reach the i-th configurations in $\mathcal{C}$'s execution. Assuming $\mathcal{C}$ is bounded, by induction it follows from lemma 6 that all $w_i \in L := L(N_{\text{sim}}(b_0, b_1))$. These words can be grouped into sets V_k containing for a state k all words w_i where the i-th configuration is in state k. Show that $[\![\Phi_k]\!]_L^{\mathsf{val}} \supseteq V_k$ holds for val defined by $\mathsf{val}(X_i) = V_i$. By standard fixed point theory it follows that these values are contained in the unique greatest fixed point and so $w_1 = \varepsilon \in V_1 \subseteq [\![\Phi_1]\!]_L^{\mathsf{val}} \subseteq [\![\Phi_{\mathcal{C}}]\!]_L$, which was to show.

Reachable markings of $N_{\text{sim}}(b_0, b_1)$ are related to counter values in an obvious way. Assuming $\varepsilon \in [\![\Phi_{\mathcal{C}}]\!]_L$, it can be shown via lemma 6 that the words w_i as defined above reach markings corresponding to the i-th configuration. Thus, $\Phi_{\mathcal{C}}$ follows the operations done by $\mathcal{C}$ and since in $N_{\text{sim}}(b_0, b_1)$ only bounded markings are reachable, the execution of $\mathcal{C}$ is bounded by the same numbers. ◀

Because the bounded execution problem is undecidable, we obtain:

▶ **Corollary 11.** *Given $\mathcal{C}$, it is undecidable if $\exists b_0, b_1 \in \mathbb{N}$ such that $N_{sim}(b_0, b_1) \models \Phi_{\mathcal{C}}$.*

4 Undecidability of Petri Net Synthesis from the Nu-calculus

This section characterises the reachability graph of Petri nets $N_{\text{sim}}(b_0, b_1)$ via a formula $\Phi_{N_{\text{sim}}}$ independent of b_0 and b_1. Our goal is to show that, given a formula $\beta \in L_\nu$, the existence of a bounded Petri net satisfying β is undecidable. For this, we first want to show that if a bounded Petri net satisfies $\Phi_{N_{\text{sim}}} \wedge \Phi_{\mathcal{C}}$, then there are numbers b_0, b_1 so that $N_{\text{sim}}(b_0, b_1) \models \Phi_{\mathcal{C}}$. The undecidability then follows via corollary 11. In the following sections, various parts of $\Phi_{N_{\text{sim}}}$ are introduced. Section 4.3 then shows the result sketched above.

4.1 Auxiliary Formulas

We begin with a formula that signifies that a word $w \in \Sigma^*$ can be fired infinitely often:

$$\text{NoEffect}(w) = \nu X_1.(\langle w \rangle X_1)$$

▶ **Lemma 12.** *$N \models \text{NoEffect}(w)$ for a bounded Petri net N if and only if $M_0[w\rangle M_0$.*

Proof. By definition $[\![\text{NoEffect}(w)]\!]_L = \bigcup\{V \subseteq L \mid \{v \in L \mid vw \in V\} \supseteq V\}$. This means it is the largest subset W of L that satisfies $v \in W \Rightarrow vw \in W$. Thus, the premise $\varepsilon \in [\![\text{NoEffect}(w)]\!]_L$ is equivalent to $\{w\}^* \subseteq [\![\text{NoEffect}(w)]\!]_L$. Since always $[\![\beta]\!]_L \subseteq L$, we derive $\{w\}^* \subseteq L$. In any *bounded* Petri net it holds that $\{w\}^* \subseteq L$ iff $M_0[w\rangle M_0$. ◀

By $M_0[aa'\rangle M_0 \Rightarrow M_0 = M_0 + E(a) + E(a')$, we get that $\text{Inv}(a, a')$ expresses that transitions $a, a' \in \Sigma$ have opposite effects:

$$\text{Inv}(a, a') = \text{NoEffect}(aa')$$

▶ **Corollary 13.** *$N \models \text{Inv}(a, a')$ for a bounded Petri net N if and only if $M_0[aa'\rangle$ and $E(a) = -E(a')$.*

Next, we define a formula that requires some formula β to hold globally. Here, X_1 is a fresh variable which does not appear in β.

$$\text{Global}(\beta) = \nu X_1.([\Sigma]X_1 \wedge \beta)$$

▶ **Lemma 14.** *$\varepsilon \in [\![\text{Global}(\beta)]\!]_L^{\mathsf{val}}$ if and only if $L = [\![\beta]\!]_L^{\mathsf{val}}$.*

Proof. We begin with $[\![[\Sigma]X_1 \wedge \beta]\!]_L^{\mathsf{val}} = \{w \in [\![\beta]\!]_L^{\mathsf{val}} \mid \forall a \in \Sigma\colon wa \in \mathsf{val}(X_1) \vee wa \notin L\}$. The fixed point produces $[\![\text{Global}(\beta)]\!]_L^{\mathsf{val}} = \{w \in [\![\beta]\!]_L^{\mathsf{val}} \mid \forall a \in \Sigma\colon wa \in L \Rightarrow wa \in [\![\beta]\!]_L^{\mathsf{val}}\} = \{w \in [\![\beta]\!]_L^{\mathsf{val}} \mid \forall u \in \Sigma^*\colon wu \in L \Rightarrow wu \in [\![\beta]\!]_L^{\mathsf{val}}\}$. The following implications follow: $\varepsilon \in [\![\text{Global}(\beta)]\!]_L^{\mathsf{val}} \Rightarrow L = [\![\beta]\!]_L^{\mathsf{val}} \Rightarrow L = [\![\text{Global}(\beta)]\!]_L^{\mathsf{val}} \Rightarrow \varepsilon \in [\![\text{Global}(\beta)]\!]_L^{\mathsf{val}}$. ◀

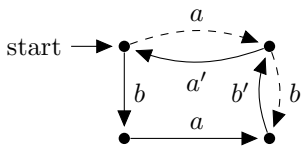

Figure 2 Illustration for $\mathrm{Swp}_h(a, b, a', b')$. Whenever the dashed arcs are present, the solid arcs are required. The interpretation begins in the upper left corner.

Next we define formulas which require two events to be swappable in the language, using events with opposite effects to 'undo' the firing of a transition. Swapping ab into ba is expressed by Swp_h and the opposite direction is added by Swp. This is extended to all combinations of the four events a, b, a' and b' by Swap. Figure 2 illustrates $\mathrm{Swp}_h(a, b, a', b')$.

$$\mathrm{Swp}_h(a, b, a', b') = \mathrm{Global}([a][b]\langle b'\rangle\langle a'\rangle\langle b\rangle\langle a\rangle \mathtt{true})$$
$$\mathrm{Swp}(a, b, a', b') = \mathrm{Swp}_h(a, b, a', b') \wedge \mathrm{Swp}_h(b, a, b', a')$$
$$\mathrm{Swap}(a, b, a', b') = \mathrm{Swp}(a, b, a', b') \wedge \mathrm{Swp}(a, b', a', b) \wedge \mathrm{Swp}(a', b, a, b') \wedge \mathrm{Swp}(a', b', a, b)$$

▶ **Lemma 15.** *Let N be a Petri net with $E(a) = -E(a')$ and $E(b) = -E(b')$. Then $N \models \mathrm{Swp}(a, b, a', b')$ if and only if $\forall w \in L(N)\colon wab \in L(N) \iff wba \in L(N)$.*

Proof. First, we have $\varepsilon \in [\![\mathrm{Swp}_h(a, b, a', b')]\!]_L$ iff for all $wab \in L$ also $wabb'a'ba \in L$ by the definition of the semantics and by lemma 14. Since a and a' (and b and b') have opposite effects, this can be restated as: For all reachable markings M, we have $M[ab\rangle \Rightarrow M[ba\rangle$. This shows both directions of the lemma. ◀

▶ **Corollary 16.** *For a net N with $E(a) = -E(a')$ and $E(b) = -E(b')$, $N \models \mathrm{Swap}(a, b, a', b')$ if and only if $\forall c \in \{a, a'\}, d \in \{b, b'\}, w \in L(N)\colon wcd \in L(N) \iff wdc \in L(N)$.*

4.2 Characterisation of our Class of Nets via the Nu-Calculus

We will characterise $N_{\mathrm{sim}}(b_0, b_1)$ by a formula $\Phi_{N_{\mathrm{sim}}} = \Phi_{\mathrm{E}} \wedge \Phi_{\mathrm{swap}} \wedge \Phi^1_{\mathrm{dec}} \wedge \Phi^2_{\mathrm{dec}} \wedge \Phi_{\mathrm{dep}} \wedge \Phi_{\mathrm{zero}}$. With Φ_{E}, we will restrict the effects of transitions, Φ_{swap} requires swaps to be possible, and Φ^1_{dec} forbids the decrement of a counter with value zero. The formula Φ^2_{dec} requires a counter decrement to be possible after every increment and the formula Φ_{dep} ensures that counters are independent. The correct behaviour of the zero-test event is required by Φ_{zero}.

$$\Phi_{\mathrm{E}} = \mathrm{Inv}(\oplus, \ominus) \wedge \mathrm{NoEffect}(\odot) \wedge \mathrm{Inv}(\boxplus, \boxminus) \wedge \mathrm{NoEffect}(\boxdot)$$

By lemma 12 and corollary 13, Φ_{E} expresses that the increment and decrement transitions have opposite effects and that the test transitions have no effect on the marking. It can easily be seen that all Petri nets in the class $N_{\mathrm{sim}}(b_0, b_1)$ have these properties.

Another property of $N_{\mathrm{sim}}(b_0, b_1)$ is that it contains two independent subnets. The order in which transitions from different subnets fire is arbitrary. By Φ_{E}, transitions $\odot$ and $\boxdot$ have no effect and do not need to be considered. For the other events, Swap is used. Additionally, in the initial marking, the decrement operations are disabled by ϕ^1_{dec}:

$$\Phi_{\mathrm{swap}} = \mathrm{Swap}(\oplus, \boxplus, \ominus, \boxminus) \qquad \Phi^1_{\mathrm{dec}} = \not\to^{\ominus} \wedge \not\to^{\boxminus}$$

▶ **Lemma 17.** *For a bounded Petri net N over Σ such that $N \models \Phi_E \wedge \Phi_{swap} \wedge \Phi^1_{dec}$, for every reachable marking M there are words $w \in \{\oplus\}^*$ and $v \in \{\boxplus\}^*$ so that $M_0[vw\rangle M$ and $M_0[wv\rangle M$.*

Proof. Since M is a reachable marking, there is a word $u \in \Sigma^*$ so that $M_0[u\rangle M$. By $N \models \Phi_{\mathrm{E}}$ and lemma 12, events $\odot$ and $\boxdot$ have no effect and can be removed from u without problems. By corollary 16 and $N \models \Phi_{\mathrm{swap}}$, we can freely swap remaining events belonging to different counters. Every subword ba of u with $a \in \{\oplus, \ominus\}$ and $b \in \{\boxplus, \boxminus\}$ is now swapped into ab. This results in a word from $\{\oplus, \ominus\}^* \cdot \{\boxplus, \boxminus\}^*$. Again by $N \models \Phi_{\mathrm{E}}$ and by corollary 13, we know that $\oplus$ and $\ominus$ (respectively $\boxplus$ and $\boxminus$) have opposite effects. Thus all subwords $\oplus\ominus$, $\ominus\oplus$, $\boxplus\boxminus$ and $\boxminus\boxplus$ can be removed. A firing sequence $u = vw$ of the required form reaching M remains and induction over corollary 16 swaps this into wv. Note that by Φ^1_{dec}, neither v nor w begin with a decrement operation and thus both only contain increments. ◀

Whenever a counter is incremented, it can be decremented afterwards:

$$\Phi^2_{\mathrm{dec}} = \mathrm{Global}([\oplus]{\rightarrow}^{\ominus}) \wedge \mathrm{Global}([\boxplus]{\rightarrow}^{\boxminus})$$

▶ **Lemma 18.** *If $N \models \Phi^2_{dec}$ then for all $w \in \Sigma^*$ it holds that $w\oplus \in L(N) \Rightarrow w\oplus\ominus \in L(N)$ and $w\boxplus \in L(N) \Rightarrow w\boxplus\boxminus \in L(N)$.*

Proof. We only show the first part. Let $w\oplus \in L(N)$. Since $L(N)$ is prefix closed and by lemma 14, we have $w \in L(N) = [\![[\oplus]{\rightarrow}^{\ominus}]\!]_{L(N)} = \{u \in L(N) \mid u\oplus \notin L(N) \vee u\oplus\ominus \in L(N)\}$. By $w\oplus \in L(N)$, w must satisfy the second part of the disjunction. Thus, $w\oplus\ominus \in L(N)$. ◀

The formulas defined so far allow the simulation of counters. However, it is still possible that the counters are interdependent and can, for example, reach the values $(2, 3)$ and $(3, 2)$, but not $(3, 3)$. Such dependencies are not present in $N_{\mathrm{sim}}(b_0, b_1)$ and we exclude them via the formula Φ_{dep}:

$$\Phi_{\mathrm{dep}} = \mathrm{Global}([\ominus][\boxplus]{\rightarrow}^{\oplus})$$

We use this formula in the next lemma to express that, if both counters can be incremented, then they can also be incremented one after another. Since L_ν does not allow implications, the situation that both counters can be incremented is captured as $[\ominus][\boxplus]$. Φ_{dep} only makes a requirement on the first counter, because the second counter follows via Φ_{swap}.

▶ **Lemma 19.** *If $N \models \Phi_E \wedge \Phi_{swap} \wedge \Phi^2_{dec} \wedge \Phi_{dep}$, then for every $w \in \Sigma^*$, we have $w\oplus \in L(N) \wedge w\boxplus \in L(N) \Rightarrow w\boxplus\oplus \in L$*

Proof. First, we can apply lemma 18 and get $w\oplus \in L(N) \Rightarrow w\oplus\ominus \in L(N)$. By $N \models \Phi_{\mathrm{E}}$, we know that $\oplus\ominus$ has no effect, so w and $w\oplus\ominus$ reach the same marking. Because of $w\boxplus \in L(N)$, this means that we also have $w\oplus\ominus\boxplus \in L(N)$. Thus, after $w\oplus$, the sequence $\ominus\boxplus$ is enabled and Φ_{dep} yields that $\oplus$ is enabled afterwards, so $w\oplus\ominus\boxplus\oplus \in L(N)$. Since $\oplus\ominus$ has no effect, we can remove this part and get $w\boxplus\oplus \in L(N)$. ◀

▶ **Definition 20.** For a Petri net N define the associated numbers $b_0(N), b_1(N) \in \mathbb{N} \cup \{\infty\}$ to be the supremum of possible values so that $M_0[\oplus^{b_0(N)}\rangle$ and $M_0[\boxplus^{b_1(N)}\rangle$.

We can now characterise the state space of a Petri net satisfying the formulas defined so far:

▶ **Lemma 21.** *For a bounded Petri net N over Σ with $N \models \Phi_E \wedge \Phi_{swap} \wedge \Phi^1_{dec} \wedge \Phi^2_{dec} \wedge \Phi_{dep}$, a marking M is reachable if and only if it is reachable via $M_0[\oplus^{j_0}\boxplus^{j_1}\rangle M$ with $j_i \leq b_i(N)$.*

Proof. First we show that all sequences $\oplus^{j_0}\boxplus^{j_1}$ are enabled. By definition we have $\oplus^{j_0} \in L(N)$ and $\boxplus^{j_1} \in L(N)$. Applying lemma 19 and corollary 16 (to swap the firing sequences into the needed form) inductively shows that $\oplus^{j_0}\boxplus^{j_1} \in L(N)$. It remains to show that no

more markings are reachable. According to lemma 17, we only have to consider markings M reachable through words of our form. Thus, it only remains to show that $j_0 \leq b_0(N)$ and $j_1 \leq b_1(N)$. However, corollary 16 can be used to bring either the part $\oplus^{j_0}$ or $\boxplus^{j_1}$ to the beginning of the word. Now, the definitions of $b_0(N)$ and $b_1(N)$ guarantee that these bounds are not exceeded. ◀

$\Phi_{\text{zero}} = \Phi^0_{\text{zero}} \wedge \Phi^1_{\text{zero}}$ requires that the zero test is enabled when a counter has the value zero, no matter which value the other counter has, and is disabled otherwise:

$$\Phi^0_{\text{zero}} = (\nu X_1.[\boxplus]X_1 \wedge \rightarrow^{\odot}) \wedge \text{Global}([\oplus]\not\rightarrow^{\odot})$$
$$\Phi^1_{\text{zero}} = (\nu X_1.[\oplus]X_1 \wedge \rightarrow^{\boxdot}) \wedge \text{Global}([\boxplus]\not\rightarrow^{\boxdot})$$

For a language satisfying this formula, the first part of Φ^0_{zero} requires that for any $\boxplus^i \in L$, also $\boxplus^i\odot \in L$. The second part states that no word may end in $\oplus\odot$ and thus the zero test is not possible after an increment.

▶ **Lemma 22.** *Let N be a bounded Petri net with $N \models \Phi_E \wedge \Phi_{swap} \wedge \Phi^1_{dec} \wedge \Phi_{zero}$ and $j_0, j_1 \in \mathbb{N}$ so that $w = \oplus^{j_0}\boxplus^{j_1} \in L(N)$. Then $w\odot \in L(N) \iff j_0 = 0$ and $w\boxdot \in L(N) \iff j_1 = 0$.*

Proof. If $j_1 = 0$, then the first part of Φ^1_{zero} requires $w\boxdot = \oplus^{j_0}\boxdot \in L(N)$ for any value of j_0. Similarly, if $j_1 > 0$, the second part requires $w\boxdot = \oplus^{j_0}\boxplus^{j_1}\boxdot \notin L(N)$. For the analogous statement for the first counter, we apply lemma 17 to get $\boxplus^{j_1}\oplus^{j_0} \in L(N)$ and make an analogous argument afterwards. ◀

▶ **Lemma 23.** *Let N be a bounded Petri net such that $N \models \Phi_E \wedge \Phi_{zero}$. Then the transition $\oplus$ has a negative effect on some place p and $\boxplus$ has a negative effect on some place q, where $p = q$ is allowed, and both $b_0(N) \neq 0 \neq b_1(N)$ and $b_0(N) \neq \infty \neq b_1(N)$ hold.*

Proof. Observe that by Φ^0_{zero}, transition $\odot$ is initially enabled, but disabled after $\oplus$ (which has to be enabled by Φ_{E}). Thus, $\oplus$ consumes tokens required by $\odot$ from a place p.

For the second part, by corollary 13, $\text{Inv}(\oplus, \ominus)$ (part of Φ_{E}) requires $\oplus$ to be enabled in the initial marking. Thus, $b_0(N) > 0$. For $b_0(N) \neq \infty$, we observe that the initial token count $M_0(p)$ on the place p from which $\oplus$ consumes tokens is finite and $\oplus$ becomes disabled eventually ($b_0(N) < \infty$). A similar argument can be made for q and $0 < b_1 < \infty$. ◀

4.3 Undecidability of Petri Net Synthesis from the Nu-calculus

With the definitions and results of the previous sections, we now prove that $\Phi_{N_{\text{sim}}} := \Phi_{\text{E}} \wedge \Phi_{\text{swap}} \wedge \Phi^1_{\text{dec}} \wedge \Phi_{\text{dep}} \wedge \Phi^2_{\text{dec}} \wedge \Phi_{\text{zero}}$ characterises the reachability graph of $N_{\text{sim}}(b_0, b_1)$:

▶ **Lemma 24.** *Let N be a bounded Petri net over the alphabet Σ with $N \models \Phi_{N_{sim}}$, then $\text{RG}(N)$ and $\text{RG}(N_{sim}(b_0(N), b_1(N)))$ are isomorphic.*

Proof. Lemma 23 shows that $b_0(N) \neq \infty \neq b_1(N)$. Thus, $N_{\text{sim}}(b_0(N), b_1(N))$ is well-defined. By lemma 6 and lemma 21, all reachable markings in either net are reached via firing sequences $\oplus^{j_0}\boxplus^{j_1}$ with $j_0 \leq b_0(N)$ and $j_1 \leq b_1(N)$. By lemma 23 and by the structure of $N_{\text{sim}}(b_0(N), b_1(N))$, different such firing sequences reach different markings. Thus, we can uniquely identify markings of either net with firing sequences $\oplus^{j_0}\boxplus^{j_1}$ and use this relation as a bijection between markings.

It remains to show that this bijection preserves the firing of transitions. By the structure of $N_{\text{sim}}(b_0(N), b_1(N))$ and by lemma 22, the zero test is enabled in a marking M of either net exactly if the corresponding counter was not increased to reach M. By Φ_{E} and the structure

of $N_{\text{sim}}(b_0(N), b_1(N))$, the zero test has no effect and reaches the marking M again. By a similar argument, this time including lemma 18, a decrement is possible if the corresponding counter has a non-zero value and reaches the marking where the corresponding counter was incremented one time less often. Finally, by the structure of $N_{\text{sim}}(b_0(N), b_1(N))$ and lemma 21, a counter can be incremented as long as it is below its maximal value.

Altogether this shows that we have an isomorphism between $\text{RG}(N_{\text{sim}}(b_0(N), b_1(N)))$ and $\text{RG}(N)$ that preserves the firing of transitions. ◀

▶ **Theorem 25.** *It is undecidable whether there exists a bounded Petri net N with transitions $\Sigma = \{\oplus, \ominus, \odot, \boxplus, \boxminus, \boxdot\}$ so that $N \models \beta$ for a given formula $\beta \in L_\nu$ without free variables.*

Proof. We use a reduction from the problem of finding $b_0, b_1 \in \mathbb{N}$ so that $N_{\text{sim}}(b_0, b_1) \models \Phi_{\mathcal{C}}$, which is undecidable according to corollary 11. Define $\beta_{\mathcal{C}} = \Phi_{N_{\text{sim}}} \wedge \Phi_{\mathcal{C}}$. We show that $\exists N \colon N \models \beta_{\mathcal{C}} \iff \exists b_0, b_1 \in \mathbb{N} \colon N_{\text{sim}}(b_0, b_1) \models \Phi_{\mathcal{C}}$.

Assume a bounded Petri net N with $N \models \beta_{\mathcal{C}}$. By lemma 24, $\text{RG}(N)$ and $A := \text{RG}(N_{\text{sim}}(b_0(N), b_1(N)))$ are isomorphic. From this, we derive $L(\text{RG}(N)) = L(A)$ and because the semantics of L_ν are defined on languages, we arrive at $N_{\text{sim}}(b_0(N), b_1(N)) \models \beta_{\mathcal{C}}$. By definition of the conjunction, this gives us $N_{\text{sim}}(b_0(N), b_1(N)) \models \Phi_{\mathcal{C}}$. We can easily compute $b_0(N), b_1(N) \in \mathbb{N}$ via their definition.

For the other direction, assume that there are numbers $b_0, b_1 \in \mathbb{N}$ so that $N_{\text{sim}}(b_0, b_1) \models \Phi_{\mathcal{C}}$. Without loss of generality we can assume that $b_0 \neq 0 \neq b_1$, because if the execution of $\mathcal{C}$ is bounded by (b_0, b_1), it is also bounded by $(b_0 + 1, b_1 + 1)$ (apply lemma 10). It remains to show that $\forall x, y > 0 \colon N_{\text{sim}}(x, y) \models \Phi_{N_{\text{sim}}}$, because we can combine this to $\Phi_{N_{\text{sim}}} \wedge \Phi_{\mathcal{C}} = \beta_{\mathcal{C}}$ and arrive at $N_{\text{sim}}(b_0, b_1) \models \beta_{\mathcal{C}}$. To satisfy $\Phi_{N_{\text{sim}}}$, all its parts have to hold. The formula Φ_{E} only makes requirements about the effect of transitions by requiring some infinite sequences to be initially enabled (see lemma 12 and corollary 13). These sequences are possible in $N_{\text{sim}}(b_0, b_1)$, but only if $b_0 > 0$ and $b_1 > 0$, which we can assume. The rest follows directly from the semantics of $\Phi_{N_{\text{sim}}}$ and the structure of the Petri net together with corollary 16. ◀

5 Undecidability of Synthesis from Modal Transition Systems

We will show our main result in theorem 30 via an equivalence between the conjunctive nu-calculus and deterministic modal transition systems. This equivalence is established via another formalism, modal specifications, which is known to be equivalent to the conjunctive nu-calculus. Similar equivalences are known for more expressive models [5], but depend on a $\vee$-operator, which L_ν does not have, for transforming a formula to an automaton. A different approach follows.

Remember that a finite automaton $A = (Q, \Sigma, \to, F, q_0)$ is a lts equipped with a set $F \subseteq Q$ of final states. The language of A is $L_f(A) = \{w \in \Sigma^* \mid \exists q \in F \colon q_0 \xrightarrow{w} q\}$. Every regular language is the language of a unique minimal deterministic finite automaton A_L [14].

▶ **Definition 26** ([18])**.** A *modal transition system* is a tuple $\mathcal{M} = (S, \Sigma, \to_\Box, \to_\Diamond, s_0)$ where S is a finite set of *states*, Σ an alphabet, s_0 the *initial state* and $\to_\Box, \to_\Diamond \subseteq S \times \Sigma \times S$ with $\to_\Box \subseteq \to_\Diamond$ are its *must* and *may arcs*, respectively. The *maximal implementation of* $\mathcal{M}$ is the lts $\overline{\mathcal{M}} = (S, \Sigma, \to_\Diamond, s_0)$. $\mathcal{M}$ is *deterministic* if $\overline{\mathcal{M}}$ is.

A lts $A = (Q, \Sigma, \to, q_0)$ is an *implementation of* $\mathcal{M}$, written $A \models \mathcal{M}$, if a relation $R \subseteq Q \times S$ exists so that $(q_0, s_0) \in R$ and for all $(q, s) \in R$ and all $a \in \Sigma$ the following holds:

1. If $s \xrightarrow{a}_\Box s'$, then $\exists q' \in Q$ with $q \xrightarrow{a} q'$ and $(q', s') \in R$.
2. If $q \xrightarrow{a} q'$, then $\exists s' \in S$ with $s \xrightarrow{a}_\Diamond s'$ and $(q', s') \in R$.

Also, $L \models \mathcal{M}$ for a prefix-closed language L, iff there is a det. lts $A \models \mathcal{M}$ with $L = L(A)$.

▶ **Lemma 27.** *For any modal transition system $\mathcal{M} = (S, \Sigma, \to_\Box, \to_\Diamond, s_0)$ and any prefix-closed language L, both $\overline{\mathcal{M}} \models \mathcal{M}$ and $L \models \mathcal{M} \Rightarrow L \subseteq L(\overline{\mathcal{M}})$ hold. If $A \models \mathcal{M}$ for a lts $A = (Q, \Sigma, \to, q_0)$ and $\mathcal{M}$ and A are both deterministic, then $R = \{(q, s) \mid \exists w \in \Sigma^* \colon q_0 \xrightarrow{w} q \wedge s_0 \xrightarrow{w}_\Diamond s\}$ is a relation witnessing $A \models \mathcal{M}$.*

▶ **Definition 28** ([11])**.** A *modal specification* is a tuple $S_m = (\{C_a\}_{a\in\Sigma}, I)$ describing a class of languages where $C_a \subseteq \Sigma^*$ contains all words that must be extendable by an additional a, while words from $I \subseteq \Sigma^*$ are forbidden to occur at all. All C_a and I are regular languages. The *completion operator* associated to S_m is the function $C_{S_m} \colon 2^{\Sigma^*} \to 2^{\Sigma^*}$ defined by $C_{S_m}(L) = \bigcup_{a\in\Sigma}(L \cap C_a) \cdot \{a\}$. A prefix-closed language satisfies S_m, written $L \models S_m$, if and only if $C_{S_m}(L) \subseteq L$ and $L \cap I = \emptyset$. A modal specification is *incoherent* if $I \cap C_{S_m}(\Sigma^*) \neq \emptyset$.

▶ **Lemma 29** ([13])**.** *For every modal specification S_m there is an equivalent coherent modal specification S'_m.*

Proof sketch. For S'_m, modify the C_a according to $C'_a = C_a \setminus \{w \in \Sigma^* \mid wa \in I\}$. ◀

We can now prove our main result:

▶ **Theorem 30.** *Given a deterministic modal transition system $\mathcal{M}$, it is undecidable if a bounded Petri net N with $L(N) \models \mathcal{M}$ exists.*

Proof. It is known that formulas of L_ν without free variables and modal specification are equally expressive [11, 13]. Therefore, invoking theorem 25 shows that it is undecidable if for a given modal specification S_m there is a bounded Petri net N with $L(N) \models S_m$. We prove this theorem by showing that modal specification and deterministic modal transition systems are equally expressive.

Given a deterministic modal transition system $\mathcal{M} = (S, \Sigma, \to_\Box, \to_\Diamond, s_0)$, we construct a modal specification $S_m(\mathcal{M})$ so that for all prefix-closed languages $L \subseteq \Sigma^* \colon L \models \mathcal{M} \iff L \models S_m(\mathcal{M})$ as follows. The set $S(a) := \{s \in S \mid s \xrightarrow{a}_\Box\}$ is the set of all states having an outgoing must arc with label a. Via this, define $A_{\mathcal{M}}^{S(a)}$ as the finite automaton based on the lts $\overline{\mathcal{M}}$ with $S(a)$ as its set of final states. Now $C_a := L_f(A_{\mathcal{M}}^{S(a)})$ is the regular language containing words after which an a is required. Finally, define the modal specification $S_m(\mathcal{M}) := (\{C_a\}_{a\in\Sigma}, I)$ where $I := \Sigma^* \setminus L(\overline{\mathcal{M}})$ is the set of forbidden words.

Analogously, given a coherent modal specification S_m, there is a deterministic modal transition system $\mathcal{M}(S_m)$ so that for all prefix-closed languages $L \subseteq \Sigma^* \colon L \models S_m \iff L \models \mathcal{M}(S_m)$. For a regular language L, let A_L be the deterministic automaton accepting it, let $S_m = (\{C_a\}_{a\in\Sigma}, I)$ be given and let A be the synchronous product of all A_{C_a} and A_I. The synchronous product of two automata is a lts constructed by taking the Cartesian product of the sets of states and defining the transition relation element-wise, i.e. if $q \xrightarrow{a} q'$ and $s \xrightarrow{a} s'$, then $(q, s) \xrightarrow{a} (q', s')$. This can be generalized to multiple automata.

The states of $\mathcal{M}(S_m)$ are the states of A and its initial state and alphabet are kept. The may arcs $\to_\Diamond$ are based on the arcs of A. However, all arcs that reach states which belong to final states of A_I are removed. This ensures that the allowed behaviour between S_m and $\mathcal{M}(S_m)$ is the same. $\mathcal{M}(S_m)$ is deterministic because A is deterministic. For $\to_\Box$, for each $a \in \Sigma$ look at each state s containing a final state q of A_{C_a}. Because S_m is coherent, the arc $s \xrightarrow{a}_\Diamond$ was not removed in the previous step. The (unique) arc $s \xrightarrow{a} s'$ is added to $\to_\Box$.

The correctness of both constructions can be shown. For this, lemma 27 is useful. ◀

▶ **Example 31.** Figure 3 shows a deterministic modal transition system $\mathcal{M}$. We will now construct $S_m(\mathcal{M})$. We have $S(a) = S(b) = \{1\}$, since for both letters this is the only state

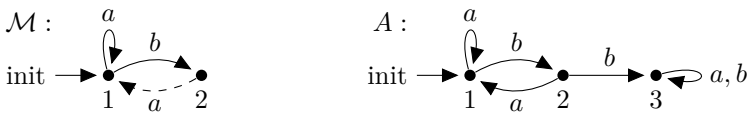

Figure 3 On the left, a modal transition system $\mathcal{M}$ is shown. May arcs are dashed and solid lines represent may and must arcs.On the right, the product automaton A being the synchronous product of A_{C_a}, A_{C_b} and A_I is depicted. For A_{C_a} and A_{C_b}, state 1 is the only final state. For A_I, this is state 3.

with an outgoing must arc, so $C_a = C_b = L_f(A_{\mathcal{M}}^{S(a)})$. The reader may easily verify that $L_f(A_{\mathcal{M}}^{S(a)}) = L((a+ba)^*)$. For I, derive $L(\overline{\mathcal{M}}) = L((a+ba)^*(\varepsilon+b))$ and $I = \Sigma^* \setminus L(\overline{\mathcal{M}}) = L((a+ba)^*bb(a+b)^*)$. This finishes the construction of $S_m(\mathcal{M}) = (\{C_e\}_{e\in\{a,b\}}, I)$

For the other direction, we begin with the modal specification $S_m := S_m(\mathcal{M})$ that was just derived and construct $\mathcal{M}(S_m)$. The synchronous product of A_{C_a}, A_{C_b} and A_I is shown in Figure 3. The may arcs of $\mathcal{M}(S_m)$ are all arcs of A except those reaching a final state of A_I. Thus, the arc from state 2 to state 3 and the two loops around state 3 are removed. For the must arcs, the arcs leaving a final state of A_{C_a} (resp. A_{C_b}) labelled with the corresponding event are considered. These are both arcs leaving state 1. The only may arc which is not also a must arc is the arc from state 2 to state 1. The constructed modal transition system $\mathcal{M}(S_m)$ is the same as $\mathcal{M}$ from Figure 3, except that it also contains an isolated state 3.

6 Conclusion

In this paper, the problem of finding a bounded Petri net implementing a given deterministic modal transition system was shown to be undecidable. This was done by first showing this problem to be undecidable for the conjunctive nu-calculus. The main result followed via an equivalence between the nu-calculus and deterministic modal transition systems. This result also settles the undecidability for the more powerful class of non-deterministic modal transition systems.

Several interesting questions are still open. The author believes that the presented approach can also be applied to the pure and bounded Petri net synthesis problem. The main difficulty is the zero-test transition, which would have to be split into two parts.

Another direction is to find decidable subcases of the Petri net synthesis problem. This could be done via structural restrictions on the modal transition systems or via limiting the class of synthesised Petri nets. For example, for a given k, the k-bounded Petri net synthesis problem from modal transition systems is decidable, because k-bounded Petri nets can be restricted to arc weights $\leq k$ without loss of generality. Only a finite number of Petri nets remain and each can be model-checked against the specification.

Possible structural restrictions on the modal transition systems might forbid must-arcs to form loops or might require each specification to be reachable via must-arcs.

Acknowledgements. I am grateful for the thoughtful remarks and suggestions by Eric Badouel, Eike Best, Valentin Spreckels, and Harro Wimmel. Also, I'd like to thank the anonymous reviewers for their helpful comments.

References

1 Adam Antonik, Michael Huth, Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. 20 years of modal and mixed specifications. *Bulletin of the EATCS*, 95:94–129, 2008.

2 André Arnold and Damian Niwiński. *Rudiments of μ-calculus*. North Holland, 2001.

3 Eric Badouel, Luca Bernardinello, and Philippe Darondeau. *Petri Net Synthesis*. Springer, 2015. doi:10.1007/978-3-662-47967-4.

4 Eric Badouel, Benoît Caillaud, and Philippe Darondeau. Distributing finite automata through petri net synthesis. *Formal Aspects of Computing*, 13(6):447–470, 2002. doi:10.1007/s001650200022.

5 Nikola Benes, Benoît Delahaye, Uli Fahrenberg, Jan Kretínský, and Axel Legay. Hennessy-milner logic with greatest fixed points as a complete behavioural specification theory. In Pedro R. D'Argenio and Hernán C. Melgratti, editors, *CONCUR 2013*, volume 8052 of *LNCS*, pages 76–90. Springer, 2013. doi:10.1007/978-3-642-40184-8_7.

6 Nikola Benes, Jan Kretínský, Kim Larsen, and Jirí Srba. On determinism in modal transition systems. *Theoretical Computer Science*, 410(41):4026–4043, 2009. doi:10.1016/j.tcs.2009.06.009.

7 Eike Best and Philippe Darondeau. Petri net distributability. In Edmund M. Clarke, Irina Virbitskaite, and Andrei Voronkov, editors, *PSI 2011, Revised Selected Papers*, volume 7162 of *LNCS*, pages 1–18. Springer, 2011. doi:10.1007/978-3-642-29709-0_1.

8 Glenn Bruns. An industrial application of modal process logic. *Science of Computer Programming*, 29(1-2):3–22, 1997. doi:10.1016/S0167-6423(96)00027-5.

9 Philippe Darondeau. Distributed implementations of Ramadge-Wonham supervisory control with Petri nets. In Eduardo Camacho, editor, *Proceedings of the 44th IEEE CDC-ECC 2005*, pages 2107–2112. IEEE, 2005. doi:10.1109/CDC.2005.1582472.

10 Javier Esparza. On the decidability of model checking for several μ-calculi and petri nets. In Sophie Tison, editor, *CAAP 1994*, volume 787 of *LNCS*, pages 115–129. Springer, 1994. doi:10.1007/BFb0017477.

11 Guillaume Feuillade. Modal specifications are a syntactic fragment of the Mu-calculus. Research Report RR-5612, INRIA, 2005. URL: https://hal.inria.fr/inria-00070396.

12 Guillaume Feuillade. *Spécification logique de réseaux de Petri*. PhD thesis, Université de Rennes I, 2005.

13 Guillaume Feuillade and Sophie Pinchinat. Modal specifications for the control theory of discrete event systems. *DEDS*, 17(2):211–232, 2007. doi:10.1007/s10626-006-0008-6.

14 John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, Mass, 1st edition, 1979.

15 Petr Jancar. Nonprimitive recursive complexity and undecidability for petri net equivalences. *Theoretical Computer Science*, 256(1-2):23–30, 2001. doi:10.1016/S0304-3975(00)00100-6.

16 Dexter Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983. doi:10.1016/0304-3975(82)90125-6.

17 Jan Kretínský and Salomon Sickert. MoTraS: A tool for modal transition systems and their extensions. In Dang Van Hung and Mizuhito Ogawa, editors, *ATVA 2013*, volume 8172 of *LNCS*, pages 487–491. Springer, 2013. doi:10.1007/978-3-319-02444-8_41.

18 Kim Larsen. Modal specifications. In Joseph Sifakis, editor, *AVMFSS*, volume 407 of *LNCS*, pages 232–246. Springer, 1989. doi:10.1007/3-540-52148-8_19.

19 Kim Larsen and Bent Thomsen. A modal process logic. In *LICS 1988*, pages 203–210. IEEE, 1988. doi:10.1109/LICS.1988.5119.

20 Marvin Lee Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1967.

21 Rob J. van Glabbeek, Ursula Goltz, and Jens-Wolfhard Schicke-Uffmann. On characterising distributability. *Logical Methods in Computer Science*, 9(3), 2013. doi:10.2168/LMCS-9(3:17)2013.