On the Hardness of Synthesizing Boolean Nets

Ronny Tredup and Christian Rosenke

Universität Rostock, Institut für Informatik, Theoretische Informatik,
Albert-Einstein-Straße 22, 18059, Rostock

Abstract. Boolean Petri nets are differentiated by types of nets based on which of the interactions nop, inp, out, set, res, swap, used, and free they apply or spare. From the 256 thinkable types only a few have yet been explicitly defined, as for instance contextual nets {nop, inp, out, used, free} and trace nets {nop, inp, out, set, res, used, free}. The synthesis problem relative to a specific type of nets τ is to find, for a given transition system A, a boolean τ-net with state graph isomorphic to A. It is known to be NP-hard for elementary nets systems {nop, inp, out} and tractable for flip-flop nets {nop, inp, out, swap}. This paper presents a general reduction scheme for the NP-hardness of boolean net synthesis and identifies 67 new types with a hard synthesis problem.

1 Introduction

Boolean Petri nets have been widely regarded as a fundamental model for concurrent systems. These Petri nets allow at most one token per place in every reachable marking. Accordingly, a place p can be regarded as a boolean condition which is *true* if p contains a token and is *false* if p is empty, respectively. A place p and a transition t of a boolean Petri net are connected by one of the following (boolean) *interactions*: *no operation* (nop), *input* (inp), *output* (out), set, *reset* (res), *inverting* (swap), *test if true* (used), and *test if false* (free). An interaction defines which pre-condition p has to satisfy to activate t and it determines p's post-condition after t has fired: inp (out) mean that p has to be *true* (*false*) to allow t's firing and if t fires then p become *false* (*true*). The interaction free (used) says that if t is activated then p is *false* (*true*) and t's firing as no impact on p. The other interactions nop, set, res, swap are pre-condition free, that is, neither *true* nor *false* prevent t's firing. Moreover, nop means that the firing of t has no impact and leaves p's boolean value unchanged. By res (set), t's firing determine p to be *false* (*true*). Finally, swap says that if t fires then it inverts p's boolean value.

Boolean Petri nets are differentiated by types of nets τ accordingly to the boolean interactions they allow. Since we have eight interactions to choose from, this results in a total of 256 different types. Yet, research has explicitly defined seven of them: *Elementary net systems* {nop, inp, out} [10], *Contextual nets* {nop, inp, out, used, free} [7], *event/condition nets* {nop, inp, out, used} [2], *inhibitor nets* {nop, inp, out, free} [9], *set nets* {nop, inp, set, used} [6], *trace nets* {nop, inp, out, set, res, used, free} [3], and *flip flop nets* {nop, inp, out, swap} [11].

Type of net τ	Complexity status	Quantity
$\tau = \{\mathsf{nop}, \mathsf{res}\} \cup \omega$, $\omega \subseteq \{\mathsf{inp}, \mathsf{used}, \mathsf{free}\}$	polynomial time	8
$\tau = \{\mathsf{nop}, \mathsf{set}\} \cup \omega$, $\omega \subseteq \{\mathsf{out}, \mathsf{used}, \mathsf{free}\}$	polynomial time	8
$\tau = \{\mathsf{nop}, \mathsf{swap}\} \cup \omega$, $\omega \subseteq \{\mathsf{inp}, \mathsf{out}, \mathsf{used}, \mathsf{free}\}$	polynomial time	16
$\tau = \{\mathsf{nop}\} \cup \omega$, $\omega \subseteq \{\mathsf{used}, \mathsf{free}\}$	polynomial time	4
$\tau = \{\mathsf{nop}, \mathsf{inp}, \mathsf{free}\}$ or $\tau = \{\mathsf{nop}, \mathsf{inp}, \mathsf{used}, \mathsf{free}\}$	NP-complete	2
$\tau = \{\mathsf{nop}, \mathsf{out}, \mathsf{used}\}$ or $\tau = \{\mathsf{nop}, \mathsf{out}, \mathsf{used}, \mathsf{free}\}$	NP-complete	2
$\tau = \{\mathsf{nop}, \mathsf{set}, \mathsf{res}\} \cup \omega$, $\emptyset \neq \omega \subseteq \{\mathsf{used}, \mathsf{free}\}$	NP-complete	3
$\tau = \{\mathsf{nop}, \mathsf{inp}, \mathsf{out}\} \cup \omega$, $\omega \subseteq \{\mathsf{used}, \mathsf{free}\}$	NP-complete	4
$\tau = \{\mathsf{nop}, \mathsf{inp}, \mathsf{res}, \mathsf{swap}\} \cup \omega$, $\omega \subseteq \{\mathsf{used}, \mathsf{free}\}$	NP-complete	4
$\tau = \{\mathsf{nop}, \mathsf{out}, \mathsf{set}, \mathsf{swap}\} \cup \omega$, $\omega \subseteq \{\mathsf{used}, \mathsf{free}\}$	NP-complete	4
$\tau = \{\mathsf{nop}, \mathsf{inp}, \mathsf{set}\} \cup \omega$, $\omega \subseteq \{\mathsf{out}, \mathsf{res}, \mathsf{swap}, \mathsf{used}, \mathsf{free}\}$	NP-complete	24+8
$\tau = \{\mathsf{nop}, \mathsf{outres}\} \cup \omega$, $\omega \subseteq \{\mathsf{inp}, \mathsf{set}, \mathsf{swap}, \mathsf{used}, \mathsf{free}\}$	NP-complete	24+8

Fig. 1. Summary of the complexity results for boolean net synthesis. Grey colored area: Results of [14] reestablishing the result for flip flop nets [11]. Green colored area: Results of this paper. The last two rows intersect in eight supersets of $\{\mathsf{nop}, \mathsf{inp}, \mathsf{out}, \mathsf{set}, \mathsf{res}\}$ and the eighth row includes the already investigated elementary net systems [1]. Altogether, this paper discovers 67 *new* types with an NP-hard synthesis problem.

This paper is devoted to a computational complexity analysis of the boolean net synthesis problem subject to a target type of nets. Synthesis relative to a specific type of nets τ is the challenge to find for a given *transition system* (TSs, for short) A, a boolean τ-net N whose state graph is isomorphic to A if it exists. The complexity of boolean net synthesis has originally been investigated for elementary net systems [1], where it is NP-complete to decide if general TSs can be synthesized. In [15, 12] this has been confirmed even for strongly restricted input TSs. On the contrary, [11] shows for flip flop nets, simply extending elementary net systems by swap, synthesis is doable in polynomial time. Inspired by these results, it is the (global) goal of our research to obtain a dichotomy result that fully characterizes which synergies of interactions make synthesis intractable or tractable, respectively. After resolving the complexity of synthesis for elementary nets [1, 15, 12] and flip flop nets [11], the next big step towards a complete characterization of boolean net synthesis is taken in [14]. Here, besides the already investigated flip flop nets, 42 further boolean types of nets are covered at one blow, cf. Figure 1. Besides the 128 practically less relevant types without nop, there are 84 nop-afflicted boolean types of nets left where the synthesis' complexity has not been settled, yet.

In this paper, we tackle 67 of them with a common NP-hardness proof scheme and, additionally, reestablish the result for elementary net systems [1]. Except flip flop nets, our result covers all types of nets previously considered in the literature. In particular, we show that synthesis is hard for all types that are a superset of $\{\mathsf{nop}, \mathsf{inp}, \mathsf{out}\}$ that excludes swap and for all supersets of $\{\mathsf{nop}, \mathsf{inp}, \mathsf{set}\}$ and $\{\mathsf{nop}, \mathsf{out}, \mathsf{res}\}$, cf. Figure 1.

Aside from the actual identification of 67 new types with hard net synthesis, this paper's contribution is also a very generic reduction scheme for NP-hardness

proofs of boolean net synthesis. This methodology significantly generalizes preliminary methods that we developed in [15, 12] to derive the hardness of synthesizing elementary net systems from strongly restricted TSs. Unlike those premature approaches, the present solution abstracts from individual types of nets and bases on the throughout analysis of properties gained by available interactions. Moreover, the approach from [14], although again slightly similar, is incompatible with the generic scheme in this paper.

To develop the generic reduction scheme, we deal with the synthesis problem's decision version, called *feasibility*. The reason is that complexity analysis rather works with decision problems than search problems. Instead of really finding a net N with state graph isomorphic to a given TS, it is sufficient for feasibility to just decide if the target type contains N. If feasibility is NP-complete, then synthesis is an NP-hard computational problem with no obvious efficient solutions.

To simplify our argumentation is it meaningful to detach our notions from Petri nets and focus on TSs. For this purpose, we use the well known equality between feasibility and the conjunction of the state separation property (SSP) and the event state separation property (ESSP) [2], which are solely defined on the input TSs. The presented polynomial time reduction scheme translates the NP-complete cubic monotone one-in-three 3-SAT problem [8] into the ESSP of the considered 68 boolean net types. As we also make sure that given boolean expressions φ are transformed to TSs $A(\varphi)$ where the ESSP relative to the considered type implies the SSP, we always show the NP-completeness of the ESSP and feasibility at the same time. Instead of 68 individual proofs, our scheme covers all cases by just three reductions following a common pattern.

Due to space limitation, we are not able to present all proofs. However, all omitted proofs are given in the technical report [13].

2 Preliminary Notions

This section provides short formal definitions of all preliminary notions used in the paper. A *transition system* (TS, for short), $A = (S, E, \delta)$ is a directed labeled graph with nodes S, events E and partial transition function $\delta : S \times E \longrightarrow S$, where $\delta(s, e) = s'$ is interpreted as $s \xrightarrow{e} s'$. An event e *occurs* at a state s, denoted by $s \xrightarrow{e}$, if $\delta(s, e)$ is defined. An *initialized* TS $A = (S, E, \delta, s_0)$ is a TS with a distinct state $s_0 \in S$. TSs in this paper are *deterministic* by design as their state transition behavior is given by a (partial) function. Initialized TSs are also required to make every state *reachable* from s_0 by a directed path.

x	$\mathsf{nop}(x)$	$\mathsf{inp}(x)$	$\mathsf{out}(x)$	$\mathsf{set}(x)$	$\mathsf{res}(x)$	$\mathsf{swap}(x)$	$\mathsf{used}(x)$	$\mathsf{free}(x)$
0	0		1	1	0	1		0
1	1	0		1	0	0	1	

Fig. 2. All interactions in I. An empty cell means that the column's function is undefined on the respective x. The entirely undefined function is missing in I.

A (boolean) *type of net* $\tau = (\{0,1\}, E_\tau, \delta_\tau)$ is a TS such that E_τ is a subset of the boolean interactions: $E_\tau \subseteq I = \{\mathsf{nop}, \mathsf{inp}, \mathsf{out}, \mathsf{set}, \mathsf{res}, \mathsf{swap}, \mathsf{used}, \mathsf{free}\}$. The interactions $i \in I$ are binary partial functions $i : \{0,1\} \to \{0,1\}$ as defined in the listing of Figure 2. For all $x \in \{0,1\}$ and all $i \in E_\tau$ the transition function of τ is defined by $\delta_\tau(x,i) = i(x)$. Notice that I contains all possible binary partial functions $\{0,1\} \to \{0,1\}$ except for the entirely undefined function $\perp$. Even if a type τ includes $\perp$, this event can never occur, so it would be useless. Thus, I is complete for deterministic boolean types of nets, and that means there are a total of 256 of them. By definition, a (boolean) type τ is completely determined by its event set E_τ. Hence, in the following we will identify τ with E_τ, cf. Figure 3. Moreover, for readability, we group interactions by $\mathsf{enter} = \{\mathsf{out}, \mathsf{set}, \mathsf{swap}\}$, $\mathsf{exit} = \{\mathsf{inp}, \mathsf{res}, \mathsf{swap}\}$, $\mathsf{keep}^+ = \{\mathsf{nop}, \mathsf{set}, \mathsf{used}\}$, and $\mathsf{keep}^- = \{\mathsf{nop}, \mathsf{res}, \mathsf{free}\}$.

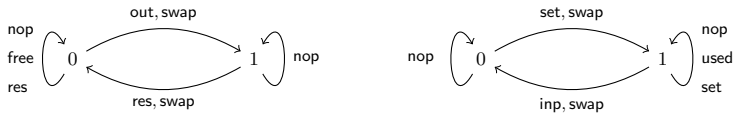


Fig. 3. Left: $\tau = \{\mathsf{nop}, \mathsf{out}, \mathsf{res}, \mathsf{swap}, \mathsf{free}\}$. Right: $\tilde{\tau} = \{\mathsf{nop}, \mathsf{inp}, \mathsf{set}, \mathsf{swap}, \mathsf{used}\}$. τ and $\tilde{\tau}$ are isomorphic. The isomorphism $\phi : \tau \to \tilde{\tau}$ is given by $\phi(s) = 1 - s$ for $s \in \{0,1\}$, $\phi(i) = i$ for $i \in \{\mathsf{nop}, \mathsf{swap}\}$, $\phi(\mathsf{out}) = \mathsf{inp}$, $\phi(\mathsf{res}) = \mathsf{set}$ and $\phi(\mathsf{free}) = \mathsf{used}$.

A boolean Petri net $N = (P, T, M_0, f)$ of type τ, (τ-net, for short) is given by finite and disjoint sets P of places and T of transitions, an initial marking $M_0 : P \longrightarrow \{0,1\}$, and a (total) flow function $f : P \times T \to \tau$. The meaning of a boolean net is to realize a certain behavior by firing sequences of transitions. In particular, a transition $t \in T$ can fire in a marking $M : P \longrightarrow \{0,1\}$ if $\delta_\tau(M(p), f(p,t))$ is defined for all $p \in P$. By firing, t produces the next marking $M' : P \longrightarrow \{0,1\}$ where $M'(p) = \delta_\tau(M(p), f(p,t))$ for all $p \in P$. This is denoted by $M \xrightarrow{t} M'$. Given a τ-net $N = (P, T, M_0, f)$, its behavior is captured by a transition system $A(N)$, called the state graph of N. The state set of $A(N)$ consists of all markings that, starting from initial state M_0, can be reached by firing a sequence of transitions. For every reachable marking M and transition $t \in T$ with $M \xrightarrow{t} M'$ the state transition function δ of A is defined as $\delta(M,t) = M'$.

Boolean net synthesis for a type τ is going backwards from input TS $A = (S, E, \delta, s_0)$ to the computation of a τ-net N with $A(N)$ isomorphic to A, if such a net exists. In contrast to $A(N)$, the abstract states S of A miss any information about markings they stand for. Accordingly, the events E are an abstraction of N's transitions T as they relate to state changes only globally without giving the information about the local changes to places. After all, the transition function $\delta : S \times E \to S$ still tells us how states are affected by events.

To prove net synthesis of τ-nets NP-hard, we show the NP-completeness of the corresponding decision version: *τ-feasibility* is the problem to decide the existence of a τ-net N with $A(N)$ isomorphic to the given TS A. To describe feasibility

without referencing the searched τ-net N, in the sequel, we introduce the *τ-state separation property* (*τ-SSP*, for short) and the *τ-event state separation property* (*τ-ESSP*, for short) for TSs. In conjunction, they are equivalent to τ-feasibility. The following notion of τ-regions allows us to define the announced properties.

A τ-region of a given $A = (S, E, \delta, s_0)$ is a pair (sup, sig) of *support* $sup : S \to S_\tau = \{0, 1\}$ and *signature* $sig : E \to E_\tau = \tau$ where every transition $s \xrightarrow{e} s'$ of A leads to a transition $sup(s) \xrightarrow{sig(e)} sup(s')$ of τ. While a region divides S into the two sets $sup^{-1}(b) = \{s \in S \mid sup(s) = b\}$ for $b \in \{0, 1\}$, the events are cumulated by $sig^{-1}(i) = \{e \in E \mid sig(e) = i\}$ for all available interactions $i \in \tau$. We also use $sig^{-1}(\tau') = \{e \in E \mid sig(e) \in \tau'\}$ for $\tau' \subseteq \tau$.

For a TS $A = (S, E, \delta, s_0)$ and a type of nets τ, a pair of states $s \neq s' \in S$ is *τ-separable* if there is a τ-region (sup, sig) such that $sup(s) \neq \sup(s')$. Accordingly, A has the τ-SSP if all pairs of distinct states from A are τ-separable. Secondly, an event $e \in E$ is called *τ-inhibitable* at a state $s \in S$ if there is a τ-region (sup, sig) where $sup(s) \xrightarrow{sig(e)}$ does not hold, that is, the interaction $sig(e) \in \tau$ is not defined on input $sup(s) \in \{0, 1\}$. A has the τ-ESSP if for all states $s \in S$ it is true that all events $e \in E$ that do not occur at s, meaning $\neg s \xrightarrow{e}$, are τ-inhibitable at s. It is well known from [2] that a TS A is τ-feasible, that is, there exists a τ-net N with $A(N)$ isomorphic to A, if and only if A has τ-SSP *and* the τ-ESSP. Types of nets τ and $\tilde{\tau}$ have an isomorphism ϕ if $s \xrightarrow{i} s'$ is a transition in τ if and only if $\phi(s) \xrightarrow{\phi(i)} \phi(s')$ is one in TS $\tilde{\tau}$. By the following lemma, we benefit from the eight isomorphisms that map nop to nop, swap to swap, inp to out, set to res, used to free, and vice versa:

Lemma 1 (Without proof). *If τ and $\tilde{\tau}$ are isomorphic types of nets then a TS A has the (E)SSP for τ if and only if A has the (E)SSP for $\tilde{\tau}$.*

3 Main Result

Theorem 1 (Main result). *Let $\tau_1 = \{\mathsf{nop}, \mathsf{inp}, \mathsf{out}\}$, $\tau_2 = \{\mathsf{nop}, \mathsf{inp}, \mathsf{res}, \mathsf{swap}\}$, $\tilde{\tau}_2 = \{\mathsf{nop}, \mathsf{out}, \mathsf{set}, \mathsf{swap}\}$, $\tau_3 = \{\mathsf{nop}, \mathsf{inp}, \mathsf{set}\}$ and $\tilde{\tau}_3 = \{\mathsf{nop}, \mathsf{out}, \mathsf{res}\}$. If $\tau = \tau' \cup \omega$ for $\tau' \in \{\tau_1, \tau_2, \tilde{\tau}_2\}$ with $\omega \subseteq \{\mathsf{used}, \mathsf{free}\}$ or $\tau \supseteq \tau_3$ or $\tau \supseteq \tilde{\tau}_3$ then τ-feasibility is NP-complete.*

In total, Theorem 1 covers 68 types, cf. Figure 1, including the *elementary net systems* [1]. It is straight forward that τ-feasibility is a member of NP for all considered type of nets τ: By definition, there are at most $|S|^2$ pairs of states (s, s') to separate and at most $|E| \cdot |S|$ pairs (e, s) of event and state where e has to be inhibited at s. In a non-deterministic computation, one can simply guess and check in polynomial time for all pairs the region that separates s and s', respectively inhibits e at s, or reject the input if such a region does not exist. Hence, for the proof of Theorem 1 it remains to prove τ-feasibility to be NP-hard for all types of nets. Although this demands for 68 NP-hardness proofs, we manage to reduce it to three. Every reduction bases on one scheme using the

cubic monotone one-in-three-3-SAT problem, (P1, for short), which has been shown to be NP-hard in [8]:

CUBIC MONOTONE ONE-IN-THREE-3-SAT (P1)

Instance: negation-free boolean expression $\varphi = \{\zeta_0, \ldots, \zeta_{m-1}\}$ of three-clauses $\zeta_0, \ldots, \zeta_{m-1}$ with variable set $V(\varphi)$, every variable occurs in exactly three clauses

Question: Is there a subset $M \subseteq V(\varphi)$ such that $|M \cap \zeta_i| = 1$ for $i \in \{0, \ldots, m-1\}$?

Starting from a common construction principle, we choose one of our three reductions by a turn-switch σ. In every switch position $\sigma_1, \sigma_2, \sigma_3$, the chosen reduction works for multiple boolean types based on mutually shared interactions and isomorphisms. Before we set out the details, the following subsection introduces our way of easily combining gadget TSs for our NP-completeness proofs.

3.1 Unions of Transition Systems

If $A_0 = (S_0, E_0, \delta_0, s_0^0), \ldots, A_n = (S_n, E_n, \delta_n, s_0^n)$ are (initialized) TSs with pairwise disjoint states (but not necessarily disjoint events) we say that $U(A_0, \ldots, A_n)$ is their *union*. By $S(U)$, we denote the entirety of all states in $A_0, \ldots, A_n$ and $E(U)$ summarizes all events. For a flexible formalism, we allow to build unions recursively: Firstly, we allow empty unions and identify every TS A with the union containing only A, that is, $A = U(A)$. Next, if $U_1 = U(A_0^1, \ldots, A_{n_1}^1), \ldots, U_m = (A_0^m, \ldots, A_{n_m}^n)$ are unions (possibly with $U_i = U()$ or $U_i = A_i$) then $U(U_1, \ldots, U_m)$ is the union $U(A_0^1, \ldots, A_{n_1}^1, \ldots, A_0^m, \ldots, A_{n_m}^n)$.

We lift the concepts of regions, SSP, and ESSP to unions $U = U(A_0, \ldots, A_n)$ as follows: A τ-region (sup, sig) of U consists of $sup : S(U) \to S_\tau$ and $sig : E(U) \to E_\tau$ such that, for all $i \in \{0, \ldots, n\}$, the projections $sup_i(s) = sup(s), s \in S_i$ and $sig_i(e) = sig(e), e \in E_i$ provide a region (sup_i, sig_i) of A_i. U has the τ-SSP if for all different states $s, s' \in S(U)$ of the same TS A_i there is a τ-region (sup, sig) of U with $sup(s) \neq sup(s')$. Moreover, U has the τ-ESSP if for all events $e \in E(U)$ and all states $s \in S(U)$ with $\neg s \xrightarrow{e}$ there is a τ-region (sup, sig) of U such that $\neg sup(s) \xrightarrow{sig(e)}$. Naturally, U is called τ-feasible if it has the τ-SSP *and* τ-ESSP. To merge a union $U = U(A_0, \ldots, A_n)$ into a single TS, we define the joining $A(U)$: If $s_0^0, \ldots, s_0^n$ are the initial states of U's TSs then $A(U) = (S(U) \cup \bot, E(U) \cup \odot \cup \ominus, \delta, \bot_0)$ is a TS with additional connector states $\bot = \{\bot_0, \ldots, \bot_n\}$ and fresh events $\odot = \{\odot_0, \ldots, \odot_n\}$, $\ominus = \{\ominus_1, \ldots, \ominus_n\}$ joining the individual TSs of U by δ as defined in Figure 4.

$$\delta(s,e) = \begin{cases} s_0^i, & \text{if } s = \bot_i \text{ and } e = \odot_i, \\ \bot_{i+1}, & \text{if } s = \bot_i \text{ and } e = \ominus_{i+1}, \\ \delta_i(s,e), & \text{if } s \in S_i \text{ and } e \in E_i, \end{cases}$$

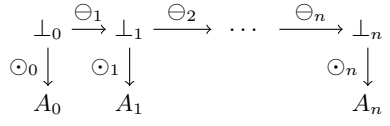


Fig. 4. Left: $A(U)$'s transition function δ. Right: An abstract representation of $A(U)$.

Hence, $A(U)$ puts the connector states into a chain of the events from $\ominus$ and links the initial states of TSs from U to this chain using events from $\odot$. The following lemma certifies the validity of the joining operation for the unions and the types of nets that occur in our reduction scheme.

Lemma 2. *Let τ be a type of nets such that $\mathsf{nop}, \mathsf{inp} \in \tau$ and $\tau \cap \mathsf{enter} \neq \emptyset$. If $U = U(A_0, \ldots, A_n)$ is a union of TSs $A_0, \ldots, A_n$ where, for every event e in $E(U)$, there is at least one state s in $S(U)$ with $\neg s \xrightarrow{e}$ then U has the τ-(E)SSP if and only if $A(U)$ has the τ-(E)SSP.*

Notice that Lemma 2 does not cover all types mentioned in Theorem 1. However, this is not necessary as every type τ from Theorem 1 missed by Lemma 2 is indirectly covered in the Lemma by an isomorphic type $\tilde{\tau}$.

3.2 The General Reduction Scheme

Our general scheme can be set up to a specific reduction by the turn switch σ. In each of its three positions, σ covers a whole collection of net types. Therefore, we simply understand the positions $\sigma_1, \sigma_2, \sigma_3$ as the type sets managed by the respective reductions:

$$\sigma_1 = \{\tau_1 \cup \omega \mid \omega \subseteq \{\mathsf{used}, \mathsf{free}\}\} \cup \{\tau_3 \cup \omega \mid \omega \subseteq \{\mathsf{out}, \mathsf{res}, \mathsf{used}, \mathsf{free}\}$$
$$\sigma_2 = \{\tau_3 \cup \{\mathsf{swap}\} \cup \omega \mid \omega \subseteq \{\mathsf{out}, \mathsf{res}, \mathsf{used}, \mathsf{free}\}\}$$
$$\sigma_3 = \{\tau_2 \cup \omega \mid \omega \subseteq \{\mathsf{used}, \mathsf{free}\}\}$$

The input of our scheme is the switch position $\sigma \in \{\sigma_1, \sigma_2, \sigma_3\}$ and an instance φ of P1. The result is a union U_φ^σ of gadget TSs satisfying the conditions of Lemma 2, that is, U_φ^σ is τ-feasible if and only if $A(U_\varphi^\sigma)$ is τ-feasible for all $\tau \in \sigma$. Moreover, the union U_φ^σ satisfies the following properties:

1. The variables $V(\varphi)$ are a subset of $E(U_\varphi^\sigma)$, the union events.
2. Event $k \in E(U_\varphi^\sigma)$ is to inhibit at state $h_{0,6} \in S(U_\varphi^\sigma)$ and there are events $V = \{v_0, \ldots, v_{3m-1}\} \subseteq E(U_\varphi^\sigma)$ and $W = \{w_0, \ldots, w_{3m-1}\} \subseteq E(U_\varphi^\sigma)$.
3. If $\tau \in \sigma$ and (sup, sig) a τ-region inhibiting k at $h_{0,6}$ then one of the following conditions is true:
   (a) $sig(k) = \mathsf{inp}$ and $V \subseteq sig^{-1}(\mathsf{enter})$ and $W \subseteq sig^{-1}(\mathsf{keep}^-)$,
   (b) $sig(k) = \mathsf{out}$ and $V \subseteq sig^{-1}(\mathsf{exit})$ and $W \subseteq sig^{-1}(\mathsf{keep}^+)$.
4. If (sup, sig) is a region of U_φ^σ satisfying Condition 3.a or Condition 3.b then $M = \{X \in V(\varphi) \mid sig(X) \neq \mathsf{nop}\}$ is a one-in-three model of φ.
5. If φ has a one-in-three model M and $\tau \in \sigma$ then there is a τ-region (sup, sig) with $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$. Especially, (sup, sig) inhibits k at $h_{0,6}$ and satisfies Condition 3.a.
6. If $\tau \in \sigma$ and k is τ-inhibitable at $h_{0,6}$ then U_φ^σ has the τ-ESSP and the τ-SSP.

A polynomial time reduction scheme with these properties proves Theorem 1 as the following implications are justified:

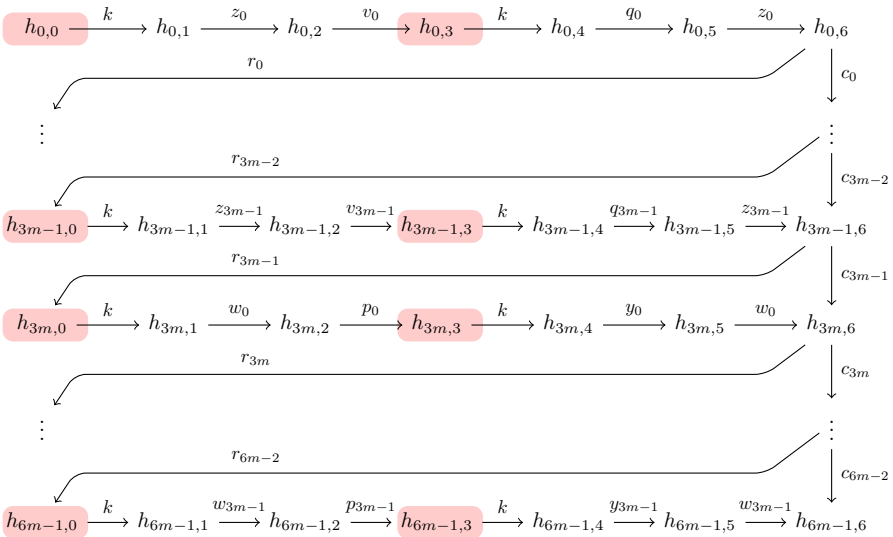
φ is one-in-three satisfiable $\overset{5.}{\Longrightarrow}$ k is τ-inhibitable at $h_{0,6}$ in U_φ^σ $\overset{6.}{\Longrightarrow}$ U_φ^σ has the τ-ESSP & τ-SSP $\overset{\text{def.}}{\Longrightarrow}$ U_φ^σ is τ-feasible $\overset{\text{def.}}{\Longrightarrow}$ U_φ^σ has the τ-ESSP $\overset{\text{def.}}{\Longrightarrow}$ k is τ-inhibitable at $h_{0,6}$ in U_φ^σ $\overset{3./4.}{\Longrightarrow}$ φ is one-in-three satisfiable.

Especially, φ is one-in-three satisfiable if and only if U_φ^σ is τ-feasible. By Lemma 2, this proves NP-hardness of τ-feasibility for all τ in the positions $\sigma_1, \sigma_2, \sigma_3$. Secondly, every remaining type $\tilde{\tau}$ of Theorem 1 is isomorphic to one of the already covered cases τ. Hence, by Lemma 1, this also proves NP-hardness of $\tilde{\tau}$-feasibility which, by feasibility being in NP, justifies Theorem 1.

In the sequel, we develop the reduction of U_φ^σ and show that it satisfies Condition 1-Condition 5. Notice that this proves φ is one-in-three satisfiable if and only if k is inhibitable at $h_{0,6}$. Due to space limitation, the proof that U_φ^σ also has Condition 6 is moved to the technical report [13].

3.3 Details for Condition 2 and Condition 3

To satisfy Condition 2, every union U_φ^σ implements the following transition system H that provides the event k, the state $h_{0,6}$, where $\neg h_{0,6} \xrightarrow{k}$, and the events of $Z = \{z_0, \ldots, z_{3m-1}\}$, $V = \{v_0, \ldots, v_{3m-1}\}$ and $W = \{w_0, \ldots, w_{3m-1}\}$ (the colored areas are to be explained later):



So far Condition 2 is already satisfied. For Condition 3 we observe that, by definition, there are basically four interactions possibly useful for $sig(k)$: inp, out, used, free. The other interactions res, set, swap, nop are defined on both 0 and 1 and, hence, not suitable to inhibit events. H alone generally does not guarantee that a region (sup, sig) inhibiting k at $h_{0,6}$ satisfies Condition 3. Thus, to achieve this goal other gadgets are necessary. By theirs different types (having

different interactions), it depends on σ *which* gadgets are necessary. We proceed step by step and develop the construction for σ_1, σ_2 and σ_3 in the given order. In the sequel, if not explicitly stated otherwise, by (sup, sig) we refer to a τ-region of U^{σ}_{φ}, $\tau \in \sigma$, that inhibits k at $h_{0,6}$. The union $U^{\sigma_1}_{\varphi}$ implements additionally the following two TSs F_0, F_1:

$$F_0 = f_{0,0} \xrightarrow{k} f_{0,1} \xrightarrow{n_0} f_{0,2} \xrightarrow{z_0} f_{0,3} \xrightarrow{k} f_{0,4} \qquad F_1 = f_{1,0} \xrightarrow{q_0} f_{1,1} \xrightarrow{k} f_{1,2}$$

We argue that every region (sup, sig) of $U^{\sigma_1}_{\varphi}$ satisfies Condition 3: If $sig(k) = \mathsf{used}$ then, by definition of used we get $sup(s) = sup(s') = 1$ for every transition $s \xrightarrow{k} s'$. Hence, we have $sup(f_{0,3}) = sup(f_{1,1}) = sup(h_{0,4}) = 1$. By definition of $\mathsf{inp}, \mathsf{res}$ we have that $\xrightarrow{e} s$ and $sig(e) \in \{\mathsf{inp}, \mathsf{res}\}$ implies $sup(s) = 0$. Hence, by $\xrightarrow{z_0} f_{0,3}$ and $\xrightarrow{q_0} f_{1,1}$ we have that $sig(z_0), sig(q_0) \notin \{\mathsf{inp}, \mathsf{res}\}$. Moreover, we observe that $\mathsf{swap} \notin \tau$ for all $\tau \in \sigma_1$. Thus, $sig(z_0), sig(q_0) \in \mathsf{keep}^+$ such that $sup(h_{0,4}) = sup(h_{0,5}) = sup(h_{0,6}) = 1$ which contradicts $\neg sup(h_{0,6}) \xrightarrow{sig(k)}$. Hence, $sig(k) \neq \mathsf{used}$. Similarly, one argues that $sig(k) \neq \mathsf{free}$ implies that $sup(h_{0,6}) = 0$, again a contradiction. Hence, we have that $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$ or $sig(k) = \mathsf{out}$ and $sup(h_{0,6}) = 1$.

As a next step, we show that $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$ implies $sig(v_0) \in \mathsf{enter}$ and $sig(z_0) \in \mathsf{keep}^-$: By $sig(k) = \mathsf{inp}$ and $\xrightarrow{k} h_{0,1}$ and $h_{0,3} \xrightarrow{k}$ we have that $sup(h_{0,1}) = 0$ and $sup(h_{0,3}) = 1$. By $\xrightarrow{z_0} h_{0,6}$, $sup(h_{0,6}) = 0$ and $\mathsf{swap} \notin \tau$ we have that $sig(z_0) \in \mathsf{keep}^-$. Moreover, by $sup(h_{0,1}) = 0$ and $sig(z_0) \in \mathsf{keep}^-$ it is $sup(h_{0,2}) = 0$ which, by $h_{0,2} \xrightarrow{v_0} h_{0,3}$ and $sup(h_{0,3}) = 1$, implies $sig(v_0) \in \mathsf{enter}$. Notice, that this reasoning purely bases on $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$. Similarly, one argues that $sig(k) = \mathsf{out}$ and $sup(h_{0,6}) = 1$ implies $sig(v_0) \in \mathsf{exit}$ and $sig(z_0) \in \mathsf{keep}^+$. $U^{\sigma_1}_{\varphi}$ uses for every $i \in \{0, \ldots, 6m-2\}$ the following TS $G^{c,c}_i$ to transfer the support-value $sup(h_{0,6})$ to the states $h_{1,6}, \ldots, h_{6m-1,6}$:

$$G^{c,c}_i = \begin{array}{ccc} g^{c,c}_{i,0} & \xrightarrow{c_i} & g^{c,c}_{i,1} \\ {\scriptstyle k}\downarrow & & \downarrow{\scriptstyle k} \\ g^{c,c}_{i,2} & \xrightarrow[c_i]{} & g^{c,c}_{i,3} \end{array}$$

By doing so, it transfers the outcome of the latter observation to the events $v_1, \ldots, v_{3m-1}$ and $w_0, \ldots, w_{3m-1}$: If $sig(k) = \mathsf{inp}$, then we have $sup(g^{c,c}_{i,0}) = sup(g^{c,c}_{i,0}) = 1$ and $sup(g^{c,c}_{i,2}) = sup(g^{c,c}_{i,3}) = 0$, that is, $sig(c_i) = \mathsf{nop}$. Symmetrically, if $sig(k) = \mathsf{out}$ then $sig(c_i) = \mathsf{nop}$. Hence, if $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$ then $sup(h_{i,6}) = 0$ for all $i \in \{0, \ldots, 6m-1\}$. Perfectly similar to the discussion for z_0 and v_0 we obtain that $V \subseteq sig^{-1}(\mathsf{enter})$ and $W \subseteq sig^{-1}(\mathsf{keep}^-)$, respectively. Symmetrically, $sig(k) = \mathsf{out}$ and $sup(h_{0,6}) = 1$ imply $V \subseteq sig^{-1}(\mathsf{exit})$ and $W \subseteq sig^{-1}(\mathsf{keep}^+)$. Hence, $U^{\sigma_1}_{\varphi}$ satisfies Condition 3.

Unfortunately, if $\sigma \in \{\sigma_2, \sigma_3\}$ then the introduced gadgets F_0, F_1 (alone) are not powerful enough to ensure Condition 3. This is mainly due to the interaction swap. However, we tackle this problem by the application of other gadgets. While, due to theirs different interaction sets, σ_2 and σ_3 have different requirements,

both of $U_\varphi^{\sigma_2}$ and $U_\varphi^{\sigma_3}$ implement for every $i \in \{0, \dots, 6m-2\}$ the gadget $G_i^{c,c}$ and for $i \in \{0, \dots, 3m-1\}$ following gadget TSs $G_i^{\text{-},q}$ and $G_i^{\text{-},y}$:

$$G_i^{\text{-},q} = \begin{array}{ccc} g_{i,0}^{\text{-},q} & \xrightarrow{\text{-}} & g_{i,1}^{\text{-},q} \\ {\scriptstyle k}\downarrow & & \downarrow{\scriptstyle k} \\ g_{i,2}^{\text{-},q} & \xrightarrow[q_i]{} & g_{i,3}^{\text{-},q} \end{array} \qquad G_i^{\text{-},y} = \begin{array}{ccc} g_{i,0}^{\text{-},y} & \xrightarrow{\text{-}} & g_{i,1}^{\text{-},y} \\ {\scriptstyle k}\downarrow & & \downarrow{\scriptstyle k} \\ g_{i,2}^{\text{-},y} & \xrightarrow[y_i]{} & g_{i,3}^{\text{-},y} \end{array}$$

Here and in the sequel, the purpose of underscore-labeled edges is essentially to ensure reachability of the TSs. Every underscore represents an arbitrary unique event occurring only at this edge. For the sake of readability we do not define these events explicitly. Again for readability, we define $Q = \{q_0, \dots, q_{3m-1}\}$ and $Y = \{y_0, \dots, y_{3m-1}\}$. Moreover, $U_\varphi^{\sigma_2}$ adds the TS F_0 (originally introduced for σ_1) and the next TS $G_0^{\text{n},\text{-}}$ while $U_\varphi^{\sigma_3}$ uses also F_0 and the following TS F_2:

$$G_0^{\text{n},\text{-}} = \begin{array}{ccc} g_{0,0}^{\text{n},\text{-}} & \xrightarrow{n_0} & g_{0,1}^{\text{n},\text{-}} \\ {\scriptstyle k}\downarrow & & \downarrow{\scriptstyle k} \\ g_{0,2}^{\text{n},\text{-}} & \xrightarrow[\text{-}]{} & g_{0,3}^{\text{n},\text{-}} \end{array} \qquad F_2 = \begin{array}{ccc} f_{2,0} & \xrightarrow{n_0} & f_{2,1} \\ {\scriptstyle k}\downarrow & & \uparrow{\scriptstyle k} \\ f_{2,2} & \xrightarrow[\text{-}]{} & f_{2,3} \end{array}$$

That $U_\varphi^{\sigma_2}$ and $U_\varphi^{\sigma_3}$ differ in $G_0^{\text{n},\text{-}}$ and F_2, respectively, is mainly due to the fact that the interactions of theirs types have different requirements to satisfy Condition 5 and Condition 6. To argue for $U_\varphi^{\sigma_3}$'s and $U_\varphi^{\sigma_4}$'s functionality, respectively, we firstly show that $sig(k) \in \{\mathsf{inp}, \mathsf{out}\}$ for (sup, sig): If $sig(k) = \mathsf{used}$ then $s \xrightarrow{k} s'$ implies $sup(s) = sup(s') = 1$. Applying this to $F_2, G_0^{\text{n},\text{-}}$ and $G_0^{\text{-},q}$ we obtain that $sig(n_0), sig(q_0) \in \mathsf{keep}^+$. By $sig(n_0) \in \mathsf{keep}^+$ and $sup(f_{0,1}) = 1$ we obtain $sup(f_{0,2}) = 1$ which, by $sup(f_{0,3}) = 1$, implies $sig(z_0) \in \mathsf{keep}^+$. Finally, by $sup(h_{0,3}) = 4$ and $sig(z_0), sig(q_0) \in \mathsf{keep}^+$, we get $sup(h_{0,6}) = 1$ which contradicts $\neg \sup(h_{0,6}) \xrightarrow{sig(k)}$. Similarly, the assumption $sig(k) = \mathsf{free}$ yields the same contradiction. Thus, $sig(k) \in \{\mathsf{inp}, \mathsf{out}\}$.

We argue that $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$ implies $V \subseteq sig^{-1}(\mathsf{enter})$ and $W \subseteq sig^{-1}(\mathsf{keep}^-)$: By $sig(k) = \mathsf{inp}$ we get again $sig(c_i) = \mathsf{nop}$ and, thus, $sup(h_{i,6}) = 0$ for all i in question. Moreover, by $sig(k) = \mathsf{inp}$ we get $sup(s) = 0$ for all $\xrightarrow{k} s$. Applying this to $G_i^{\text{-},q}$ and $G_i^{\text{-},y}$ yields $Q, Y \subseteq sig^{-1}(\mathsf{keep}^-)$. By $Q, Y \subseteq sig^{-1}(\mathsf{keep}^-)$ and $sup(h_{i,4}) = 0$ (for all $i \in \{0, \dots, 6m-1\}$) we get $sup(h_{i,5}) = 0$ which, by $sup(h_{i,6}) = 0$, implies $W, Z \subseteq sig^{-1}(\mathsf{keep}^-)$. Finally, by $Z \subseteq sig^{-1}(\mathsf{keep}^-)$ and $sup(h_{i,1}) = 0$ we conclude $sup(h_{i,2}) = 0$ implying with $sup(h_{i,3}) = 1$ that $sig(v_i) \in \mathsf{enter}$ for every $i \in \{0, \dots, 3m-1\}$: $V \subseteq sig^{-1}(\mathsf{enter})$. Symmetrically, $sig(k) = \mathsf{out}$ and $sup(h_{0,6}) = 1$ implies $W \subseteq sig^{-1}(\mathsf{keep}^+)$ and $V \subseteq sig^{-1}(\mathsf{exit})$. Hence, $U_\varphi^{\sigma_2}$ and $U_\varphi^{\sigma_3}$ satisfy Condition 3.

3.4 Details for Condition 1 and Condition 4

Essential for the realization of Condition 1 and Condition 4 is the observation that for every τ-region $R = (sup, sig)$ and path $P = s \xrightarrow{e_1} \dots \xrightarrow{e_n} s'$ the image $R(P) = sup(s) \xrightarrow{sig(e_1)} \dots \xrightarrow{sig(e_n)} sup(s')$ is a path in τ. Moreover, if $sup(s) \neq sup(s') = 0$ then there has to be an event e_i mapped to an interaction of τ that allows a state change in τ: $sig(e_i) \notin \{\mathsf{nop}, \mathsf{used}, \mathsf{free}\}$. Our target is

to exploit this observation in the following way: Firstly, represent every clause $\zeta_i = \{X_{i,0}, X_{i,1}, X_{i,2}\}$ by a path $P_{i,0} = t_{i,0,2} \xrightarrow{X_{i,0}} t_{i,0,3} \xrightarrow{X_{i,1}} t_{i,0,4} \xrightarrow{X_{i,2}} t_{i,0,5}$. Secondly, ensure that region (sup, sig) (inhibiting k at $h_{0,6}$) requires $sup(t_{i,0,2}) \neq sup(t_{i,0,5})$ for every $i \in \{0, \ldots, m-1\}$. Thirdly, ensure that there is exactly one variable event $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ whose image $sig(X)$ allows the necessary state change in τ and that both of the others are mapped to nop. Such a region implies that $M = \{X \in V(\varphi) \mid sig(X) \neq \mathsf{nop}\}$ is a one-in-three model of φ as two different clauses ζ_i, ζ_j are represented by different paths $P_{i,0}, P_{j,0}$. Let's discuss the case that region (sup, sig) satisfies $sup(t_{i,0,2}) = 1$ and $sup(t_{i,0,5}) = 0$. Unfortunately, generally there are many possibilities for the signature of the variable events $X_{i,0}, X_{i,1}, X_{i,2}$ and not even one of them need to be mapped to nop. For example, $sig(X) = \mathsf{res}$ or $sig(X) = \mathsf{swap}$ for $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ would be possible, too.

We attack this problem as follows: Firstly, instead of one path for ζ_i we apply the following three forward-backward paths $P_{i,0}, P_{i,1}, P_{i,2}$ which for every variable event $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ use an additional mirror event x:

$$P_{i,0} = t_{i,0,2} \underset{x_{i,0}}{\overset{X_{i,0}}{\rightleftarrows}} t_{i,0,3} \underset{x_{i,1}}{\overset{X_{i,1}}{\rightleftarrows}} t_{i,0,4} \underset{x_{i,2}}{\overset{X_{i,2}}{\rightleftarrows}} t_{i,0,5} \qquad P_{i,1} = t_{i,1,2} \underset{x_{i,1}}{\overset{X_{i,1}}{\rightleftarrows}} t_{i,1,3} \underset{x_{i,2}}{\overset{X_{i,2}}{\rightleftarrows}} t_{i,1,4} \underset{x_{i,0}}{\overset{X_{i,0}}{\rightleftarrows}} t_{i,1,5}$$

$$P_{i,2} = t_{i,2,2} \underset{x_{i,2}}{\overset{X_{i,2}}{\rightleftarrows}} t_{i,2,3} \underset{x_{i,0}}{\overset{X_{i,0}}{\rightleftarrows}} t_{i,2,4} \underset{x_{i,1}}{\overset{X_{i,1}}{\rightleftarrows}} t_{i,2,5}$$

Notice that, by the arbitrariness of i, we have for every $X \in \{X_0, \ldots, X_{m-1}\} = V(\varphi)$ its unique mirror event $x \in \{x_0, \ldots, x_{m-1}\}$. Moreover, by definition, if $s \xrightarrow{X} s'$ is a an edge (of any forward-backward path) then $s \xleftarrow{x} s'$ is too. The paths $P_{i,0}, P_{i,1}, P_{i,2}$ are similar but the variable events are once and twice left-shifted, respectively. Secondly, we ensure that (sup, sig) also satisfies $sup(t_{i,1,2}) = sup(t_{i,2,2}) = 1$ and $sup(t_{i,1,5}) = sup(t_{i,2,5}) = 0$, that is, sup synchronizes the states $t_{i,0,2}, t_{i,1,2}, t_{i,2,2}$ and the states $t_{i,0,5}, t_{i,1,5}, t_{i,2,5}$, respectively. As a result, we have for every variable event $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ an edge $\xrightarrow{X} s$ such that $sup(s) = 0$. Consequently, we have that $sig(X) \notin \{\mathsf{out}, \mathsf{set}, \mathsf{used}\}$, as $\xrightarrow{i} x$ and $i \in \{\mathsf{out}, \mathsf{set}, \mathsf{used}\}$ requires $x = 1$. Moreover, we also have for every variable event an edge $s \xrightarrow{X}$ such that $sup(s) = 1$, such that $sig(X) \neq \mathsf{free}$. This leaves $\mathsf{nop}, \mathsf{inp}, \mathsf{res}, \mathsf{swap}$ as remaining candidates for $sig(X)$.

As a matter of fact, the construction already ensures that a variable event $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ with $sig(X) \in \{\mathsf{inp}, \mathsf{res}\}$ implies that $sig(Y) = \mathsf{nop}$ for $Y \in \{X_{i,0}, X_{i,1}, X_{i,2}\} \setminus \{Y\}$. We explicitly justify this claim only for $X = X_{i,0}$ as, by symmetry, the cases $X = X_{i,1}$ and $X = X_{i,2}$ are quite similar: By $sig(X_{i,0}) \in \{\mathsf{inp}, \mathsf{res}\}$ we have $sup(t_{i,0,3}) = 0$ which by $sup(t_{i,0,2}) = 1$ and $t_{i,0,2} \xleftarrow{x_{i,0}} t_{i,0,3}$ implies $sig(x_{i,0}) \in \{\mathsf{out}, \mathsf{set}, \mathsf{swap}\}$. Again by $sig(X_{i,0}) \in \{\mathsf{inp}, \mathsf{res}\}$ we have $sup(t_{i,2,4}) = 0$ implying with $sig(x_{i,0}) \in \{\mathsf{out}, \mathsf{set}, \mathsf{swap}\}$ that $sup(t_{i,2,3}) = 1$. By $t_{i,2,2} \xrightarrow{X_{i,2}} t_{i,2,3}$, $sup(t_{i,2,2}) = sup(t_{i,2,3}) = 1$ and $sig(X_{i,2}) \in \{\mathsf{nop}, \mathsf{inp}, \mathsf{res}, \mathsf{swap}\}$ we conclude $sig(X_{i,2}) = \mathsf{nop}$. Furthermore, by $sup(t_{i,1,5}) = 0$ and $sig(x_{i,0}) \in \{\mathsf{out}, \mathsf{set}, \mathsf{swap}\}$ we have $sup(t_{i,1,4}) = 1$. By $sig(X_{i,2}) = \mathsf{nop}$, $t_{i,1,3} \xrightarrow{X_{i,2}} t_{i,1,4}$

and $sup(t_{i,1,4}) = 1$ we obtain that $sup(t_{i,1,3}) = 1$. Finally, by $sup(t_{i,1,2}) = sup(t_{i,1,3}) = 1$, $t_{i,1,2} \xrightarrow{X_{i,1}} t_{i,1,3}$ and $sig(X_{i,1}) \in \{\mathsf{nop}, \mathsf{inp}, \mathsf{res}, \mathsf{swap}\}$ we obtain $sig(X_{i,1}) = \mathsf{nop}$.

So far, we have already argued that for types τ with $\mathsf{swap} \notin \tau$ the following is true: A τ-region that synchronizes $t_{i,0,2}, t_{i,1,2}, t_{i,2,2}$ and $t_{i,0,5}, t_{i,1,5}, t_{i,2,5}$ as announced ensures that there is exactly one variable event of $X_{i,0}, X_{i,1}, X_{i,2}$ with a signature different from nop. This concerns the types which are covered by σ_1.

For the types τ with $\mathsf{swap} \in \tau$, covered by σ_2 and σ_3, it remains to ensure the following: If $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ and $sig(X) = \mathsf{swap}$ then $sig(Y) = \mathsf{nop}$ for $Y \in \{X_{i,0}, X_{i,1}, X_{i,2}\} \setminus \{X\}$. Unfortunately, the previous construction (alone) can not afford this. However, we overcome this problem by another enhancement which we develop in the sequel.

Notice that, by the former discussion, $sig(X) = \mathsf{swap}$ already implies $sig(Y) \notin \{\mathsf{inp}, \mathsf{res}\}$ for $Y \in \{X_{i,0}, X_{i,1}, X_{i,2}\} \setminus \{X\}$. Hence, its simple maths that if $sig(X) = \mathsf{swap}$ then either *all* variable events are mapped to swap or *exactly one* of them: Either we have three changes in τ to get from 1 to 0 or exactly one. Moreover, it is easy to see that if $sig(X) = \mathsf{swap}$ for all $X \in \{X_{i,0}, X_{i,1}, X_{i,2}\}$ then $sig(x) = \mathsf{swap}$ for all $x \in \{x_{i,0}, x_{i,1}, x_{i,2}\}$. Thus, it is sufficient to prevent that *any* x is mapped to swap. To do so, the union $U_\varphi^{\sigma_2}$ installs for every mirror event $x_i \in \{x_0, \ldots, x_{m-1}\}$, that is, especially for $x_{i,0}, x_{i,1}, x_{i,2}$, the gadget TS $G_i^{x,\text{-}}$ which is defined in Figure 5.4. As (sup, sig) satisfies $sig(k) \in \{\mathsf{inp}, \mathsf{out}\}$ we have $sup(g_{i,0}^{x,\text{-}}) = sup(g_{i,1}^{x,\text{-}})$ which implies $sig(x_i) \neq \mathsf{swap}$ for $i \in \{0, \ldots, m-1\}$. Similarly, the union $U_\varphi^{\sigma_3}$ installs for every $i \in \{0, \ldots, m-1\}$ the gadget TS $G_i^{\text{-},x}$ which is defined in Figure 5.8. The reason for these differences between $U_\varphi^{\sigma_2}$ and $U_\varphi^{\sigma_3}$ is again due to theirs different types and the target to satisfy Condition 5 and Condition 6. We will give a detailed explanation later.

Altogether, we have argued that a reduction with these features ensures, that the existence of (sup, sig) implies that $M = \{X \in V(\varphi) \mid sig(X) \neq \mathsf{nop}\}$ is a one-in-three model of φ. Moreover, an analogous argument shows that $sup(t_{i,0,2}) = sup(t_{i,1,2}) = sup(t_{i,2,2}) = 0$ and $sup(t_{i,0,5}) = sup(t_{i,1,5}) = sup(t_{i,2,5}) = 1$ also implies that exactly one variable event (per clause) has a signature different from nop. But *how* can we ensure that the states $t_{i,0,2}, t_{i,1,2}, t_{i,2,2}$ and the states $t_{i,0,5}, t_{i,1,5}, t_{i,2,5}$ become synchronized? To achieve this, we enhance $P_{i,0}, P_{i,1}, P_{i,2}$ as follows: If $\sigma \in \{\sigma_1, \sigma_2\}$ then we enhance $P_{i,0}, P_{i,1}, P_{i,2}$ to $T_{i,0}, T_{i,1}, T_{i,2}$ in accordance to Figure 5.1- Figure 5.3, respectively. $T_{i,0}, T_{i,1}$ and $T_{i,2}$ apply the event k and the events $v_{3i}, v_{3i+1}, v_{3i+2}$ and $w_{3i}, w_{3i+1}, w_{3i+2}$. To make it clear: Besides the corresponding gadgets introduced in Section 3.3, $U_\varphi^{\sigma_1}$ and $U_\varphi^{\sigma_2}$ (additionally) implement the TSs $T_{i,0}, T_{i,1}$ and $T_{i,2}$ for every $i \in \{0, \ldots, m-1\}$. Moreover, $U_\varphi^{\sigma_2}$ implements also $G_i^{x,\text{-}}$ for every $i \in \{0, \ldots, m-1\}$.

The region (sup, sig) uses the latest enhancement as follows: As already discussed in the former section, $sig(k) = \mathsf{inp}$ (and $sup(h_{0,6}) = 0$) imply $V \subseteq sig^{-1}(\mathsf{enter})$ and $W \subseteq sig^{-1}(\mathsf{keep}^-)$. Moreover, by $sig(K) = \mathsf{inp}$, we have $sup(t_{i,0,1}) = sup(t_{i,1,1}) = sup(t_{i,2,1}) = 0$. Altogether, this implies $sup(t_{i,0,2}) = sup(t_{i,1,2}) = sup(t_{i,2,2}) = 1$ and $sup(t_{i,0,5}) = sup(t_{i,1,5}) = sup(t_{i,2,5}) = 0$.

Moreover, if $sig(k) = \mathsf{out}$ then $V \subseteq sig^{-1}(\mathsf{exit})$, implying $sup(t_{i,0,2}) = sup(t_{i,1,2}) = sup(t_{i,2,2}) = 0$, and $W \subseteq sig^{-1}(\mathsf{keep}^+)$, implying $sup(t_{i,0,5}) = sup(t_{i,1,5}) = sup(t_{i,2,5}) = 1$. As discussed, this implies a one-in-three model of φ.

For $U_\varphi^{\sigma_3}$ we need a slightly different construction: The union $U_\varphi^{\sigma_3}$ implements for every $i \in \{0, \dots, m-1\}$ instead of $T_{i,0}, T_{i,1}, T_{i,2}$ the transition system $T'_{i,0}, T'_{i,1}, T'_{i,2}$ defined in Figure 5.4-Figure 5.6, respectively. $T'_{i,0}, T'_{i,1}, T'_{i,2}$ also implement $P_{i,0}, P_{i,1}, P_{i,2}$ but switch the position of $v_{3i}, v_{3i+1}, v_{3i+2}$ with $w_{3i}, w_{3i+1}, w_{3i+2}$, respectively. By symmetry, the arguments that $U_\varphi^{\sigma_3}$ satisfies Condition 4 are similar to the ones for $U_\varphi^{\sigma_1}$ and $U_\varphi^{\sigma_2}$.

Altogether, we have argued that U_φ^{σ} satisfies Condition 1 and Condition 4 for $\sigma \in \{\sigma_1, \sigma_2, \sigma_3\}$. In the next section we talk about the fifth condition. By doing so, we also justify for why $U_\varphi^{\sigma_3}$'s construction different to $U_\varphi^{\sigma_2}$. An essential reason for this is that we have to fulfill both: Condition 4 *and* Condition 5.

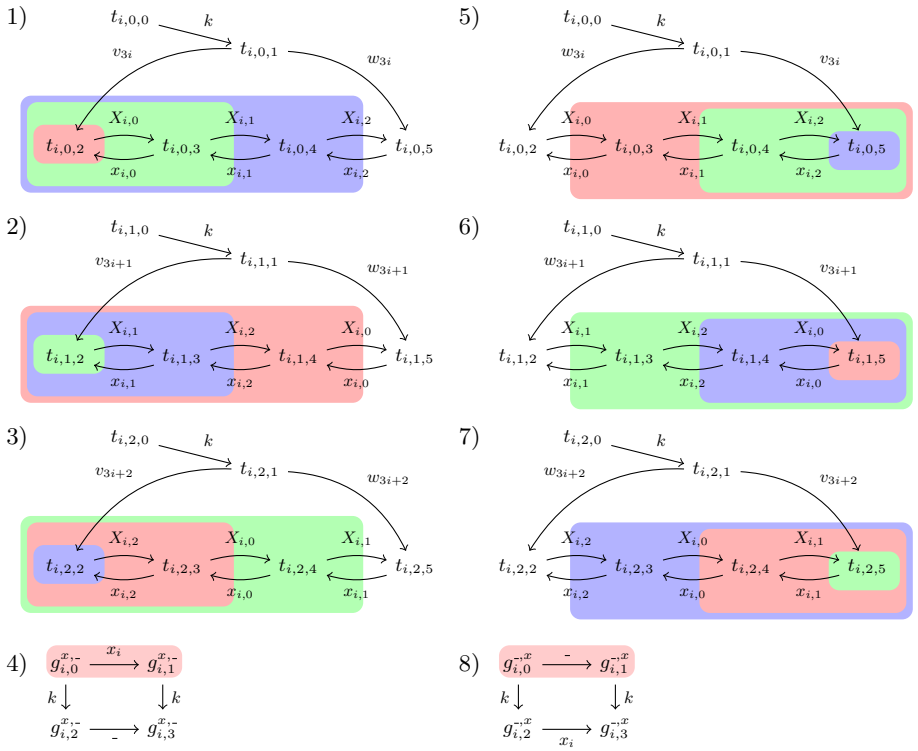


Fig. 5. (1-3) $T_{i,0}, T_{i,1}, T_{i,2}$, (4) $G_i^{x,\text{-}}$, (5-7) $T'_{i,0}, T'_{i,1}, T'_{i,2}$, (8) $G_i^{\text{-},x}$. (1-3,5-7): The colored areas mark the supports of the three possible regions for Condition 5 as described in Section 3.5: red: $X_{i,0} \in M$, green: $X_{i,1} \in M$ and blue: $X_{i,2} \in M$. The (initial) states $t_{i,0,0}, t_{i,1,0}$ and $t_{i,2,0}$ are mapped to 1 for all cases. (4,8): The colored areas define the support of (sup, sig) for Condition 5 as given in Section 3.5.

3.5 Details for Condition 5

For Condition 5, we start from a given one-in-three model M of φ and show that U_φ^σ allows a region (sup, sig) such that $sig(k) = \mathsf{inp}$ and $sup(h_{0,6}) = 0$ and $V \subseteq sig^{-1}(\mathsf{enter})$ and $W \subseteq sig^{-1}(\mathsf{keep}^-)$.

For a start, let's restrict to σ_1, σ_2 and the TSs $T_{i,0}, T_{i,1}T_{i,2}$ implemented by $U_\varphi^{\sigma_1}, U_\varphi^{\sigma_2}$ and $G_i^{x,\text{-}}$ used by $U_\varphi^{\sigma_2}$ where $i \in \{0, \ldots, m-1\}$: We define $sig(k) = \mathsf{inp}$, $sig(v_{3i}), sig(v_{3i+1}), sig(v_{3i+2}) \in \mathsf{enter}$ and $sig(w_{3i}), sig(w_{3i+1}), sig(w_{3i+2}) \in \mathsf{keep}^-$, $i \in \{0, \ldots, m-1\}$. This requires $sup(t_{i,0,2}) = sup(t_{i,1,2}) = sup(t_{i,2,2}) = 1$ and $sup(t_{i,0,5}) = sup(t_{i,1,5}) = sup(t_{i,2,5}) = 0$. That is, for every $T_{i,0}, T_{i,1}, T_{i,2}$ we have a path from 1 to 0 labeled by $X_{i,0}, X_{i,1}, X_{i,2}$ and a path from 0 to 1 labelled by $x_{i,0}, x_{i,1}, x_{i,2}$. For the former path, we define $sig(X) = \mathsf{inp}$ if $X \in M$ and $sig(X) = \mathsf{nop}$ if $Y \in V(\varphi) \setminus M$. Accordingly, for the latter path we let $sig(x) \in \mathsf{enter}$ if $sig(X) = \mathsf{inp}$ and, otherwise $sig(x) = \mathsf{nop}$, cf. Figure 5.1-Figure 5.3: The red area sketches the case $sig(X_{i,0}) = \mathsf{inp}$, the green area $sig(X_{i,1}) = \mathsf{inp}$ and the blue area $sig(X_{i,2}) = \mathsf{inp}$. States within the corresponding colored area are mapped to 1 the others to 0, respectively. If $\sigma = \sigma_1$ then, by Figure 5.1-Figure 5.3 and by recalling that every variable occurs exactly three times in exactly three clauses, it is easy to see that this yields a well defined region. Considering $T_{i,0}, T_{i,1}, T_{i,2}$ alone, this is also true for σ_2. However, for σ_2 we also need to take the TSs $G_0^{x,\text{-}}, \ldots, G_{m-1}^{x,\text{-}}$ into account. Here, by $sig(k) = \mathsf{inp}$ we have that $sup(g_{i,0}^{x,\text{-}}) = sup(g_{i,1}^{x,\text{-}}) = 1$ such that $sig(x_i) \in \mathsf{keep}^+$. Hence, we need that $sig(x_i) \in \mathsf{enter} \cap \mathsf{keep}^+$, that is, $sig(x_i) = \mathsf{set}$. Fortunately, *all* types $\tau \in \sigma_2$ has the property $\mathsf{set} \in \tau$. We argue that (sup, sig) is extendable to $U_\varphi^{\sigma_1}$ and $U_\varphi^{\sigma_2}$ and their other TSs: With the help of the colored areas of the introduced TSs we extend (sup, sig) as follows. For every implemented gadget TS, the red colored area refers to states s such that $sup(s) = 1$, otherwise, $sup(s) = 0$. If $s \xrightarrow{e} s'$ where s is in a red colored area and s' is not, define $sig(e) = \mathsf{inp}$. Similarly, if s' is in a red colored area and s is not, define $sig(e) \in \mathsf{enter}$. Finally, if either both of s, s' are red or both not then define $sig(e) = \mathsf{nop}$. One easily verifies that this yields a well defined region such such that $U_\varphi^{\sigma_1}, U_\varphi^{\sigma_2}$ satisfy Condition 5.

Unfortunately, as $\mathsf{set} \notin \tau$ for $\tau \in \sigma_3$ this region *can not* exist for σ_3 which is one reason why σ_3 needs another construction. For $U_\varphi^{\sigma_3}$ we obtain a corresponding region as follows: We define $sig(k) = \mathsf{inp}$ and $V \subseteq sig^{-1}(\mathsf{enter})$ and $W \subseteq sig^{-1}(\mathsf{keep}^-)$ which requires $sup(t_{i,0,2}) = sup(t_{i,1,2}) = sup(t_{i,2,2}) = 0$ and $sup(t_{i,0,5}) = sup(t_{i,1,5}) = sup(t_{i,2,5}) = 1$. That is, for every $T'_{i,0}, T'_{i,1}, T'_{i,2}$ we have a path from 0 to 1 which is labeled by $X_{i,0}, X_{i,1}, X_{i,2}$ and a path from 1 to 0 labelled by $x_{i,0}, x_{i,1}, x_{i,2}$. For the former path, we define $sig(X) = \mathsf{swap}$ if $X \in M$ and $sig(X) = \mathsf{nop}$ if $Y \in V(\varphi) \setminus M$, cf. Figure 5.4-Figure 5.7. Accordingly, for the latter path we let $sig(x) = \mathsf{res}$ if $sig(X) = \mathsf{swap}$ and, otherwise $sig(x) = \mathsf{nop}$. We take the TSs $G_0^{\text{-},x}, \ldots, G_{m-1}^{\text{-},x}$ into account and, by $sig(k) = \mathsf{inp}$ we have $sup(g_{i,2}^{\text{-},x}) = sup(g_{i,3}^{\text{-},x}) = 0$ which is compatible with $sig(x_i) = \mathsf{res}$. The extension of (sup, sig) to the remaining gadgets of $U_\varphi^{\sigma_3}$ works analogously to $U_\varphi^{\sigma_2}$, where border-crossing events from 0 to 1 are mapped to swap (instead of set). Altogether, we have proven that U_φ^σ satisfies Condition 5 for $\sigma \in \{\sigma_1, \sigma_2, \sigma_3\}$.

4 Concluding Remarks

In this paper we present a proof scheme to show the NP-hardness of synthesis, for 68 boolean Petri net types. Together with our previous work from [14], this already makes 111 out of 256 boolean net types where the complexity of synthesis has been figured out. In fact, with respect to the practically more relevant nop-afflicted types of nets, there are only 17 cases left open. However, in view of our main target (dichotomy result) it remains for future work to characterize the synthesis complexity for all the remaining 145 types.

The NP-hardness results of the investigated synthesis problems motivates the search for (fixed-parameter) tractable special cases. There are at least two ways to (possibly) obtain such cases: Firstly, one can put (structural) restrictions on the τ-net N to be synthesized (the output) and require, for example, that N has to be *free-choice* or a *marked graph* [4]. To investigate the impact of such output-restrictions on the complexity of boolean net synthesis is certainly an interesting direction for further research.

Secondly, one can put restrictions on the TS from which N is to synthesize (the input). One of the most obvious parameters here is the TS's *state degree* g, that is, the maximum number of ingoing and outgoing edges at a state. Actually, TSs of benchmarks of the digital hardware design community often have a low state degree [5]. However, our result show that it is very unlikely that feasibility is (fixed-parameter) tractable for parameter g: All gadget TSs have at most two ingoing and at most two outgoing edges per state, that is $g \leq 2$, and this property is preserved by the joining-operation for $A(U_\varphi^\sigma)$. Clearly, this leaves us with the question if synthesis becomes tractable if $g \leq 1$. On the one hand, that this can not generally be true has been shown in [15]. On the other hand, one observes that our reduction's functionality heavily bases on the ability to prevent a res-, set- and swap-signature of certain events. The core gadgets here are $T_{i,0}, T_{i,1}, T_{i,2}$ (implementing the paths $P_{i,0}, P_{i,1}, P_{i,2}$) and the TSs $G_i^{x,\text{-}}$ and $G_i^{\text{-},x}$. These gadgets are reliant on the possibility to have two ingoing and outgoing edges per state. Hence, the case $g \leq 1$ is an object worth to study in future work, at least for the discussed types satisfying $\tau \cap \{\mathsf{set}, \mathsf{res}, \mathsf{swap}\} \neq \emptyset$.

References

1. Badouel, E., Bernardinello, L., Darondeau, P.: The synthesis problem for elementary net systems is np-complete. Theor. Comput. Sci. **186**(1-2), 107–134 (1997). https://doi.org/10.1016/S0304-3975(96)00219-8
2. Badouel, E., Bernardinello, L., Darondeau, P.: Petri Net Synthesis. Texts in Theoretical Computer Science. An EATCS Series, Springer (2015). https://doi.org/10.1007/978-3-662-47967-4
3. Badouel, E., Darondeau, P.: Trace nets and process automata. Acta Inf. **32**(7), 647–679 (1995). https://doi.org/10.1007/BF01186645
4. Best, E.: Structure theory of petri nets: the free choice hiatus. In: Advances in Petri Nets. Lecture Notes in Computer Science, vol. 254, pp. 168–205. Springer (1986). https://doi.org/10.1007/BFb0046840

5. Cortadella, J.: Private correspondance (2017)
6. Kleijn, J., Koutny, M., Pietkiewicz-Koutny, M., Rozenberg, G.: Step semantics of boolean nets. Acta Inf. **50**(1), 15–39 (2013). https://doi.org/10.1007/s00236-012-0170-2
7. Montanari, U., Rossi, F.: Contextual nets. Acta Inf. **32**(6), 545–596 (1995). https://doi.org/10.1007/BF01178907
8. Moore, C., Robson, J.M.: Hard tiling problems with simple tiles. Discrete & Computational Geometry **26**(4), 573–590 (2001). https://doi.org/10.1007/s00454-001-0047-6
9. Pietkiewicz-Koutny, M.: Transition systems of elementary net systems with inhibitor arcs. In: ICATPN. Lecture Notes in Computer Science, vol. 1248, pp. 310–327. Springer (1997). https://doi.org/10.1007/3-540-63139-9_43
10. Rozenberg, G., Engelfriet, J.: Elementary net systems. In: Petri Nets. Lecture Notes in Computer Science, vol. 1491, pp. 12–121. Springer (1996). https://doi.org/10.1007/3-540-65306-6_14
11. Schmitt, V.: Flip-flop nets. In: STACS. Lecture Notes in Computer Science, vol. 1046, pp. 517–528. Springer (1996). https://doi.org/10.1007/3-540-60922-9_42
12. Tredup, R., Rosenke, C.: Narrowing down the hardness barrier of synthesizing elementary net systems. In: CONCUR. LIPIcs, vol. 118, pp. 16:1–16:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018). https://doi.org/10.4230/LIPIcs.CONCUR.2018.16
13. Tredup, R., Rosenke, C.: Towards completely characterizing the complexity of boolean nets synthesis. CoRR **abs/1806.03703** (2018), http://arxiv.org/abs/1806.03703
14. Tredup, R., Rosenke, C.: The complexity of synthesis for 43 boolean petri net types. In: Theory and Applications of Models of Computation. Theoretical Computer Science and General Issues, vol. 11436. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-14812-6
15. Tredup, R., Rosenke, C., Wolf, K.: Elementary net synthesis remains np-complete even for extremely simple inputs. In: Petri Nets. Lecture Notes in Computer Science, vol. 10877, pp. 40–59. Springer (2018). https://doi.org/10.1007/978-3-319-91268-4_3