313

arXiv:2311.11443v4 [cs.LO] 11 Sep 2024

Taking Complete Finite Prefixes To High Level, Symbolically[*]

Nick Würdemann[*]

Department of Computing Science,
University of Oldenburg, Oldenburg, Germany
wuerdemann@informatik.uni-oldenburg.de

Stefan Haar

Université Paris-Saclay, INRIA and LMF, CNRS and
ENS Paris-Saclay, Gif-sur-Yvette, France
stefan.haar@inria.fr

Thomas Chatain

Université Paris-Saclay, INRIA and LMF, CNRS and
ENS Paris-Saclay, Gif-sur-Yvette, France
thomas.chatain@inria.fr

Lukas Panneke

Department of Computing Science,
University of Oldenburg, Oldenburg, Germany
lukas.panneke@informatik.uni-oldenburg.de

Abstract. Unfoldings are a well known partial-order semantics of P/T Petri nets that can be applied to various model checking or verification problems. For *high-level* Petri nets, the so-called *symbolic* unfolding generalizes this notion. A complete finite prefix of a P/T Petri net's unfolding contains all information to verify, e.g., reachability of markings. We unite these two concepts and define complete finite prefixes of the symbolic unfolding of high-level Petri nets. For a class of safe high-level Petri nets, we generalize the well-known algorithm by Esparza et al. for constructing small such prefixes. We evaluate this extended algorithm through a prototype implementation on four novel benchmark families. Additionally, we identify a more general class of nets with infinitely many reachable markings, for which an approach with an adapted cut-off criterion extends the complete prefix methodology, in the sense that the original algorithm cannot be applied to the P/T net represented by a high-level net.

Keywords: Petri Nets, High-level Petri Nets, Unfoldings, Concurrency Theory

---

Introduction

Petri nets [2], also called P/T (for Place/Transition) Petri nets or low-level Petri nets, are a well-established formalism for describing distributed systems. *High-level Petri nets* [3] (also called *colored Petri nets*) are a concise representation of P/T Petri nets, allowing the places to carry tokens of different colors. Every high-level Petri net represents a P/T Petri net, here called its *expansion*[1], where the process of constructing this P/T net is called *expanding* the high-level net.

Unfoldings of P/T Petri nets are introduced by Nielsen et al. in [5]. Engelfriet generalizes this concept in [6] by introducing the notion of *branching processes*, and shows that the unfolding of a net is its maximal branching process. 0n [7], McMillan gives an algorithm to compute a complete finite prefix of the unfolding of a given Petri net. In a well-known paper [8], Esparza, Römer, and Vogler improve this algorithm by defining and exploiting a total order on the set of configurations in the unfolding. We call the improved algorithm the "ERV-algorithm". It leads to a comparably small complete finite prefix of the unfolding. In [9], Khomenko and Koutny describe how to construct the unfolding of the expansion of a high-level Petri net without first expanding it.

High-level representations on the one hand and processes (resp. unfoldings) of P/T Petri nets on the other, at first glance seem to be conflicting concepts; one being a more concise, the other a more detailed description of the net('s behavior). However, in [10], Ehrig et al. define processes of high-level Petri nets, and in [11], Chatain and Jard define *symbolic branching processes* and *unfoldings* of high-level Petri nets. The work on the latter is built upon in [4] by Chatain and Fabre, where they consider so-called "puzzle nets". Based on the construction of a symbolic unfolding, in [12], complete finite prefixes of safe time Petri nets are constructed, using time constraints associated with timed processes. In [13], using a simple example, Chatain argues that in general there exists no complete finite prefix of the symbolic unfolding of a high-level Petri net. However, this is only true for high-level Petri nets with infinitely many reachable markings such that the number of steps needed to reach them is unbounded, in which case the same arguments work for P/T Petri nets.

In this paper, we lift the concepts of complete prefixes and adequate orders to the level of symbolic unfoldings of high-level Petri nets. We consider the class of *safe* high-level Petri nets (i.e., in all reachable markings, every place carries at most one token) that have decidable guards and finitely many reachable markings. This class generalizes safe P/T Petri nets, and we obtain a generalized version of the ERV-algorithm creating a complete finite prefix of the symbolic unfolding of such a given high-level Petri net. Our results are a generalization of [8] in the sense that if a P/T Petri net is viewed as a high-level Petri net, the new definitions of adequate orders and completeness of prefixes on the symbolic level, as well as the algorithm producing them, all coincide with their P/T counterparts.

We proceed to identify an even more general class of so-called *symbolically compact* high-level Petri nets; we drop the assumption of finitely many reachable markings, and instead assume the existence of a bound on the number of steps needed to reach all reachable markings. In such a case, the expansion is possibly not finite, and the original ERV-algorithm from [8] therefore not applicable. We adapt the generalized ERV-algorithm by weakening the cut-off criterion to ensure finiteness of the resulting prefix. Still, in this cut-off criterion we have to compare infinite sets of markings. We

---

[1]The Petri net being represented is commonly referred to as the *unfolding* of the high-level Petri net in the literature. To prevent any potential confusion, we opt for the term *expansion*, as, for instance, in [4].

overcome this obstacle by symbolically representing these sets, using the decidability of the guards to decide cut-offs. Finally, we present four new benchmark families for which we report on the results of applying a prototype implementation of the generalized ERV-algorithm.

Distinctions from the Conference Version

This extended version incorporates numerous textual enhancements compared to our original work in [1]. Apart from that, we made the following changes and additions:

- The proofs that were excluded in the conference version have now been integrated into the main body of the paper.

- We substituted the running example with a more intricate and compelling one (Fig. 1 in Sec. 1), and discuss it in greater detail. Additionally, we present an example for the central concept "color conflict".

- Sec. 3 has been completely revised.

- We introduced a new subsection, found in Sec. 4.2, where we demonstrate that the generalized ERV-algorithm may not terminate when applied to input nets from $\mathbf{N_{SC}}$. This further motivates the work from the conference version of finding a new cut-off criterion (Sec. 4.3). In another new subsection, found in Sec. 4.4 we discuss the feasibility of symbolically compact nets and provide an outlook into the potential development of a symbolic reachability graph.

- We changed the definition of $pred^{\odot}$ in Sec. 5 (formerly Section 4.3). This allows for a better presentation of Theorem 5.3, and an easier proof.

- In a new section, found in Sec. 6, we report in Sec. 6.1 on a new prototype implementation of the generalized ERV-algorithm from Sec. 2.2. In Sec. 6.2 we present four new benchmark families of high-level Petri nets. In Sec. 6.3 we discuss a property of high-level Petri nets which we call *mode-determinism*, leading to an indicator for whether the symbolic unfolding is expected to be faster to construct than the low-level unfolding. In Sec. 6.4, we present the results of applying the implementation to the benchmarks from Sec. 6.2.

1. High-level Petri nets & symbolic unfoldings

In [11], symbolic unfoldings for high-level Petri nets are introduced. In Sections 1.1 and 1.2, we recall the definitions and formalism for high-level Petri nets and symbolic unfoldings from [11].

Multi-sets. For a set X, we call a function $A : X \to \mathbb{N}$ a *multi-set over X*. We denote $x \in A$ if $A(x) \geq 1$. For two multi-sets A, A' over the same set X, we write $A \leq A'$ iff $\forall x \in X : A(x) \leq A'(x)$, and denote by $A + A'$ and $A - A'$ the multi-sets over X given by $(A + A')(x) = A(x) + A'(x)$ and $(A - A')(x) = \max(A(x) - A'(x), 0)$. We use the notation $\{\!| \ldots |\!\}$ as introduced in [9]: elements in a multi-set can be listed explicitly as in $\{\!| x_1, x_1, x_2 |\!\}$, which describes the multi-set A with $A(x_1) = 2$, $A(x_2) = 1$, and $A(x) = 0$ for all $x \in X \setminus \{x_1, x_2\}$. A multi-set A is finite if there are finitely many $x \in X$ such that $x \in A$. In such a case, $\{\!| f(x) \mid x \in A |\!\}$, with $f(x)$ being an object constructed from $x \in X$, denotes the multi-set A' such that $A' = \sum_{x \in X} A(x) \cdot f(x)$, where the $A(x) \cdot y$ is the multi-set containing exactly $A(x)$ copies of y.

1.1. High-level Petri nets

A *(high-level) net structure* is a tuple $\mathcal{N} = (Col, Var, P, T, F, \iota)$ with the following components: Col and Var are the sets of *colors* and *variables*, and P and T are sets of *places* and *transitions* such that the four sets are pairwise disjoint. The *flow function* is given by $F : (P \times Var \times T) \cup (T \times Var \times P) \to \mathbb{N}$. For $t \in T$, let $Var(t) = \{v \in Var \mid \exists p \in P : (p, v, t) \in F \lor (t, v, p) \in F\}$. The function ι maps each $t \in T$ to a predicate $\iota(t)$ on $Var(t)$, called the *guard* of t. By this, $\iota(t)$ can contain other (bounded) variables, but all free variables in $\iota(t)$ must appear on arcs to or from t. A *marking* in $\mathcal{N}$ is a multi-set M over $P \times Col$, describing how often each color $c \in Col$ currently resides on each place $p \in P$. A *high-level Petri net* $N = (\mathcal{N}, \mathcal{M}_0)$ is a net structure $\mathcal{N}$ together with a nonempty set $\mathcal{M}_0$ of *initial markings*, where we assume $\forall M_0, M_0' \in \mathcal{M}_0 : \{\!| p \mid (p, c) \in M_0 |\!\} = \{\!| p \mid (p, c) \in M_0' |\!\}$, i.e., in all initial markings, the same places are marked with *the same number of colors*. We often assume the two sets Col of colors and Var of variables to be fixed. In this case, we denote a high-level net structure (resp. high-level Petri net) by $\mathcal{N} = (P, T, F, \iota)$ (resp. $N = (P, T, F, \iota, \mathcal{M}_0)$).

For two nodes $x, y \in P \cup T$, we write $x \to y$, if there exists a variable v such that $(x, v, y) \in F$. The reflexive and irreflexive transitive closures of $\to$ are denoted respectively by $\le$ and $<$. For a transition $t \in T$, we denote by $pre(t) = \{\!| (p, v) \mid (p, v, t) \in F |\!\}$ and $post(t) = \{\!| (p, v) \mid (t, v, p) \in F |\!\}$ the *preset* and *postset* of t. A *firing mode* of t is a mapping $\sigma : Var(t) \to Col$ such that $\iota(t)$ evaluates to *true* under the substitution given by σ, denoted by $\iota(t)[\sigma] \equiv true$. We then denote $pre(t, \sigma) = \{\!| (p, \sigma(v)) \mid (p, v) \in pre(t) |\!\}$ and $post(t, \sigma) = \{\!| (p, \sigma(v)) \mid (p, v) \in post(t) |\!\}$. The set of modes of t is denoted by $\Sigma(t)$. Note that such a "binding" of variables to colors is always only local, when firing the respective transition. t can *fire* in such a mode σ from a marking M if $M \ge pre(t, \sigma)$, denoted by $M[t, \sigma\rangle$. This firing leads to a new marking $M' = (M - pre(t, \sigma)) + post(t, \sigma)$, which is denoted by $M[t, \sigma\rangle M'$. We collect in the set $\mathcal{R}(\mathcal{N}, \mathcal{M})$ the markings reachable by firing a sequence of transitions in $\mathcal{N}$ from any marking in a set of markings $\mathcal{M}$. We say $\mathcal{N}$ resp. N is *finite* if P, T and F are finite. In this paper, we in particular aim to analyze the *behavior* of high-level Petri nets. To avoid any issues concerning undecidability regarding the firing relation, we assume that guards are expressed in a decidable logic, with Col as its domain of discourse.

Let $\mathcal{N} = (P, T, F, \iota)$ and $\mathcal{N}' = (P', T', F', \iota')$ be two net structures with the same sets of colors and variables. A function $h : P \cup T \to P' \cup T'$ is called a *(high-level Petri net) homomorphism*, if:

i) it maps places and transitions in $\mathcal{N}$ into the corresponding sets in $\mathcal{N}'$, i.e.,
$h(P) \subseteq P'$ and $h(T) \subseteq T'$;

ii) it is "compatible" with the guard, preset, and postset, of transitions, i.e.,
for all $t \in T$ we have $\iota(t) = \iota'(h(t))$ and $pre(h(t)) = \{\!| (h(p), v) \mid (p, v) \in pre(t) |\!\}$ and $post(h(t)) = \{\!| (h(p), v) \mid (p, v) \in post(t) |\!\}$.

For $N = (\mathcal{N}, \mathcal{M}_0)$ and $N' = (\mathcal{N}', \mathcal{M}_0')$, the homomorphisms between N and N' are the homomorphisms between $\mathcal{N}$ and $\mathcal{N}'$. Such a homomorphism h is called *initial* if additionally $\{\{\!| (h(p), c) \mid (p, c) \in M_0 |\!\} \mid M_0 \in \mathcal{M}_0\} = \mathcal{M}_0'$ holds, i.e., the initial markings in N are mapped to the initial markings in N'.

We define *P/T Petri nets* as high-level Petri nets with singletons $Col = \{\bullet\}$ and $Var = \{v_\bullet\}$ for colors and variables, i.e., in a marking, every place holds a number of tokens $\bullet$, which is the only value ever assigned to the variable $v_\bullet$ on every arc. The guard of every transition in a P/T Petri net is *true*.

(a) A safe high-level Petri net N.

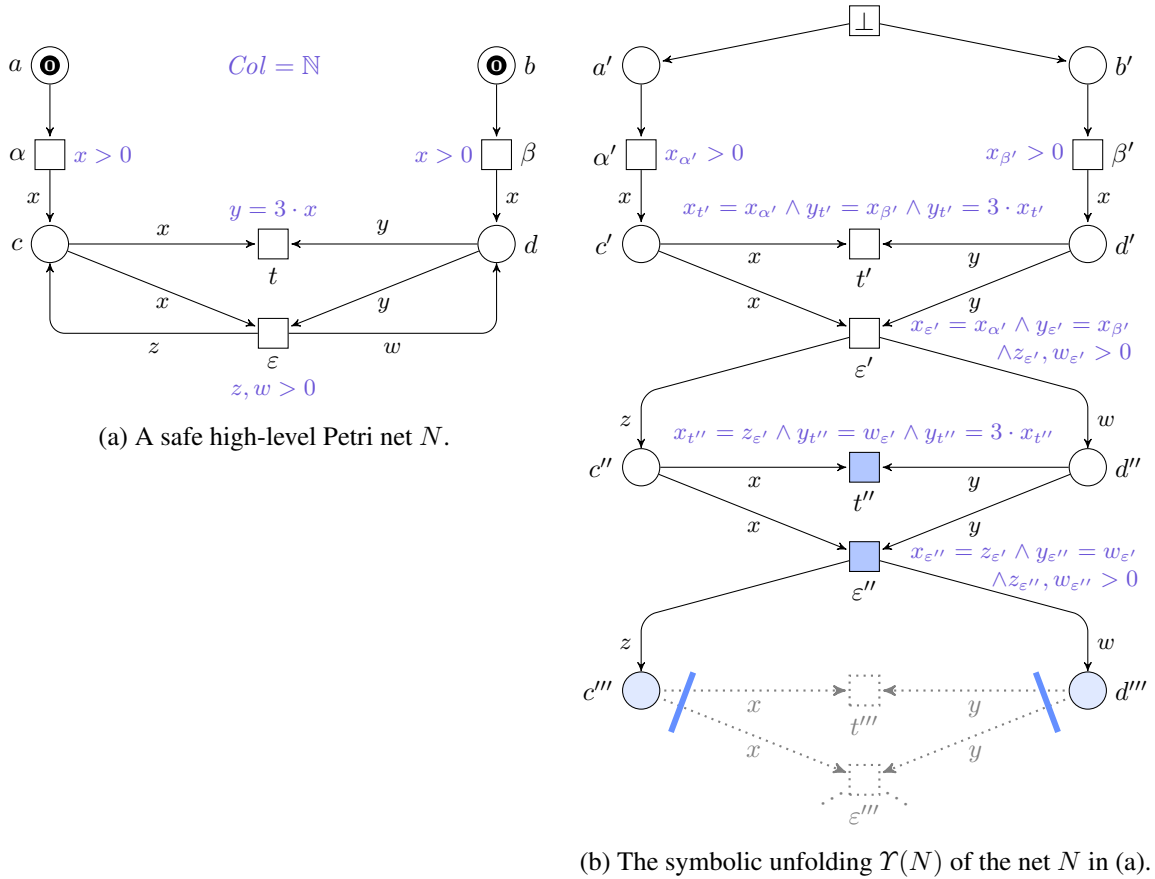(b) The symbolic unfolding $\Upsilon(N)$ of the net N in (a).

Figure 1: A safe high-level Petri net N in (a), discussed in Example 1.1, and (a prefix of) the infinite symbolic unfolding $U(N)$ in (b), discussed in Example 1.3.

Example 1.1. Let $Col = \{0, \ldots, m\}$ for a fixed m, and $Var = \{x, y, z, w\}$ be the given sets of colors and variables. In Figure 1a, the running example N of a high-level Petri net is depicted. Places are drawn as circles, and transitions as squares. The flow is described by labeled arrows, and the guards are written next to the respective transition. The set of initial markings is a singleton $\mathcal{M}_0 = \{M_0\}$ with $M_0 = \{\!| (a, 0), (b, 0) |\!\}$, which is depicted in the net. In all our examples we view 0 as a "special" color, in the sense that we employ unlabeled arcs as an abbreviation for arcs labeled with an additional variable x_0, and the guard of the respective transition having an additional term $x_0 = 0$ in its guard. Thus, we handle 0 as we would the token $\bullet$ in the P/T case.

From M_0, both α and β can fire, taking the color 0 from place a resp. b and placing a color $k \in \{1, \ldots, m\}$ on place c resp. d when firing in mode $\{x \leftarrow k\}$. The mode $\{x \leftarrow 0\}$ is for both transitions excluded by their respective guard. When both α and β fire, the net arrives at a marking $\{\!|\, (c,k), (d,\ell)\, |\!\}$. From there, ε can fire arbitrarily often, always replacing the colors k, ℓ currently residing on c, d by any colors $0 < k', \ell' \leq m$ by firing in mode $\{x \leftarrow k, y \leftarrow \ell, z \leftarrow k', w \leftarrow \ell'\}$. From every marking $\{\!|\, (c,k), (d,\ell)\, |\!\}$ satisfying $\ell = 3 \cdot k$, transition t can fire, ending the execution.

1.2.   Symbolic branching processes and unfoldings

A high-level net structure $\mathcal{N} = (Col, Var, P, T, F, \iota)$ is called *ordinary* if there is at most one arc connecting any two nodes in $\mathcal{N}$, i.e., $\forall (x, y) \in (P \times T) \cup (T \times P) : \sum_{v \in Var} F(x, v, y) \leq 1$. For such an ordinary net structure, analogously to the well-known low-level case, two nodes $x, y \in P \cup T$ are in *structural conflict*, denoted by $x \sharp y$, if $\exists p \in P \, \exists t, t' \in T : t \neq t' \wedge p \to t \wedge p \to t' \wedge t \leq x \wedge t' \leq y$.

A *high-level occurrence net* (defined below) is a high-level Petri net with an ordinary net structure that satisfies certain properties. In such a net, we call the places *conditions* and denote them by B. Transition are called *events* (denoted by E), and reachable markings are called *cuts*, where the set of initial cuts is denoted by $\mathcal{K}_0$. The flow relation is denoted by H.

The properties *i) – iii)* in the definition below are exactly the same as in the low-level case and concern solely the net structure. Property *iv)* generalizes the corresponding requirement of low-level occurrence nets to the current situation, in which, just as in the low-level case, every condition has at most one event in its preset, and that those conditions having an empty preset constitute the initial cut. Case *iv.a)* describes the conditions that initially hold a color, at the "top" of the net. Case *iv.b)* on the other hand describes the conditions "deeper" in the net, which initially do not hold a color.

Definition 1.2. (High-level occurrence net [11])
A *high-level occurrence net* is a high-level Petri net $O = (Col, Var, B, E, H, \iota, \mathcal{K}_0)$ with an ordinary net structure $(Col, Var, B, E, H, \iota)$ such that

 i) No event is in structural self-conflict, i.e., $\forall e \in E : \neg(e \sharp e)$.

 ii) No node is its own causal predecessor, i.e., $\forall x \in B \cup E : \neg(x < x)$.

 iii) The relation $<$ is well-founded, i.e., contains no infinite decreasing sequence.

 iv) For every $b \in B$, exactly one of the following holds:

    a) $\forall K_0 \in \mathcal{K}_0 : \sum_{c \in Col} K_0(b, c) = 1$ and $\{e \mid e \to b\} = \emptyset$.
    In this case we denote $pre(b) = (\bot, v^b)$.
    b) $\forall K_0 \in \mathcal{K}_0 : \sum_{c \in Col} K_0(b, c) = 0$ and there exists a unique pair (e, v) s.t. $(e, v, b) \in H$.
    In this case we denote $pre(b) = (e, v)$

We denote by $B_0 = \{b \in B \mid \exists K_0 \in \mathcal{K}_0 \, \exists c \in Col : (b, c) \in K_0\}$ the conditions from *iv.a)* occupied in all initial cuts. $\bot$ can be seen as a "special event" that fires only once to initialize the net, and produces the initial cuts $K_0 \in \mathcal{K}_0$ by assigning values to the variables v^b on "special arcs" $(\bot, v^b, b)$ towards the conditions $b \in B_0$.

In a crucial notation for this paper, we define in case *iv.a)* $\mathbf{e}(b) = \bot$, and $\mathbf{v}(b) = v^b$, and in case *iv.b)* we identify the event e by $\mathbf{e}(b)$ and the variable v by $\mathbf{v}(b)$. By this notation, $\forall b \in B$: $pre(b) = (\mathbf{e}(b), \mathbf{v}(b))$. We can say that whenever a condition b holds a color c, then it got placed there by firing $\mathbf{e}(b)$ in a mode that binds $\mathbf{v}(b)$ to the color c.

In a high-level occurrence net, we define for every event e the predicates $loc\text{-}pred(e)$ and $pred(e)$. The predicate $pred(e)$ is satisfiable iff e is not dead, i.e., there are cuts $K_0, \ldots, K_n$ with $K_0 \in \mathcal{K}_0$ and events $e_1, \ldots, e_n$, s.t. $K_0[e_1\rangle \ldots [e_n\rangle K_n[e\rangle$. This predicate is obtained by building a conjunction over all *local predicates* of events e' with $e' \le e$, and the predicate of the special event $\bot$.

The local predicate of e is, in its turn, a conjunction of two predicates expressing that (i) the guard of the event e is satisfied, and (ii) that for any $(b, v) \in pre(e)$, the value of the variable v coincides with the color that the event $\mathbf{e}(b)$ placed in b. To realize this, the variables $v \in Var(e)$ are instantiated by the index e, so that v_e describes the value assigned to v by a mode of e. Having the definition of $\mathbf{e}(b)$ and $\mathbf{v}(b)$ from above in mind, for a condition b, we abbreviate $\mathbf{v_e}(b) = \mathbf{v}(b)_{\mathbf{e}(b)}$. Formally, we have

$$loc\text{-}pred(e) \quad = \quad \iota(e)[v \leftarrow v_e]_{v \in Var(e)} \quad \wedge \bigwedge_{(b,v) \in pre(e)} v_e = \mathbf{v_e}(b)$$

$$pred(e) \quad = \quad pred(\bot) \wedge \bigwedge_{e' \le e} loc\text{-}pred(e'),$$

where $pred(\bot) = \bigvee_{K_0 \in \mathcal{K}_0} \bigwedge_{(b,c) \in K_0} (v_\bot^b = c)$ symbolically represents the set of initial cuts.

A *symbolic branching process* of a high-level Petri net N is a pair $\beta = (O, h)$ with an occurrence net $O = (Col, Var, B, E, H, \iota, \mathcal{K}_0)$ in which $pred(e)$ is satisfiable for all $e \in E$, and an initial homomorphism $h : O \to N$ that is injective on events with the same preset, i.e., $\forall e, e' \in E$: $(pre(e) = pre(e') \wedge h(e) = h(e')) \Rightarrow e = e'$.

For two symbolic branching processes $\beta = (O, h)$ and $\beta' = (O', h')$ of a high-level Petri net, β is a *prefix* of β' if there exists an injective initial homomorphism ϕ from O into O', such that $h' \circ \phi = h$. In [11] it is stated that for any given high-level Petri net N there exists a unique maximal branching process (maximal w.r.t. the prefix relation and unique up to isomorphism). This branching process is called the *symbolic unfolding*, and denoted by $\Upsilon(N) = (U(N), \pi^N)$. The value $\pi^N(x)$ is called the *label* of a node x in $U(N)$.

Example 1.3. Consider again the high-level Petri net N from Figure 1a. In Figure 1b we see (a prefix of) the infinite occurrence net $U(N)$ of the symbolic unfolding $\Upsilon(N)$. We depict the prefix with two instances of each t and ε. Each node in the unfolding is named after the represented place resp. transition (i.e., its label), equipped with a superscript. We include the "special event" $\bot$, that can only fire once, in the drawing. The guards of events are omitted, since they have the same guards as their label. Instead, the local predicate of each event is written next to it.

The local predicate of α', namely $x_{\alpha'} > 0$ expresses that the assignment of colors to variables by a mode of α' must satisfy the constraint given by the guard of its label α. Analogously for β'. The same is expressed in the local predicate of t' by $y_{t'} = 3 \cdot x_{t'}$, coming from the guard $y = 3 \cdot x$ of $\pi^N(t') = t$. Additionally, the first part of the conjunction formalizes that, since $(c', x) \in pre(t')$, the

value that a mode of t' assigns to x must be the same that a mode of $\mathbf{e}(c') = \alpha'$ assigned to $\mathbf{v}(c') = x$. This is expressed as $x_{t'} = x_{\alpha'}$. The second part of the conjunction formalizes the same for y and d'. The whole predicate of t' is then given by

$$pred(t') = x_{\alpha'} > 0 \wedge x_{\beta'} > 0 \wedge x_{t'} = x_{\alpha'} \wedge y_{t'} = x_{\beta'} \wedge y_{t'} = 3 \cdot x_{t'}.$$

Since it is satisfiable for example by $\{x_{\alpha'} \leftarrow 1, x_{\beta'} \leftarrow 3, x_{t'} \leftarrow 1, y_{t'} \leftarrow 3\}$ (meaning that t can fire in mode $\{x \leftarrow 1, y \leftarrow 3\}$ after α fired in mode $\{x \leftarrow 1\}$ and β fired in mode $\{x \leftarrow 3\}$), the node t' is not dead and an event in the unfolding.

The blue shading of event ε'' and t'' indicates that they are what we later term *cut-off events*, which leads to the *complete finite prefix* being marked by the blue thick lines being obtained by Alg. 1, as described later. The unfolding itself is infinite.

As we see in the definition of high-level occurrence nets, the notion of causality and structural conflict are the same as in the low-level case. However, a set of events in an occurrence net can also be in what we call *color conflict*, meaning that the conjunction of their predicates is not satisfiable. In a symbolic branching process, this means that the constraints on the values of the firing modes, coming from the guards of the transitions, prevent joint occurrence of all events from such a set in any *one* run of the net:

The nodes in a set $X \subseteq E \cup B$ are in *color conflict* if

$$\bigwedge_{e \in X \cap E} pred(e) \wedge \bigwedge_{b \in X \cap B} pred(\mathbf{e}(b))$$

is *not* satisfiable. The nodes of X are *concurrent* if they are *not* in color conflict, and for each $x, x' \in X'$, neither $x < x'$, nor $x' < x$, nor $x \sharp x'$ holds. A set of concurrent conditions is called a *co-set*.

Note that while a set of nodes is defined to be in structural conflict if and only if two nodes in it are in structural conflict, the same does not hold for color conflict: it is possible to have a set $\{x_1, x_2, x_3\}$ of nodes that are in color conflict, but for which every subset of cardinality 2 is *not* in color conflict. We demonstrate this on an example.

Example 1.4. (Color conflict)
In Figure 2a, a high-level Petri net with initial marking $\{\!| (p_0, 0) |\!\}$ is depicted. The only enabled transition is t_0, placing the same color $\sigma(x) \in \mathbb{Z} = \{\ldots, -1, 0, 1, 2, \ldots\}$ on each of the three places p_1, p_2, p_3 when fired in mode $\sigma = \{x \leftarrow \sigma(x)\}$. From each of these places, the color $\sigma(x)$ may be taken by a respective transition. The three transitions t_1, t_2, t_3, however, each have a guard: $\iota(t_1) = (x \leq 0)$, $\iota(t_2) = (x \neq 0)$, and $\iota(t_3) = (x \geq 0)$. Depending on the mode σ in which t_0 fired, always two of the three transitions are fireable: if $\sigma(x) = 0$ then t_1 and t_3 can both fire (in mode $\{x \leftarrow 0\}$), if $\sigma(x) < 0$ then t_1 and t_2 can fire, and if $\sigma(x) > 0$ then t_2 and t_3 can fire.

Since the high-level Petri net in Figure 2a is a high-level occurrence net and all predicates are satisfiable, it has the same structure as its own symbolic unfolding in Figure 2b. The set $\{b_1, b_2, b_3\}$ is a co-set, since the conditions are neither in conflict, nor causally related, and $\bigwedge_{b \in \{b_1, b_2, b_3\}} pred(\mathbf{e}(b))$, which is equivalent to $true$, is satisfiable, i.e., the conditions are *not* in color conflict. Consequently,
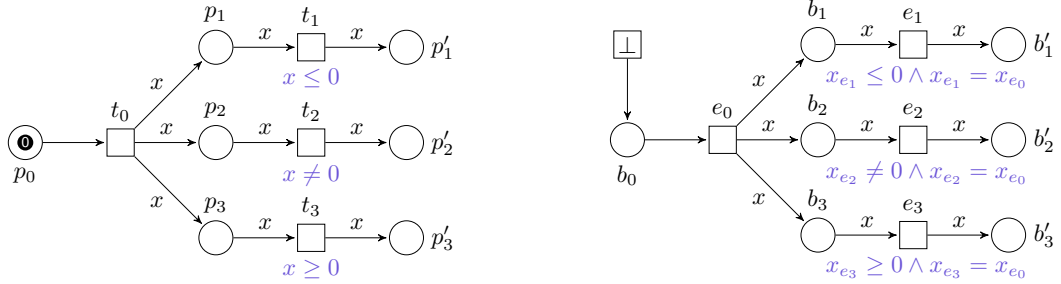
(a) A high-level Petri net with $Col = \mathbb{Z}$ and $Var = \{x\}$.

(b) The symbolic unfolding of the net in (a), with the events $\{e_1, e_2, e_3\}$ in color conflict.

Figure 2: Illustration and example of nodes in color conflict.

the set $\{e_1, e_2, e_3\}$ is also not in structural conflict, and the events are not causally related. However, there now *is* a color conflict between these three events, since $\bigwedge_{e \in \{e_1, e_2, e_3\}} pred(e)$ implies $x_{e_0} \leq 0 \wedge x_{e_0} \neq 0 \wedge x_{e_0} \geq 0$, which obviously is not satisfiable. In contrast, each of the sets $\{e_i, e_j\}$ with $i, j \in \{1, 2, 3\}$, $i \neq j$ is *not* in color conflict. This makes each of the sets $\{b_i', b_j'\}$ a co-set, while $\{b_1', b_2', b_3'\}$ is not a co-set.

Having employed the notions of conflict, we come to one of the most important definitions when dealing with unfoldings, namely *configurations*.

Definition 1.5. (Configuration [11])
A *(symbolic) configuration* is a set of high-level events that is free of structural conflict and color conflict, and causally closed. The configurations in a symbolic branching process β are collected in the set $\mathcal{C}(\beta)$.

Recall that B_0 are the initial conditions occupied in all initial cuts. For a configuration C, we define by $\text{cut}(C) := (B_0 \cup (C \rightarrow)) \setminus (\rightarrow C)$ the high-level conditions that are occupied after any execution of C. Note that $\text{cut}(C)$ is a co-set, and that $\emptyset$ is a configuration with $\text{cut}(\emptyset) = B_0$.

Let $e \in E$ be a high-level event. We define the so-called *cone configuration* $[e] := \{e' \in E \mid e' \leq e\}$. Additionally, we define the sets $Var_e := \{v_e \mid v \in Var(e)\}$ and $Var_\perp := \{v_\perp^b \mid b \in B_0\}$ of indexed variables, and for a set $E' \subseteq E \cup \{\perp\}$ we denote $Var_{E'} := \bigcup_{e \in E'} Var_e$. Note that, for every event e, $pred(e)$ is a predicate over the variables $Var_{[e] \cup \{\perp\}}$.

1.3. Properties of the symbolic unfolding.

Having recalled the definitions and formal language from [11], we now delve into the novel aspects of this paper. We state three analogues of well-known properties of the Unfolding of P/T Petri nets for the symbolic unfolding of high-level nets. These properties are:

(i) The cuts in the unfolding represent precisely the reachable markings in the net.

(ii) For every transition that can occur in the net, there is an event in the unfolding with corresponding label (and vice versa).

(iii) The unfolding is complete in the sense that for any configuration, the part of the unfolding that "lies after" that configuration is the unfolding of the original net with the initial markings being the ones represented by the configurations cut.

The properties are stated in Prop. 1.7, Prop. 1.8, and Prop. 1.9, respectively.

To express these properties, we introduce the notion of *instantiations* of configurations C, choosing a mode for every event in C without creating color conflicts. This is realized by assigning to each variable $v_e \in Var_{C \cup \{\bot\}}$ a value in Col, such that the above defined predicates evaluate to $true$. For each $e \in C$, the assignment of values to the indexed variables in Var_e corresponds to a mode of e.

Definition 1.6. (Instantiation of Configuration)
For a given configuration C, an *instantiation of C* is a function $\theta : Var_{C \cup \{\bot\}} \to Col$, such that $\forall e \in C \cup \{\bot\} : pred(e)[\theta] \equiv true$, i.e., it satisfies all predicates in the configuration. The set of instantiations of a given configurations C is denoted by $\Theta(C)$.

Note that, by definition, every configuration C has an instantiation θ. We denote by $\mathrm{cut}(C, \theta) := \{(b, c) \mid b \in \mathrm{cut}(C) \wedge \theta(\mathbf{v_e}(b)) = c\} \subseteq B \times Col$ the *cut* of an "instantiated configuration", and by $\mathrm{mark}(C, \theta) := \{\!\!\{ (h(b), c) \mid (b, c) \in \mathrm{cut}(C, \theta) \}\!\!\}$ its *marking*. We collect both of these in $\mathcal{K}(C) := \{\mathrm{cut}(C, \theta) \mid \theta \in \Theta(C)\}$ and $\mathcal{M}(C) := \{\mathrm{mark}(C, \theta) \mid \theta \in \Theta(C)\}$. Note that in this notation, for the empty configuration we have $\mathcal{K}(\emptyset) = \mathcal{K}_0$ and $\mathcal{M}(\emptyset) = \mathcal{M}_0$.

Proposition 1.7. Let N be a high-level Petri net and Υ its symbolic unfolding. Then $\mathcal{R}(N) = \{\mathrm{mark}(C, \theta) \mid C \in \mathcal{C}(\Upsilon), \theta \in \Theta(C)\}$.

Proof:
The proof is an easy induction over the number n of transitions/events needed to reach a respective marking/cut. The induction anchor $n = 0$ is proved by using that π is an initial homomorphism which gives $\mathcal{M}_0 = \{\!\!\{ (\pi(b), c) \mid (b, c) \in K_0 \}\!\!\} \mid K_0 \in \mathcal{K}_0\} = \{\!\!\{ (\pi(b), c) \mid (b, c) \in K \}\!\!\} \mid K \in \mathcal{K}(\emptyset)\} = \{\mathrm{mark}(\emptyset.\theta) \mid \theta \in \Theta(\emptyset)\}$. The induction step is realized by Prop. 1.8. □

Proposition 1.8. The symbolic unfolding $\Upsilon = (U, \pi)$ with events E of a high-level Petri net $N = (P, T, F, \iota, \mathcal{M}_0)$ satisfies $\forall C \in \mathcal{C}(\Upsilon) \, \forall \theta \in \Theta(C) \, \forall t \in T \, \forall \sigma \in \Sigma(t) :$

$$\mathrm{mark}(C, \theta)[t, \sigma\rangle \Leftrightarrow \exists e \in E : \pi(e) = t \wedge \mathrm{cut}(C, \theta)[e, \sigma\rangle.$$

Proof:
Let $U = (B, E, H, \iota, \mathcal{K}_0)$, and let $C \in \mathcal{C}(\Upsilon), \theta \in \Theta(C), t \in T, \sigma \in \Sigma(t)$.

Let $\mathrm{mark}(C, \theta)[t, \sigma\rangle$, which means

$$pre(t, \sigma) \leq \mathrm{mark}(C, \theta) = \{\!\!\{ (\pi(b), \theta(\mathbf{v_e}(b))) \mid (b, \theta(\mathbf{v_e}(b))) \in \mathrm{cut}(C, \theta) \}\!\!\},$$

Let $B' \subseteq \mathrm{cut}(C)$ be a set of conditions s.t.

$$pre(t, \sigma) = \{\!\!\{ (\pi(b), \theta(\mathbf{v_e}(b))) \mid b \in B' \}\!\!\}.$$

Aiming a contradiction, assume there is *no* $e \in E$ s.t. $\pi(e) = t$ and $\mathrm{cut}(C, \theta)[e, \sigma\rangle$: we extend Υ by such an event. We add to E an event $\widetilde{e}$ with $\pi(\widetilde{e}) = t$ and $\iota(\widetilde{e}) = \iota(t)$. Choose for every $b \in B'$ a variable $v^b \in Var$ s.t.

$$\{\!\!\{\, (\pi(b), v^b) \mid b \in B' \,\}\!\!\} = \{\!\!\{\, (p, v) \mid (p, v) \in pre(t) \,\}\!\!\} \quad (= pre(t)).$$

We define $pre(\widetilde{e}) = \{\!\!\{\, (b, v^b) \mid b \in B' \,\}\!\!\}$. Then we have $\{\!\!\{\, (\pi(b), v) \mid (b, v) \in pre(\widetilde{e}) \,\}\!\!\} = pre(t) = pre(\pi(\widetilde{e}))$. For every $(p, v) \in post(t)$, we then add $post(t)(p, v)$ conditions b with $\pi(b) = p$ to B and add $(\widetilde{e}, v, b)$ to H. We thus get $post(\pi(\widetilde{e})) = \{\!\!\{\, (\pi(b), v) \mid (b, v) \in post(\widetilde{e}) \,\}\!\!\}$. We now created a symbolic branching process bigger than Υ, contradicting that Υ is the symbolic unfolding.

Conversely, assume $\exists e \in E : \pi(e) = t \wedge \mathrm{cut}(C, \theta)[e, \sigma\rangle$. Then $pre(e, \sigma) \leq \mathrm{cut}(C, \theta)$, and therefore, $pre(t) = \{\!\!\{\, (\pi(b), v) \mid (b, v) \in pre(e) \,\}\!\!\} \leq \{\!\!\{\, (\pi(b), v) \mid (b, v) \in \mathrm{cut}(C, \theta) \,\}\!\!\} = \mathrm{mark}(C, \theta)$, meaning $\mathrm{mark}(C, \theta)[t, \sigma\rangle$. $\qquad\square$

Given a configuration C of a symbolic branching process $\beta = (O, h)$, we define $\Uparrow C$ as the pair (O', h'), where O' is the unique subnet of O whose set of nodes is $\{x \in B \cup E \mid x \notin (C \cup \to C) \wedge \forall y \in C : \neg(y \sharp x) \wedge (C \cup \{x\}$ is not in color conflict$)\}$ with the set $\mathcal{K}(C)$ of initial cuts, and h' is the restriction of h to the nodes of O'. The branching process $\Uparrow C$ is referred to as the *future* of C.

Proposition 1.9. If β is a symbolic branching process of $(\mathcal{N}, \mathcal{M}_0)$ and C is a configuration of β, then $\Uparrow C$ is a branching process of $(\mathcal{N}, \mathcal{M}(C))$. Moreover, if β is the unfolding of $(\mathcal{N}, \mathcal{M}_0)$, then $\Uparrow C$ is the unfolding of $(\mathcal{N}, \mathcal{M}(C))$.

Proof:
Let $\Uparrow C = (O', h')$ with $O' = (B', E', F', \iota', \mathcal{K}(C))$. To show that O' is an occurrence net, we have to show $i - iv$ from the Definition 1.2. $i - iii$ are purely structural properties and follow from the fact that O is an occurrence net. iv is satisfied since $\forall b \in \mathrm{cut}(C) \; \forall K \in \mathcal{K}(C) : \sum_{c \in Col} K(b, c) = 1$ and $\forall b \in B' \setminus \mathrm{cut}(C) \; \forall K \in \mathcal{K}(C) : \sum_{c \in Col} K(b, c) = 0$. h' is a homomorphism that is injective on events with the same preset since h is, and that h' is initial follows by Prop. 1.7 and Prop. 1.8.

When β is the symbolic unfolding of $(\mathcal{N}, \mathcal{M}_0)$, then the maximality of $\Uparrow C$ follows from the maximality of β, making $\Uparrow C$ the symbolic unfolding of $(\mathcal{N}, \mathcal{M}(C))$. $\qquad\square$

2.   Finite & complete prefixes of symbolic unfoldings

We combine ideas from [8] (computing small finite and complete prefixes of unfoldings) with results from [11] (symbolic unfoldings of high-level Petri nets) to define and construct complete finite prefixes of symbolic unfoldings of high-level Petri nets. We generalize the concepts and the ERV-algorithm from [8] for safe P/T Petri nets to a class of safe high-level Petri nets, and compare this generalization to the original. We will see that for P/T nets interpreted as high-level nets, all generalized concepts (i.e., complete prefixes, adequate orders, cut-off events), and, as a consequence, the result of the generalized ERV-algorithm, all coincide with their P/T counterparts.

We start by lifting the definition of completeness to the level of symbolic unfoldings. Together with Prop. 1.7 and Prop. 1.8, this can be seen as a direct translation from the low-level case described, e.g., in [8].

Definition 2.1. (Complete prefix)
Let $\beta = (O, h)$ be a prefix of the symbolic unfolding of a high-level Petri net N, with events E'. Then β is called *complete* if for every reachable marking M in N there exists $C \in \mathcal{C}(\beta)$ and $\theta \in \Theta(C)$ s.t.

  i) $M = \mathrm{mark}(C, \theta)$,   and

  ii) $\forall t \in T \, \forall \sigma \in \Sigma(t) : \; M[t, \sigma\rangle \;\Rightarrow\; \exists e \in E' : h(e) = t \; \wedge \; \mathrm{cut}(C, \theta)[e, \sigma\rangle.$

We now define the class $\mathbf{N_F}$ of high-level Petri nets for which we generalize the construction of finite and complete prefixes of the unfolding of *safe* P/T Petri nets from [8]. We discuss the properties defining this class, and describe how it generalizes safe P/T nets.

Definition 2.2. (Class $\mathbf{N_F}$)
The class $\mathbf{N_F}$ contains all finite high-level Petri nets $N = (P, T, F, \iota, \mathcal{M}_0)$ satisfying the following three properties:

(1) The net is *safe*, i.e., in every reachable marking there lies at most 1 color on every place (formally; $\forall M \in \mathcal{R}(N) \, \forall p \in P : \sum_{c \in Col} M(p, c) \leq 1$).

(2) Guards are written in a decidable theory with the set Col as its domain of discourse.

(3) The net has **f**initely many reachable markings (formally; $|\mathcal{R}(N)| < \infty$).

We require the safety property **(1)** for two reasons; on the one hand, to avoid adding to the already heavy notation. On the other hand, while we think that a generalization to bounded high-level Petri nets is possible, it comes with all the troubles known from going from safe to k-bounded in the P/T case in [8], plus the problems arising from the expressive power of the high-level formalism. We therefore postpone this generalization to future work. Note that, under the safety condition, we can w.l.o.g. assume $\mathcal{N}$ to be ordinary (i.e., $\forall x, y \in P \cup T : \sum_{v \in Var} F(x, v, y) \leq 1$), since transitions violating this property could never fire. The finiteness of $\mathcal{N}$ implies that we can assume Var to be finite.

While property **(2)** seems very strong, the goal is an algorithm that generates a complete finite prefix of the symbolic unfolding of a given high-level Petri net. The definition of symbolic branching processes requires the predicate of every event added to the prefix to be satisfiable, and the predicates are build from the guards in the given net. Thus, satisfiability checks in the generation of the prefix seem for now inevitable. An example for such a theory is Presburger arithmetic [14], which is a first-order theory of the natural numbers with addition. The guards in the example from Figure 1a are expressible in Presburger arithmetic.

We need property **(3)** to ensure that the generalized version of the cut-off criterion from [8] yields a finite prefix constructed in the generalized ERV-Algorithm. $|\mathcal{R}(N)| < \infty$ can be ensured by having a finite set Col of colors. In Sec. 4, we identify a class of high-level Petri nets with infinitely many reachable markings for which the algorithm works with an adapted cut-off criterion.

Under these three assumptions we generalize the finite safe P/T Petri nets considered in [8]: every such P/T net can be seen as a high-level Petri net with $Col = \{\bullet\}$ and all guards being *true*, and thus satisfying the three properties above. Replacing the safety property **(1)** by a respective "k-bounded property" would result in a generalization of k-bounded P/T nets. In Sec. 3, we compare the result of the generalized ERV-algorithm Alg. 1 applied to a high-level net to the result of the original ERV-algorithm from [8] applied to the high-level net's expansion.

For the rest of the section let $N = (P, T, F, \iota, \mathcal{M}_0) \in \mathbf{N_F}$ with symbolic unfolding $\Upsilon = (U, \pi) = (B, E, H, \iota, \mathcal{K}_0, \pi)$.

2.1. Generalizing adequate orders and cut-off events

We lift the concept of adequate orders on the configurations of an occurrence net to the level of symbolic unfoldings. A main property of adequate orders is the preservation by finite *extensions*, which are defined as for P/T-nets (cp. [8]):

Given a configuration C, we denote $C \cup D$ by $C \oplus D$ if $C \cup D$ is a configuration such that $C \cap D = \emptyset$. We say that $C \oplus D$ is an *extension* of C, and that D is a *suffix* of C. Obviously, for a configuration C', if $C \subsetneq C'$ then there is a nonempty suffix D of C such that $C \oplus D = C'$. For a configuration $C \oplus D$, denote by $O(C|D) = (\mathrm{cut}(C) \cup {\to}D \cup D{\to}, D, H', \mathcal{K}(C))$ the occurrence net "around D" from $\mathrm{cut}(C)$, where H' is the restriction of H to the nodes of $O(C|D)$. Note that for every finite configuration C with an extension $C \oplus D$, we have that D is a configuration of $\Uparrow C$.

We abbreviate for a marking M the fact $\exists \theta \in \Theta(C \oplus D) : \mathrm{mark}(C, \theta|_{Var_{C \cup \{\bot\}}}) = M$ by $C[\![M]\!]D$ to improve readability. Thus, $C[\![M]\!]D$ means that the transitions corresponding to the events in D can fire from $M \in \mathcal{M}(C)$.

Since we consider safe high-level Petri nets, we can relate two cuts representing the same set of places in the following way:

Definition 2.3. Let $C_1, C_2 \in \mathcal{C}(\Upsilon)$ with $\pi(\mathrm{cut}(C_1)) = \pi(\mathrm{cut}(C_2))$. Then there is a unique bijection $\phi : \mathrm{cut}(C_1) \to \mathrm{cut}(C_2)$ preserving π. We call this mapping $\phi_{C_1}^{C_2}$.

The now stated Prop. 2.4 is a weak version of the arguments in [8], where the Esparza et al. infer from the low-level version of Prop. 1.9 that if the cuts of two low-level configurations represent the same marking in the low-level net, then their futures are isomorphic, and the respective (unique) isomorphism maps the suffixes of one configuration to the suffixes of the other.

Proposition 2.4. Let C_1 and C_2 be two finite configurations in Υ, and let D be a suffix of C_1. If there is a marking $M \in \mathcal{M}(C_1) \cap \mathcal{M}(C_2)$ s.t. $C_1[\![M]\!]D$, then there is a unique monomorphism $\varphi_{1,D}^2 : O(C_1|D) \to \Uparrow C_2$ that satisfies $\varphi_{1,D}^2(\mathrm{cut}(C_1)) = \mathrm{cut}(C_2)$ and preserves the labeling π. For this monomorphism we have that $\varphi_{1,D}^2(D)$ is a suffix of C_2.

Notation. For functions $f : X \to Y$ and $f' : X' \to Y$ with $X \cap X' = \emptyset$ we define $f \uplus f' : X \cup X' \to Y$ by mapping x to $f(x)$ if $x \in X$ and to $f'(x)$ if $x \in X'$.

Proof:
By induction over the size $k = |D|$ of the suffix D.

Base case $k = 0$. This means $D = \emptyset$. Then $O(C_1|D) = (\mathrm{cut}(C_1), \emptyset, \emptyset, \mathcal{K}(C_1))$. Since $M \in \mathcal{M}(C_1) \cap \mathcal{M}(C_2)$, we know that $\pi(\mathrm{cut}(C_1)) = \pi(\mathrm{cut}(C_2))$. Since we only consider safe nets, $\varphi_{1,D}^2$ is uniquely realized by $\phi_{C_1}^{C_2} : \mathrm{cut}(C_1) \to \mathrm{cut}(C_2)$ from Def. 2.3.

Induction step. Let $k > 0$. Let $\theta \in \Theta(C_1 \oplus D)$ s.t. $\mathrm{mark}(C_1, \theta|_{Var_{C_1 \cup \{\bot\}}}) = M$. Let $e \in Min(D)$. Then for $\sigma = [v \leftarrow \theta(v_e)]_{v \in Var(e)}$ we have $M[\pi(e), \sigma\rangle$. Thus, by Prop. 1.8, $\exists e' \in E : \pi(e') = \pi(e) \wedge C_2 \oplus \{e'\} \in \mathcal{C}(\Upsilon)$. This means ${\to}e' \subseteq (B_0 \cup (C_2{\to})) \setminus ({\to}C_2)$; else, $C_2 \oplus \{e'\}$ would

not be a configuration. Thus, e' is an event in $\Uparrow C_2$. Since $\pi(e) = \pi(e')$, we get by definition of homomorphisms that $\{(\pi(b), v) \mid (b, v) \in post(e)\} = \{(\pi(b), v) \mid (b, v) \in post(e')\}$. The net N is safe, therefore we can define the bijection $\phi_1 : (e\rightarrow) \rightarrow (e'\rightarrow)$ by $\phi_1(b) = b' \Leftrightarrow \pi(b) = \pi(b')$. We now define $\varphi_1 : O(C_1|\{e\}) \rightarrow \Uparrow C_2$ by $\varphi_1 = \phi_{C_1}^{C_2} \uplus \{e \mapsto e'\} \uplus \phi_1$, which is a homomorphism satisfying the claimed conditions.

Let now $C_1' = C_1 \cup \{e\}$, $C_2' = C_2 \cup \{\varphi_1(e)\}$ and $D' = D \setminus \{e\}$. We then have for M' given by $M[\pi(e), \sigma\rangle M'$ that $C_1'[\![M']\!]D'$, $M' \in \mathcal{M}(C_1') \cap \mathcal{M}(C_2')$, and $|D'| < k$. Thus, by the induction hypothesis, we get that there is a unique monomorphism $\varphi_2 : O(C_1'|D') \rightarrow \Uparrow C_2'$ satisfying the conditions above. Since φ_1 and φ_2 coincide on $\mathrm{cut}(C_1')$, we can now define $\varphi_{1,D}^2$ by "gluing together" φ_1 and φ_2 at $\mathrm{cut}(C_1')$.

This proves the claim for finite extensions. For an infinite extension, every node also contained in a finite extension. Due to uniqueness of the homomorphisms, we can define the $\varphi_{1,D}^2$ in the case of an infinite D as the union of all homomorphisms of smaller finite extensions. $\square$

Equipped with Prop. 2.4, we can now lift the concept of adequate order to the level of symbolic branching processes. Compared to [7, 8], the monomorphism $\varphi_{1,D}^2$ defined above replaces the isomorphism I_1^2 between $\Uparrow C_1$ and $\Uparrow C_2$ for two low-level configurations C_1, C_2 representing the same marking.

Definition 2.5. (Adequate order)
A partial order $\prec$ on the set of finite configurations of the symbolic unfolding of a high-level Petri net is an *adequate order* if:

  i) $\prec$ is well-founded,

 ii) $C_1 \subset C_2$ implies $C_1 \prec C_2$, and

iii) $\prec$ is preserved by finite extensions in the following way: if C_1, C_2 are two finite configurations, and $C_1 \oplus D$ is a finite extension of C_1 such that there is a marking $M \in \mathcal{M}(C_1) \cap \mathcal{M}(C_2)$ satisfying $C_1[\![M]\!]D$, then the monomorphism $\varphi_{1,D}^2$ from above satisfies $C_1 \prec C_2 \Rightarrow C_1 \oplus D \prec C_2 \oplus \varphi_{1,D}^2(D)$.

In the case of a P/T net we have $|\mathcal{M}(C)| = 1$ for every configuration C, and therefore, Def. 2.5 coincides with its P/T version [8]. We could alternatively generalize the P/T case by replacing '$\exists M \in \mathcal{M}(C_1) \cap \mathcal{M}(C_2)$ s.t. $C_1[\![M]\!]D$' by '$\mathcal{M}(C_1) = \mathcal{M}(C_2)$', and use the isomorphism I_1^2 between $\Uparrow C_1$ and $\Uparrow C_2$ to define preservation by finite extension. However, in the upcoming generalization of the ERV-algorithm from [8], the generalized cut-off criterion exploits property iii) of adequate orders. Using '$\mathcal{M}(C_1) = \mathcal{M}(C_2)$' would produce an exponential blowup of the generated prefix's size. This is circumvented by using '$\exists M \in \mathcal{M}(C_1) \cap \mathcal{M}(C_2)$ s.t. $C_1[\![M]\!]D$', which however leads to obtaining merely a monomorphism that depends on the considered suffix D, instead of an isomorphism between the futures. We now show that this monomorphism sufficient:

The upcoming proof that the generalized ERV-algorithm is complete is structurally analogous to the respective proof in [8]. It uses that, under the conditions of Def. 2.5 iii), we also have $C_2 \prec C_1 \Rightarrow C_2 \oplus \varphi_{1,D}^2(D) \prec C_1 \oplus D$. This result would directly be obtained if $\varphi_{1,D}^2$ was an isomorphism, as I_1^2 is in the low-level case. However, a monomorphism is an isomorphism when its codomain is restricted

to its range. This idea is used in the proof of the following proposition, which states that $\varphi^2_{1,D}$ indeed satisfies the above property.

Proposition 2.6. Let $\prec$ be an adequate order. Under the conditions of Def. 2.5 iii) the monomorphism $\varphi^2_{1,D}$ also satisfies $C_2 \prec C_1 \Rightarrow C_2 \oplus \varphi^2_{1,D}(D) \prec C_1 \oplus D$.

Proof:
Let $D' = \varphi^2_{1,D}(D)$. We first show that $\varphi^1_{2,D'}(D') = D$.

Let $\varphi_1 : O(C_1|D) \to \varphi^2_{1,D}(O(C_1|D))$ be the isomorphism that acts on $O(C_1|D)$ as $\varphi^2_{1,D}$ does, and let $\varphi_2 : O(C_2|D') \to \varphi^1_{2,D'}(O(C_2|D'))$ be the isomorphism that acts on $O(C_2|D')$ as $\varphi^1_{2,D'}$ does. Since $\varphi_1^{-1} : \varphi^2_{1,D}(O(C_1|D)) \to O(C_1|D)$ and $O(C_1|D) \subset \Uparrow C_1$, and $\varphi_1^{-1}(\varphi^2_{1,D}(D)) = D$ is a suffix of C_1, we get by Prop. 2.4 that $\varphi_1^{-1} = \varphi_2$, which means $\varphi^1_{2,D'}(D') = D$.

Assume now $C_2 \prec C_1$. From the proof of Prop. 2.4 we see that $C_2 \llbracket M \rrbracket \varphi^2_{1,D}(D)$. Thus, we get by the definition of adequate order and the result above that $C_2 \oplus \varphi^2_{1,D}(D) \prec C_1 \oplus \varphi^1_{2,\varphi^2_{1,D}(D)}(\varphi^2_{1,D}(D))$ $= C_1 \oplus D$.   □

In [8], Esparza et al. discuss three adequate orders on the configurations of the low-level unfolding. In particular, they present a *total* adequate order that uses the *Foata normal form* of configurations. Using such a total order in the algorithm limits the size of the resulting finite and complete prefix: it contains at most $|\mathcal{R}(N)|$ non cut-off events. All three adequate orders presented in [8] can be directly lifted to the configurations of the symbolic unfolding by exchanging every low-level term by its high-level counterpart. The lifted order using the Foata normal form is still a total order. We include these discussions in App. A.1.

We now define cut-off events in a symbolic unfolding. In the low-level case [8], e is a cut-off event if there is another event e′ satisfying $[e'] \prec [e]$ and $\mathrm{mark}([e]) = \mathrm{mark}([e'])$, which ensures that the future of e needs not be considered further. In the high-level case, we generalize these conditions to high-level events e. However, we do not require the existence of *one* other high-level event e' with $[e'] \prec [e]$ and $\mathcal{M}([e]) = \mathcal{M}([e'])$. While this would still be a valid cut-off criterion and would lead to finite and complete prefixes, the upper bound on the size of such a prefix would be exponential in the number of markings in the original net. Instead, we check whether $\mathcal{M}([e])$ is contained in the union of *all* $\mathcal{M}([e'])$ with $[e'] \prec [e]$. This criterion expresses that we have already seen every marking in $\mathcal{M}([e])$ in the prefix β under construction, and therefore need not consider the future of e any further. By this, we obtain the same upper bounds as in [8], as discussed later.

Definition 2.7. (Cut-off event)
Let $\prec$ be an adequate order on the configurations of the symbolic unfolding of a high-level Petri net. Let β be a prefix of the symbolic unfolding containing a high-level event e. The high-level event e is a *cut-off* event in β (w.r.t. $\prec$) if $\mathcal{M}([e]) \subseteq \bigcup_{[e'] \prec [e]} \mathcal{M}([e'])$.

For P/T Petri nets, this definition corresponds to the cut-off events defined in [8], since in this case $|\mathcal{M}([e])| = 1$ for all events e.

2.2.   The generalized ERV-algorithm

We present the algorithm for constructing a finite and complete prefix of the symbolic unfolding of a given high-level Petri net. It is a generalization of the ERV-algorithm from [8], and is structurally equal (and therefore looks very similar). However, the algorithm is contingent upon the previous section's work of generalizing adequate orders and cut-off events, which ultimately enables us to adopt this structure.

A crucial concept of the ERV-algorithm is the notion of "possible extensions", i.e., the set of individual events that extend a given prefix of the unfolding. In Def. 2.8, we lift this concept to the high-level formalism. We do so by isolating the procedure of adding high-level events in the algorithm from [11] which generates the complete symbolic unfolding of a given high-level Petri net (but does not terminate if the symbolic unfolding is infinite).

We define the data structures similarly to [8]. There, an event is given by a tuple $e = (t, B')$ with $h(e) = t \in T$ and $pre(e) = B' \subseteq B$, and a condition given by a tuple $b = (p, e)$ with $h(b) = p \in P$ and $pre(b) = \{e\} \subseteq E$. The finite and complete prefix is a set of such events and transitions.

In the high-level case, we need more information inside the tuples. A high-level event is given by a tuple $e = (t, X, pred)$ described by $h(e) = t$, $pre(e) = X \subseteq B \times Var$, and $pred(e) = pred$. Analogously, a high-level condition is given by a tuple $b = (p, (e, v), pred)$, where $h(b) = p$, $pre(b) = (e, v) \in (E \times Var) \cup (\{\bot\} \times \{v^b \mid b \in B_0\})$, and $pred(\mathbf{e}(b)) = pred$.

Definition 2.8. (Possible extensions)
Let $\beta = (O, h)$ be a branching process of a high-level Petri net N. The *possible extensions* $PE(\beta)$ are the set of tuples $e = (t, X, pred)$ where t is a transition of N, and $X \subseteq B \times Var$ satisfying

- $\{b \mid (b, v) \in X\}$ is a co-set, and $pre(t) = \{(h(b), v) \mid (b, v) \in X\}$,
- $pred = \textit{loc-pred} \wedge \left(\bigwedge_{(b,v) \in X} pred(\mathbf{e}(b)) \right)$ is satisfiable,
  where $\textit{loc-pred} = \iota(t)[v \leftarrow v_e]_{v \in Var(e)} \wedge \left(\bigwedge_{(b,v) \in X} v_e = \mathbf{v_e}(b) \right)$,
- β does not contain $(t, X, pred)$.

Since the notion of co-set in high-level occurrence nets is achieved by the direct translation from low-level occurrence nets plus the "color conflict freedom", possible extensions in a prefix β can be found by searching first for sets of conditions that are not in structural conflict as in the low-level case, and then checking whether these sets are in color conflict.

Alg. 1 is a generalization of the ERV-Algorithm in [8] for complete finite prefixes of the low-level unfolding. The structure is taken from there, with the only difference being the special initial transition $\bot$. It takes as input a high-level Petri net $N \in \mathbf{N_F}$ and assumes a given adequate order $\prec$.

Example 2.9. Consider the running example N from Figure 1a. Alg. 1 produces the complete finite prefix marked by the blue line in Figure 1b. Cut-off events are shaded blue.

Starting with the initial conditions a' and b', the possible extensions are α' and β'. assuming $[\alpha'] \prec [\beta']$, we first attach α' and a condition c' corresponding to the output place c of α, and then analogously β' and the condition d'.

For α' we have $\mathcal{M}([\alpha']) = \{\!\{(c, n), (b, 0)\}\!\} \mid n \in \{1, \dots, m\}\}$ and analogously, for β' we have $\mathcal{M}([\beta']) = \{\!\{(a, 0), (d, n)\}\!\} \mid n \in \{1, \dots, m\}\}$. Since we have not seen these markings before,

---

Algorithm 1: Generalization of the ERV-Algorithm from [8] for complete finite prefixes.

Data: High-level Petri net $N = (P, T, F, \iota, \mathcal{M}_0) \in \mathbf{N_F}$.

Result: A complete finite prefix Fin of the symbolic unfolding of N.

$Fin \leftarrow \{\bot\}$;

$pred(\bot) \leftarrow \bigvee_{M_0 \in \mathcal{M}_0} \bigwedge_{(p,c) \in M_0} v_\bot^{b_p} = c$;

foreach $p \in P_0$ **do**

    Create a fresh condition $b_p = (p, (\bot, v^{b_p}), pred(\bot))$;

    $Fin \leftarrow Fin \cup \{b\}$;

end

$pe \leftarrow PE(Fin)$;

$cut\text{-}off \leftarrow \emptyset$;

while $pe \neq \emptyset$ **do**

    Pick $e = (t, X, pred)$ from pe such that $[e]$ is minimal w.r.t. $\prec$;

    if $[e] \cap cut\text{-}off = \emptyset$ **then**

        $Fin \leftarrow Fin \cup \{e\}$;

        foreach $(p, v) \in post(t)$ **do**

            Create a fresh condition $b = (p, (e, v), pred)$;

            $Fin \leftarrow Fin \cup \{b\}$;

        end

        $pe \leftarrow PE(Fin)$;

        if e is a cut-off event of Fin **then**

            $cut\text{-}off \leftarrow cut\text{-}off \cup \{e\}$;

        end

    else

        $pe \leftarrow pe \setminus \{e\}$

    end

end

---

neither α' nor β' is a cut-off event. Thus, we have the possible extensions t' and ε'. For t' we have $\mathcal{M}([t']) = \{\!\{\,\}\!\}$, since no tokens are in the net after firing t. However, we have not seen the empty marking $\{\!\{\,\}\!\}$ before, so formally, t' is not a cut-off event.

For ε' we have $\mathcal{M}([\varepsilon']) = \{\!\{(c, n), (d, n')\}\!\} \mid n, n' \in \{1, \dots, m\}\}$. Corresponding cuts can be reached in the prefix constructed so far by concurrently firing α' and β'. However, no marking $\{\!\{(c, n), (d, n')\}\!\}$ is represented by a *cone* configuration before ε', and therefore ε' does *not* satisfy $\mathcal{M}([\varepsilon']) \subseteq \bigcup_{[e'] \prec [\varepsilon']} \mathcal{M}([e'])$. This means ε' is not a cut-off event and we have to proceed with the possible extensions t'' and ε''.

Since $\mathcal{M}([t'']) = \{\!\{\,\}\!\} = \mathcal{M}([t'])$ with $[t'] \prec [t'']$, have that t'' is a cut-off event. This, however, has no impact on the prefix since we cannot continue after t'' anyway. For ε'' we have $\mathcal{M}([\varepsilon'']) = \{\!\{(c, n), (d, n')\}\!\} \mid n, n' \in \{1, \dots, m\}\} = \mathcal{M}([\varepsilon'])$ with $[\varepsilon'] \prec [\varepsilon'']$. This makes ε'' also a cut-off event. We therefore have no more possible extensions, and the algorithm terminates. In the figure, this is indicated by the blue lines.

We now prove correctness of Alg. 1 analogously to [8], by stating two propositions – one each to show that the prefix is finite and complete, respectively. The proof structure is also as in [8], but adapted to the setting of high-level Petri nets and symbolic unfoldings.

Proposition 2.10. After applying Alg. 1 to a high-level Petri net $N \in \mathbf{N_F}$, the result Fin is finite.

Given an event e, define the *depth* of e as the length of the longest chain of events $e_1 < e_2 < \cdots < e$; the depth of e is denoted by $d(e)$.

Proof:
As in [8], we prove the following results (1) – (3):

(1) For every event e of Fin, $d(e) \leq |\mathcal{R}(N)| + 1$,

(2) For every event e of Fin, the sets $pre(e)$ and $post(e)$ are finite, and

(3) For every $k \geq 0$, Fin contains only finitely many events e such that $d(e) \leq k$.

This works exactly as in [8], with minor adaptations to the generalization of cut-offs in the symbolic unfolding in (1):

(1) Let $n = |\mathcal{R}(N)|$. Every chain of events $e_1 < e_2 < \cdots < e_n < e_{n+1}$ in the unfolding contains an event e_i, $i > 1$, s.t. $\mathcal{M}([e_i]) \subseteq \bigcup_{j=1}^{i-1} \mathcal{M}([e_j])$, since, if every $\mathcal{M}([e_j])$, $j = 1, \ldots, n$, contains a marking not contained in $\bigcup_{k=1}^{j-1} \mathcal{M}([e_k])$, then finally $\bigcup_{j=1}^{n} \mathcal{M}([e_j])$ contains all n markings. This makes e_{n+1} a cut-off event.

(2) By the construction in the algorithm we see that there is a bijection between $post(e)$ and $post(h(e))$, and similarly for $pre(e)$ and $pre(h(e))$. The result then follows from the finiteness of N.

(3) By complete induction on k. The base case, $k = 0$, is trivial. Let E_k be the set of events of depth at most k. We prove that if E_k is finite then E_{k+1} is finite. By (2) and the induction hypothesis, $post(E_k)$ is finite. Since $\{b \mid \exists v \in Var : (b,v) \in pre(E_{k+1})\} \subseteq \{b \mid \exists v \in Var : (b,v) \in post(E_k)\}$, we get by property *iv* in the definition of occurrence nets that E_{k+1} is finite.
□

Proposition 2.11. After applying Alg. 1 to a high-level Petri net $N \in \mathbf{N_F}$, the result Fin is complete.

The proof of this proposition also has the same general structure as the respective proof in [8]. However here we use the generalizations of adequate order, possible extensions, and the cut-off criterion to symbolic branching processes.

Proof:
We first show that for every reachable marking in N there exists a configuration in Υ satisfying a) from the definition of complete prefixes, and then show that one of these configurations (a minimal one) also satisfies b).

(1) Let M be an arbitrary reachable marking in N. Then by Prop. 1.7, we have that there is a $C_1 \in \mathcal{C}(\Upsilon)$ s.t. $M \in \mathcal{M}(C_1)$. Let $\theta_1 \in \Theta(C_1)$ s.t. $M = \mathrm{mark}(C_1.\theta_1)$. If C is not a configuration in Fin, then it contains a cut-off event e_1, and so $C_1 = [e_1] \oplus D$ for some set D of events. Let $M_1 = \mathrm{mark}([e_1].\theta_1|_{Var_{[e_1] \cup \{\bot\}}}) \in \mathcal{M}([e_1])$. By the definition of cut-off event, there exists an event e_2 with $[e_2] \prec [e_1]$ and $M_1 \in \mathcal{M}([e_2])$. Since we have $C_1 [\![M_1]\!] D$, we get by Prop. 2.4 that the monomorphism $\varphi_1 := \varphi_{[e_1],D}^{[e_2]} : O([e_1]|D) \to \Uparrow[e_2]$ exists and that $\varphi_1(D)$ is a suffix of $[e_2]$. By Prop. 2.6 we know

$$C_2 := [e_2] \oplus \varphi_1(D) \prec [e_1] \oplus D = C_1.$$

Let $\theta_2' \in \Theta([e_2])$ s.t. $M_1 = \mathrm{mark}([e_2], \theta_2')$. Define now $\theta_2 \in \Theta(C_2)$ by $\theta_2 = \theta_2' \uplus \theta_2''$, where $\theta_2'' : Var_{\varphi_1(D)} \to Col$ is given by $\theta_2''(v_{\varphi_1(e)}) = \theta_1(v_e)$. By this construction we get $M = \mathrm{mark}(C_2, \theta_2) \in \mathcal{M}(C_2)$.

If C_2 is not a configuration of Fin, then we can iterate the procedure and find a configuration C_3 such that $C_3 \prec C_2$ and $M \in \mathcal{M}(C_3)$. The procedure cannot be iterated infinitely often because $\prec$ is well-founded. Therefore, it terminates in a configuration of Fin.

(2) Let now C be a minimal configuration w.r.t. $\prec$ s.t. $M \in \mathcal{M}(C)$, and let $t \in T$, $\sigma \in \Sigma(t)$ s.t. $M[t, \sigma\rangle$. If C contains some cut-off event, then we can apply the arguments of a) to conclude that Fin contains a configuration $C' \prec C$ such that $M \in \mathcal{M}(C')$. This contradicts the minimality of C. So C contains no cut-off events. Let $\theta \in \Theta(C)$ s.t. $M = \mathrm{mark}(C, \theta)$. Since $pre(t.\sigma) \subseteq M$, we have that there is a co-set $B_{t,\sigma} \subseteq \mathrm{cut}(C)$ s.t. $pre(t, \sigma) = \{(h(b), \theta(\mathbf{v_e}(b))) \mid b \in B_{t,\sigma}\}$.

Let now $X := \{(b, v) \mid b \in B_{t,\sigma}, (h(b), v) \in pre(t)\}$. We then have $\forall (b, v) \in X : \sigma(v) = \theta(\mathbf{v_e}(b))$.

We now show that

$$pred := \iota(t)[v \leftarrow v_e]_{v \in Var(e)} \wedge \Big(\bigwedge_{(b,v) \in X} v_e = \mathbf{v_e}(b) \Big) \wedge \bigwedge_{(b,v) \in X} pred(\mathbf{e}(b))$$

is satisfiable. Let $\theta' := \theta \uplus (\sigma \circ [v_e \mapsto v]_{v \in Var(e)})$. Then

- $\iota(t)[v \leftarrow v_e]_{v \in Var(e)}[\theta'] \equiv \iota(t)[\sigma] \equiv true$, and
- $\big(\bigwedge_{(b,v) \in X} v_e = \mathbf{v_e}(b) \big)[\theta'] \equiv \big(\bigwedge_{(b,v) \in X} \sigma(v) = \theta(\mathbf{v_e}(b)) \big) \equiv true$, and
- $\bigwedge_{(b,v) \in X} pred(\mathbf{e}(b))[\theta'] \equiv \bigwedge_{(b,v) \in X} pred(\mathbf{e}(b))[\theta] \equiv true$, since $\theta \in \Theta(C)$.

Thus, $pred[\theta'] \equiv true$. Therefore, $e = (t, X, pred)$ is a possible extension and added in the execution of the algorithm. Then we directly have $e \notin C$, $h(e) = t$, and with the same arguments as in a), we get $C \cup \{e\} \in \mathcal{C}(Fin)$ and $\theta \uplus (\sigma \circ [v_e \mapsto v]_{v \in Var(e)}) \in \Theta(C \cup \{e\})$, which means $\mathrm{cut}(C, \theta)[e, \sigma\rangle$. Since we chose θ independently of t and σ, this concludes the proof.

$\square$

Notice that by this construction, as described in [8], we get that if $\prec$ is a total order, then Fin contains at most $|\mathcal{R}(N)|$ non cut-off events. As mentioned in Sec. 2.1, the total adequate order defined in [8] can be lifted to the configurations in the symbolic unfolding, where it again is total (cp. App. A.1). Thus, we generalized the possibility to construct such a small complete finite prefix by application of Alg. 1 with $\prec$ being a total adequate order.

3. High-level versus P/T expansion

Every high-level Petri net represents a P/T Petri net (in which the places can only carry a number tokens with color ●) with the same behavior, called its expansion. Markings in a P/T Petri net describe only how many tokens lie on each place. Each transitions only has one possible firing mode that takes and/or places a fixed number of tokens from resp. onto each connected place.

In this section we state in Lemma 3.2 that the expansion of a finite complete prefix of the unfolding of a high-level Petri net is a finite and complete prefix of the unfolding of the expanded high-level Petri net. This means the generalization of complete prefixes is "canonical", and compatible with the established low-level concepts. We then compare for our running example the results of

- applying the generalized ERV-algorithm Alg. 1 to obtain a complete finite prefix of the symbolic unfolding of a given high-level Petri net, and

- first expanding a given high-level Petri net and then applying the ERV-algorithm from [8] for a complete finite prefix of the (P/T) unfolding.

The procedure of constructing the represented P/T Petri net $\mathrm{Exp}(N)$ (called the *expansion*) of a high-level Petri net N is well established (cp., e.g., Chapter 2.4 in [3]), and we describe it here only briefly; the places of $\mathrm{Exp}(N)$ are given by $\mathsf{P} = \{p.c \mid p \in P, c \in Col\}$, and its transitions by $\mathsf{T} = \{t.\sigma \mid t \in T, \sigma \in \Sigma(t)\}$. There is an arc from $p.c$ to $t.\sigma$ iff $(p, c) \in pre(t, \sigma)$, and analogously for arcs from transitions to places. Markings in $\mathrm{Exp}(N)$ are functions $\mathsf{M} : \mathsf{P} \to \mathbb{N}$, describing how often the only color ● lies on each place $p.c$. Every such marking corresponds to a marking M in the high-level net N, with $M(p, c) = \mathsf{M}(p.c)$, and a transition t can fire in mode σ from M iff $t.\sigma$ can fire from M. Thus, we say that N and $\mathrm{Exp}(N)$ have the same behavior. For a finite high-level Petri net N, the expansion $\mathrm{Exp}(N)$ is finite iff Col is finite.

The *(low-level) unfolding* of a P/T Petri net N is a tuple $\Upsilon(\mathsf{N}) = (\mathsf{U}(\mathsf{N}), \pi^{\mathsf{N}})$, where $\mathsf{U}(\mathsf{N})$ is an *occurrence net*, and $\pi^{\mathsf{N}} : \mathsf{U}(\mathsf{N}) \to \mathsf{N}$ is a *Petri net homomorphism* such that $(\mathsf{U}(\mathsf{N}), \pi^{\mathsf{N}})$ is a *maximum branching process* of N. Since the definition of (low-level) branching processes and homomorphisms is firstly well established in the literature, and secondly so similar to their corresponding high-level definitions, we omit them here and refer the reader for example to [6, 8].

With the notation of low-level unfoldings of P/T nets we can define for a high-level occurrence net O, the P/T occurrence net $\mathrm{Exp}_\mathsf{O}(O) := \mathsf{U}(\mathrm{Exp}(O))$ (i.e., the occurrence net from the unfolding $\Upsilon(\mathrm{Exp}(O)) = (\mathsf{U}(\mathrm{Exp}(O)), \pi^{\mathrm{Exp}(O)})$). We abbreviate $\pi^{\mathrm{Exp}(O)}$ by π^O. The operator Exp_O therefore maps high-level occurrence nets to occurrence nets (cf. [4]). Let now $\beta = (O, h)$ be a symbolic branching process of N. Then we can define the *expanded symbolic branching process* $\mathrm{Exp}_\mathsf{O}(\beta) = (\mathrm{Exp}_\mathsf{O}(O), \mathsf{h})$ of $\mathrm{Exp}(N)$ with the homomorphism $\mathsf{h} : \mathrm{Exp}_\mathsf{O}(O) \to \mathrm{Exp}(N)$, defined

by $h(e) = t.\sigma \Leftrightarrow \pi^O(e) = e.\sigma \wedge h(e) = t$ and $h(b) = p.c \Leftrightarrow \pi^O(b) = b.c \wedge h(b) = p$ for events e resp. conditions b in $\mathrm{Exp}_O(O)$. The following diagram serves as an overview:
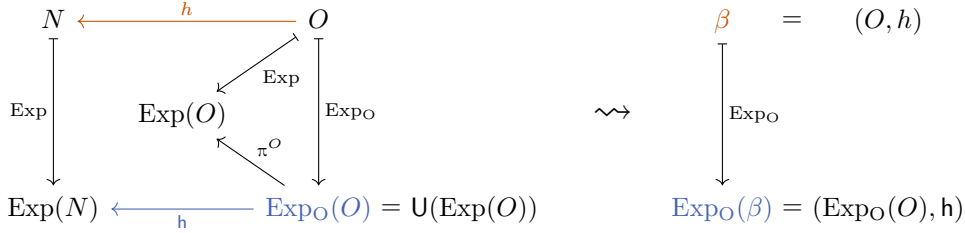
The following result is shown in [4]. It states that, for a high-level Petri net N, the unfolding of N's expansion is isomorphic to the expanded symbolic unfolding of N.

Lemma 3.1. ([4], Sec. 4.1)
$\Upsilon(\mathrm{Exp}(N)) \simeq \mathrm{Exp}_O(\Upsilon(N))$.

With this result, we state the following:

Lemma 3.2. Let N be a high-level Petri net and β be a prefix of $\Upsilon(N)$. Then β is finite and complete if and only if $\mathrm{Exp}_O(\beta)$ is a finite and complete prefix of $\Upsilon(\mathrm{Exp}(N))$.

The proof uses the results from Prop. 1.7 and Prop. 1.8, since the definition of completeness on the symbolic level is a direct translation from its P/T analogue.

Proof:
Let $\beta = (O, h)$ be finite and complete. From Lemma 3.1 we already know that $\mathrm{Exp}_O(O) \subseteq \mathsf{U}(\mathrm{Exp}(N))$. Since $\mathrm{Exp}_O(\beta)$ is a branching process of $\mathrm{Exp}(N)$, we see that is a prefix of the unfolding of $\mathrm{Exp}(N)$. Also, $\mathrm{Exp}_O(\beta)$ is obviously finite since O is a finite high-level occurrence net.

We now prove that $\mathrm{Exp}_O(\beta) = (\mathrm{Exp}_O(O), h)$ is complete. Let M be a reachable marking in $\mathrm{Exp}(N)$. Then the high-level marking M defined by $M(p, c) = \mathsf{M}(p.c)$ is reachable in N. Thus, since β is complete, there is a configuration $C \in \mathcal{C}(\beta)$ and an instantiation $\theta \in \Theta(C)$ satisfying a) and b) from Def. 2.1. This means there is a firing sequence $K_0[e_1, \sigma_1\rangle K_1 \ldots [e_n, \sigma_n\rangle K_n$ with $\{e_1, \ldots, e_n\} = C$, $\sigma_i = \theta \circ [v \mapsto v_{e_i}]_{v \in Var(e_i)}$, and $K_n = \mathrm{cut}(C, \theta)$ (meaning $M = \mathrm{mark}(C, \theta) = \{\!| (h(b), c) \mid (b, c) \in K_n |\!\}$). Then, in $\mathrm{Exp}(O)$, the marking $\{\!| b.c \mid (b, c) \in K_n |\!\}$ is reachable from the initial marking $\{\!| b.c \mid (b, c) \in K_0 |\!\}$ by the firing sequence $(e_1.\sigma_1, \ldots, e_n.\sigma_n)$. Thus, there is a configuration $\mathsf{C} = \{e_1, \ldots, e_n\}$ in $\mathsf{U}(\mathrm{Exp}(O)) = \mathrm{Exp}_O(O)$ with $\forall i : \pi^O(e_i) = e_i.\sigma_i$. Then, by the definition of h, we get $\mathrm{mark}^h(\mathsf{C}) := \{\!| h(b) \mid b \in \mathrm{cut}(\mathsf{C}) |\!\} = \{\!| h(b).c \mid (b, c) \in K_n |\!\} = \mathsf{M}$.

Let now $t.\sigma \in \mathsf{T}$ s.t. $\mathsf{M}[t.\sigma\rangle$. Then $M[t, \sigma\rangle$. Since C, θ satisfy property b) from Def. 2.1, we know that $\exists e \in E$ s.t. $e \notin C$, $h(e) = t$, and $C, \theta[e, \sigma\rangle$. This means, in $\mathrm{Exp}(\beta)$, we have $\{\!| b.c \mid (b, c) \in K_n |\!\}[e.\sigma\rangle$. Thus, there exists an e in $\mathsf{U}(\mathrm{Exp}(\beta))$ such that $\mathsf{C}[e\rangle$ and $\pi^O(e) = e.\sigma$, which again means that $h(e) = h(e).\sigma = t.\sigma$. This proves that $\mathrm{Exp}_O(\beta)$ is complete.

The other direction works analogously. $\qquad\square$

We can now compare the two complete finite prefixes resulting from the original ERV-algorithm from [8] applied to $\text{Exp}(N)$ and the generalized ERV-algorithm Alg. 1 applied to $N \in \mathbf{N_F}$. From the definition of the generalized cut-off criterion we get that both these prefixes have the same depth. However, due to the high-level representation, the breadth of the symbolic prefix can be substantially smaller. This is the case for our running example:
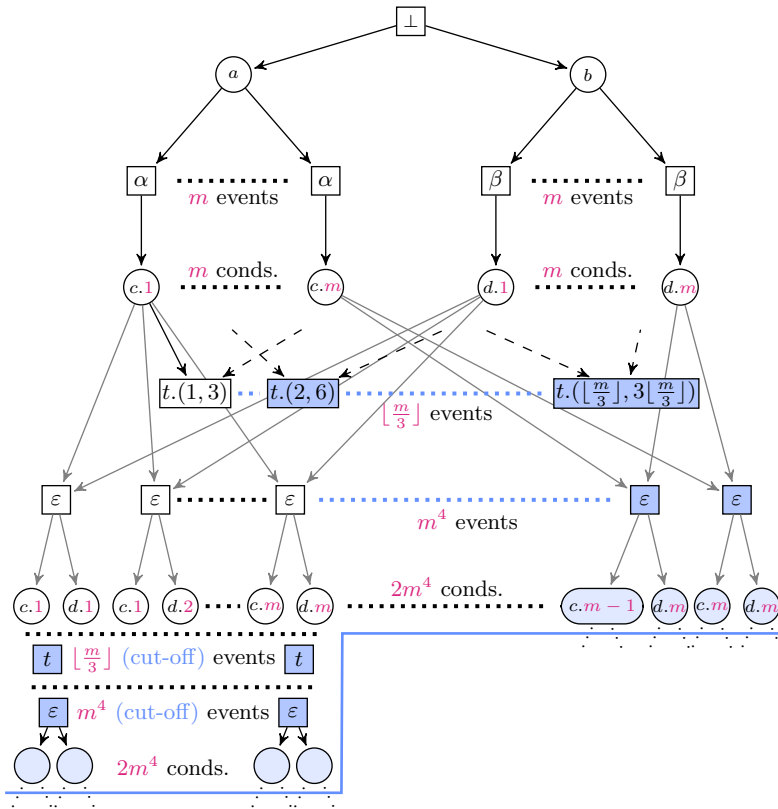

Figure 3: The complete finite prefix of $\Upsilon(\text{Exp}(N))$ of the running example N from Figure 1a calculated by the original ERV-Algorithm.

Example 3.3. Consider again $N \in \mathbf{N_F}$ from Figure 1a with $Col = \{0, 1, \dots, m\}$ for a fixed $m > 0$. The finite complete prefix of $\Upsilon(\text{Exp}(N))$ is depicted in Fig. 3. Instead of giving each event/condition a distinct name, we indicated the label of each node inside of it. For events with label ε we even omitted the mode, since it is derivable from the connected conditions. Cut-off events and their output conditions are again shaded blue, and the blue line indicates the complete finite prefix resulting from the original ERV-algorithm.

After firing an instance of α and an instance of β, we arrive at conditions with labels $c.k$ and $d.\ell$. If these satisfy $\ell = k \cdot 3$ then we can fire an instance of t, which means we have $\lfloor \frac{m}{3} \rfloor$ such events. Only the first is no cut-off event, since their configurations all represent the same (empty) marking.

This empty marking is however also the reason why we cannot continue even after the first instance of t.

For each combination of an output condition of an instance of α (the m instances of c) and an output condition of an instance of β (the m instances of d), we have m^2 possibilities to fire an instance of ε. The reason for this is that in the high-level net N, the output variables are independent of the input variables. This leads to m^4 many ε-events of depth 2.

All except the first m^2 of those are cut-off events. After the non cut-off events, however, we have to repeat the part from above for the ERV-algorithm to terminate. All in all the complete finite prefix contains $6m^4 + 4m + 2\lfloor \frac{m}{3} \rfloor + 2$ nodes for every fixed m in the color class $Col = \{0, 1, \ldots, m\}$. The complete finite prefix of the *symbolic* unfolding $\Upsilon(N)$ that is shown in Figure 1b, on the other hand has the same number of nodes for every m.

Generalizing this example to a family of nets gives the following proposition:

Proposition 3.4. For every $n \in \mathbb{N}$, there is a family $(N_m^n)_{m \in \mathbb{N}^{>0}}$ of high-level nets in $N_m^n \in \mathbf{N_F}$ such that every N_m^n has the set of colors $Col = \{0, \ldots, m\}$ and satisfies that

- the complete finite prefix of $\Upsilon(N_m^n)$ obtained by Alg. 1 has the same number of nodes for every m,

- the number of nodes in the low-level prefix of $\Upsilon(\mathrm{Exp}(N_m^n))$ obtained by the original ERV-algorithm is greater than m^n.

In particular, the benchmark family Fork And Join, presented in Sec. 6.2.1 satisfies this property.

4. Handling infinitely many reachable markings

Unfoldings of unbounded P/T Petri nets (i.e., with infinitely many markings) have been investigated in [15, 16], and in [17] concurrent well-structured transition systems with infinite state space are un-folded. When applying the generalized ERV-algorithm, Alg. 1, to high-level Petri nets with infinitely many reachable markings (therefore violating (3) from the definition of $\mathbf{N_F}$), the proof for finiteness of the resulting prefix does not hold anymore: the proof of Prop. 2.10, step (1), is a generalization of the proof of the respective claim in [8] (which uses the pigeonhole principle). It is argued that we cannot have $|\mathcal{R}(N)| + 1$ consecutive events s.t. their cone configurations each generate a marking in the net not seen before, and we thus have a cut-off event. When we deal with infinitely many markings, this argument cannot be made.

In this section, we introduce a class $\mathbf{N_{SC}}$ of safe high-level nets, called symbolically compact, that have possibly infinitely many reachable markings (and therefore an infinite expansion), generalizing the class $\mathbf{N_F}$. We then proceed to make adaptions to Alg. 1 (i.e., to the used cut-off criterion), so that it generates a finite and complete prefix of the symbolic unfolding for any $N \in \mathbf{N_{SC}}$.

The following Lemma precisely describes the finite high-level Petri nets for which a finite and complete prefix of the symbolic unfolding exists. They are characterized by having a bound on the number of steps needed to arrive at every reachable marking. For the proof we argue that in the case of such a bound, the symbolic unfolding up to depth $n + 1$ is a finite and complete prefix, and that in the absence of such a bound no depth of a prefix suffices for it to be complete.

Lemma 4.1. For a finite high-level Petri net $N = (\mathcal{N}, \mathcal{M}_0)$ there exists a finite and complete prefix of $\Upsilon(N)$ if and only if there exists a bound $n \in \mathbb{N}$ such that every marking in $\mathcal{R}(N)$ is reachable from a marking in $\mathcal{M}_0$ by firing at most n transitions.

Proof:
From Prop. 1.7 and Prop. 1.8 we see that for a finite high-level Petri net with such a bound n, the prefix of the symbolic unfolding containing exactly the events e with $d(e) \leq n + 1$ is complete. Finiteness of this prefix follows from the finiteness of the original net and the definition of homomorphism.

Assume now that no such bound exists, and, for the purpose of contradiction, assume that there is a finite and complete prefix β of $\Upsilon(N)$. Denote $\widetilde{n} = \max\{|C| \mid C \in \mathcal{C}(\beta)\} < \infty$. Then there exists a marking $M \in \mathcal{R}(N)$ for which we have to fire at least $\widetilde{n} + 1$ transitions to reach it. Again from Prop. 1.7 and Prop. 1.8 it follows that a configuration C with $M \in \mathcal{M}(C)$ must contain at least $\widetilde{n} + 1$ events, contradicting that β is complete. $\qquad\qquad\square$

4.1. Symbolically compact high-level Petri nets

We use the result of Lemma 4.1 to define the class $\mathbf{N_{SC}}$ of high-level nets for which we adapt the algorithm for constructing finite and complete prefixes of the symbolic unfolding.

Definition 4.2. (Class $\mathbf{N_{SC}}$)
A finite high-level Petri net N is called *symbolically compact* if it satisfies **(1)** and **(2)** from Def. 2.2, and

(3*) There is a bound $n \in \mathbb{N}$ on the number of transition firings needed to reach all markings in $\mathcal{R}(N)$.

We denote the class containing all symbolically compact high-level Petri nets by $\mathbf{N_{SC}}$.

Note that in the case of a (finite, safe) P/T net, property **(3*)** is equivalent to **(3)** (i.e., $|\mathcal{R}(N)| < \infty$). However, this is *not* true for all high-level nets N: while $|\mathcal{R}(N)| < \infty$ still implies **(3*)** (meaning $\mathbf{N_F} \subseteq \mathbf{N_{SC}}$), the reverse implication does not hold, as our running example from Figure 1a demonstrates when we change the set of colors to $Col = \mathbb{N}$: it then still satisfies **(1)** and **(2)**, with $\mathcal{R}(N) = \{\{\!|\,(a, 0), (b, 0)\,|\!\}, \{\!|\ |\!\}\} \cup \{\{\!|\,(c, n), (b, 0)\,|\!\}, \{\!|\,(a, 0), (d, n')\,|\!\}, \{\!|\,(c, n), (d, n')\,|\!\} \mid n, n' \in \mathbb{N}\}$. So we have infinitely many markings that can all be reached by firing at most two transitions, meaning the net satisfies **(3*)** and is therefore symbolically compact.

Lemma 4.1 implies that the class $\mathbf{N_{SC}}$ of symbolically compact nets contains exactly all high-level Petri nets satisfying **(1)** and **(2)** for which a finite and complete prefix of the symbolic unfolding exists (independently of whether the number of reachable markings is finite). Since the reachable markings of a high-level Petri net and its expansion correspond to each other, this observation leads to an interesting subclass $\mathbf{N_{SC}} \setminus \mathbf{N_F}$ of symbolically compact high-level Petri nets that have infinitely many reachable markings. For every net N in this subclass

- there exists a finite and complete prefix of $\Upsilon(N)$,   but
- there does *not* exist a finite and complete prefix of $\Upsilon(\mathrm{Exp}(N))$.

In particular, the original ERV-algorithm cannot be applied to $\mathrm{Exp}(N)$, since this expansion is an infinite net.

An example for such a net is our running example from Figure 1a when we replace the color class $Col = \{0, 1, \ldots, m\}$ by $Col = \mathbb{N}$. Much simpler is the following net, also with $Col = \mathbb{N}$:

$$p \; \textcircled{\small 0} \quad \overset{x}{\longrightarrow} \quad \square \; t$$
$$\underset{y}{\longleftarrow}$$
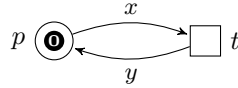
Obviously, every reachable marking $\{\!| \, (p, n) \, |\!\}$ with $n \in \mathbb{N}$ can be reached by firing t one time in mode $\{x \leftarrow 0, y \leftarrow n\}$, so the net is symbolically compact. The expansion of this net however is infinite, and the original ERV-algorithm does not terminate when applied to it.

4.2. Insufficiency of the cut-off criterion for N_{SC}

Naturally, the question arises whether the generalized ERV-Algorithm, Alg. 1, also yields a finite and complete prefix of a symbolically compact input net. For many examples (like the simple one above) this is the case. However, there are symbolically compact high-level Petri nets for which Alg. 1 does *not* terminate.

The criterion for nontermination of Alg. 1 is that in the unfolding of the net, there is an infinite sequence of cone configurations $[e_1], [e_2], \ldots$ with $[e_1] \prec [e_2] \prec \ldots$ such that

- $\forall i \in \mathbb{N}_{>0} : \{e_1, \ldots, e_i\} \in \mathcal{C}(\Upsilon)$, i.e., $(e_1, e_2, \ldots)$ is a fireable sequence in the symbolic unfolding,

- $\forall i \in \mathbb{N}_{>0} : \mathcal{M}([e_i]) \nsubseteq \bigcup_{[e'] \prec [e_i]} \mathcal{M}([e'])$, i.e., no event e_i is a cut-off event.

Note that in the second condition, the e' in the union are arbitrary events in the unfolding, and not restricted to the sequence $(e_1, e_2, \ldots)$. An example for such a net is shown in Figure 4a:

The set of colors is given by $Col = \mathbb{N}$. Initially there is a token 0 in each of the places a and c. The token on a can cycle between a and b by transitions α and β. Analogously, the other token can cycle between c and d by γ and δ. Additionally, in the inital marking, there is a color 1 on place p. This number can be increased by 1 by firing t. Thus, every number n can be placed on p by $n - 1$ firings of t. When, however, the two cycling tokens of color 0 are on places b and d, an arbitrary number n can be placed directly on p by firing ε. The net therefore is symbolically compact.

Examine now the unfolding in Figure 4b. Cut-off events and their output conditions are again shaded blue. For a cleaner presentation do not write the local predicate next to each event. For the event ε' we have $\mathcal{M}([\varepsilon']) = \{\!| \, (b, 0), (d, 0), (p, n) \, |\!\} \mid n \in \mathbb{N}\}$. This means by firing no event, only β'', only δ'', or both β'' and δ'' from the corresponding cut, we can represent every reachable marking in the net.

The sequence $[t^1], [t^2], \ldots$ of cone configurations, with the corresponding events shaded orange, now satisfies the criterion from above: The first condition is obviously satisfied. The sequence of events corresponds to firing t infinitely often, always increasing the number on p by 1. The cones $[t^i]$ are the only cone configurations where the cuts represent markings with *no tokens* on the two places b and d. For all other cones in the unfolding, there is a 0 on b and/or a 0 on d. Thus, no event t^i is a cut-off event. This means if Alg. 1 is applied to the net in Figure 4a it does not terminate, building a prefix containing every t^i with $i \in \mathbb{N}^+$.
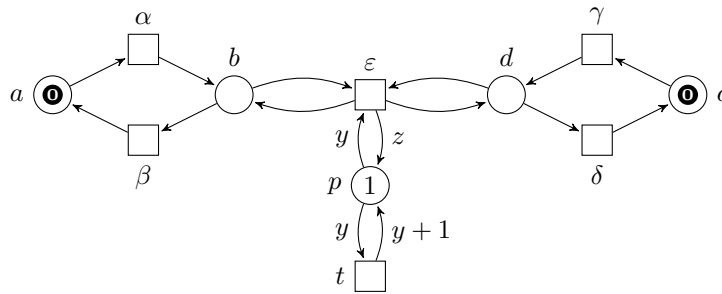
(a) A symbolically compact net with $Col = \mathbb{N}$.
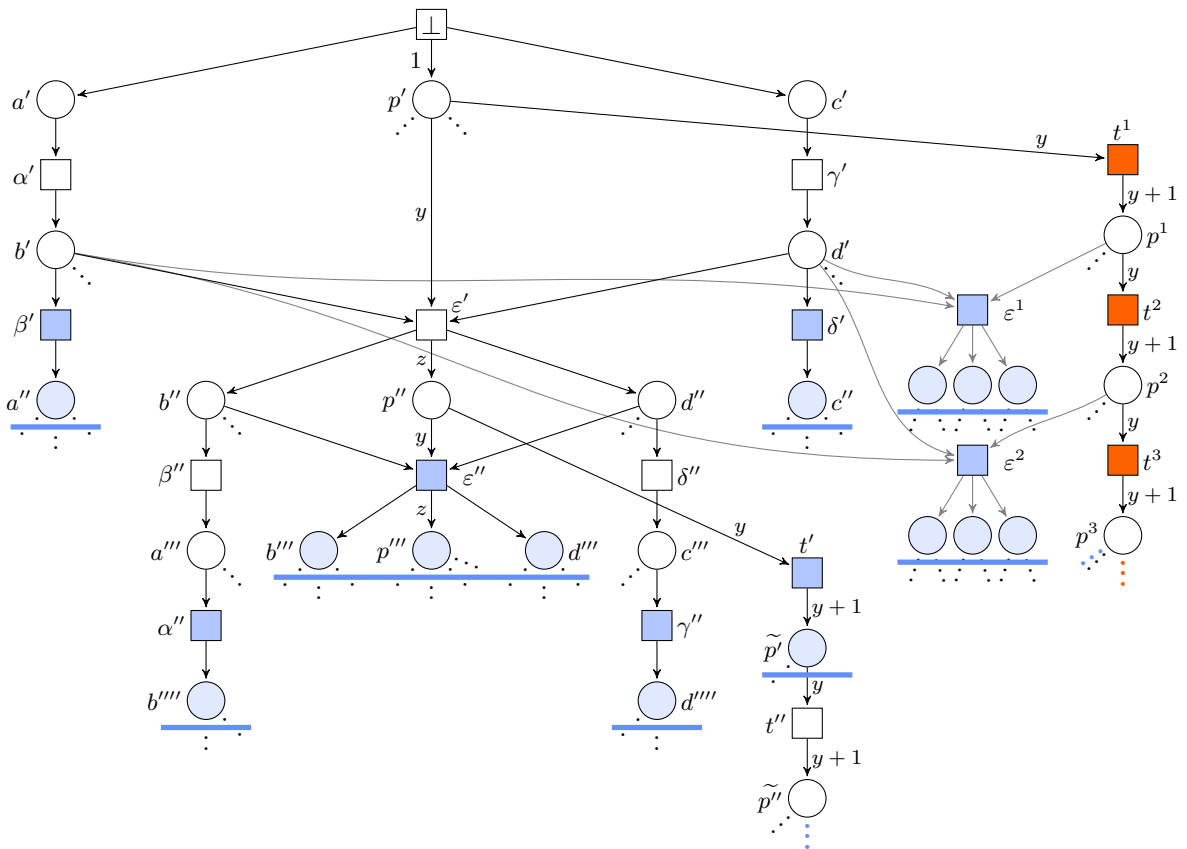
(b) A prefix of the symbolic unfolding of the net in (a).

Figure 4: A symbolically compact net in (a) where Alg. 1, trying to build a complete finite prefix of the symbolic unfolding shown in (b), does not terminate.

We have now shown that there are nets in $\mathbf{N_{SC}} \setminus \mathbf{N_F}$ where Alg. 1 does not terminate due to the insufficient cut-off criterion. A compelling avenue for further research lies in exploring the existence of subclasses $\mathbf{N}$ between $\mathbf{N_F}$ and $\mathbf{N_{SC}}$ (i.e., $\mathbf{N_F} \subset \mathbf{N} \subset \mathbf{N_{SC}}$) for which the cut-off criterion suffices and the algorithm terminates.

4.3. The finite prefix algorithm for symbolically compact nets

As previously discussed, the argument that states the existence of one event in a chain of $|\mathcal{R}(N)| + 1$ consecutive events, such that every marking represented by its cone configuration is contained in the union of all markings represented by previous cone configurations, cannot be applied in the case of an infinite number of reachable markings. Consequently, Alg. 1 may not terminate when applied to a net in $\mathbf{N_{SC}} \setminus \mathbf{N_F}$. However, condition **(3*)** guarantees that every marking reached by a cone configuration $[e]$ with depth $> n$ can be reached by a configuration C containing no more than n events.

For the algorithm to terminate, we need to adjust the cut-off criterion since we do not know whether C is also a cone configuration, as demanded in Def. 2.7. Therefore, we define *cut-off* events*, that generalize cut-off events. They only require that every marking in $\mathcal{M}([e])$ has been observed in a set $\mathcal{M}(C)$ for *any* configuration $C \prec [e]$, rather than just considering cone configurations:

Definition 4.3. (Cut-off* event)
Under the assumptions of Def. 2.7, the high-level event e is a *cut-off** event (w.r.t. $\prec$) if $\mathcal{M}([e]) \subseteq \bigcup_{C \prec [e]} \mathcal{M}(C)$.

We additionally assume that the used adequate order satisfies $|C_1| < |C_2| \Rightarrow C_1 \prec C_2$, so that every event with depth $> n$ will be a cut-off event. Since all adequate orders discussed in [8] satisfy this this property (cp. App. A.1), this is a reasonable requirement. This adaption and assumption now lead to:

Theorem 4.4. Assume a given adequate order $\prec$ to satisfy $|C_1| < |C_2| \Rightarrow C_1 \prec C_2$. When replacing in Alg. 1 the term "cut-off event" by "cut-off* event", it terminates for any input net $N \in \mathbf{N_{SC}}$, and generates a complete finite prefix of $\Upsilon(N)$.

Proof:
The properties of symbolic unfoldings that we stated in Sec. 1.3 are independent on the class of high-level nets. Def. 2.3 only uses that the considered net is safe, and so do Prop. 2.4 and Prop. 2.6. We therefore only have to check that the correctness proof for the algorithm still holds. In the proof of Prop. 2.10 (*Fin* is finite), the steps (2) and (3) are independent of the used cut-off criterion. In step (1), however, it is shown that the depth of events never exceeds $|\mathcal{R}(N)| + 1$. This is not applicable when $|\mathcal{R}(N)| = \infty$, as argued above. Instead we show:

(1*) For every event e of *Fin*, $d(e) \leq n+1$, where n is the bound on the number of transitions needed to reach all markings in $\mathcal{R}(N)$.

In the proof of Prop. 2.11, the cut-off criterion is used to show (by an infinite descent approach), for any marking $M \in \mathcal{R}(N)$ the existence of a minimal configuration $C \in Fin$ with $M \in \mathcal{M}(C)$. Due to the similarity of cut-off and cut-off*, this proof can easily be adapted to work as before:

Assume that at some point during the algorithm, we reach a state $(B', E', H', \iota', \mathcal{K}'_0)$ of the prefix under construction, such that there occurs a chain of events $e_1 < e_2 < \cdots < e_{n+1}$. We prove that e_{n+1} must be a cut-off* event. Let $M \in \mathcal{M}([e_{n+1}])$. Then, by definition of $\mathbf{N_{SC}}$, M can be reached by firing at most n transitions. Accordingly, from Prop. 1.8, we get that there is a configuration $C \in \Upsilon$ containing at most n events such that $M \in \mathcal{M}(C)$. As in the proof of Prop. 2.11, we can now follow that there is a configuration $\widetilde{C} \in \mathcal{C}(\Upsilon)$ such that $M \in \mathcal{M}(\widetilde{C})$ and $\widetilde{C} \prec C$, that contains no cut-off

event and is therefore in Fin. Since $|C| \leq n < n+1 \leq |[e_{n+1}]|$, we follow $\widetilde{C} \prec [e_{n+1}]$. So we have that $\forall M \in \mathcal{M}([e_{n+1}]) \exists \widetilde{C} \prec [e_{n+1}]$, which means that e_{n+1} is a cut-off* event. This proves that Fin is finite.

It remains to show termination. In the case of nets in $\mathbf{N_F}$, every object is finite, which, together with Prop. 2.10, leads to termination of the algorithm. For nets in $\mathbf{N_{SC}} \setminus \mathbf{N_F}$, however, there is at least one event e in Fin s.t. $|\mathcal{M}([e])| = \infty$. Thus, we have to show that we can check the cut-off* criterion in finite time. This follows from Cor. 5.4 in the next section, which is dedicated to symbolically representing markings generated by configurations. $\square$

4.4. Feasibility of symbolically compact nets and cut-off*

To check the cut-off* criterion of an event added to a prefix of the unfolding, we have to compare the set of markings represented by the cut of the event's cone configuration to *all* markings represented by cuts of smaller configurations. This means that we possibly have to store the whole state space.

This realization gives rise to two questions. Firstly, how do we manage the storage of an infinite number of markings? This query is addressed in Sec. 5, where we demonstrate how to symbolically represent the markings represented by a configuration's cut and how to check the cut-off* criterion within finite time. The prototype implementation outlined in Sec. 6.1 utilizes these methods for the $\mathbf{N_F}$-case.

The second question that arises asks how the complete finite prefix resulting from the generalized ERV-algorithm with the cut-off* criterion relates to a reachability graph – both in terms of size and computation time. However, as symbolically compact nets possibly have an infinite expansion, the reachability graph can be infinitely broad. Thus, at present this method provides a more viable solution compared to calculating the infinite reachability graph. However, we give an outlook on how a (finite) symbolic reachability tree of a symbolically compact net could possibly be constructed.

Outlook: Symbolic Reachability Trees of Symbolically Compact Nets.   The idea of a symbolic reachability tree has been realized for algebraic Petri nets in [18] by Karsten Wolf. However, in contrast to this work, we think that for the class of symbolically compact nets we can build a symbolic reachability tree that is complete.

The idea is to gradually extend for every subset P' of places a formula $\mathcal{R}_{P'}$ that symbolically describes all reachable markings that we have seen so far and have colors on exactly all places in P'. Initially, all formulae are $false$, except for $\mathcal{R}_{P_0}$, where P_0 are the initially marked places. $\mathcal{R}_{P_0}$ symbolically represents the set of initial markings.

The symbolic reachability tree is then constructed by starting with a root n_0 labeled with $f_{n_0} = \mathcal{R}_{P_0}$ representing the set of initial markings. For every transition t, we can determine whether t can fire in any mode from any marking represented by f_{n_0} by a satisfiability check. If t can fire, we add a new node n', and label it with a formula f' that symbolically represents all markings reached from firing t in any mode from any marking in $\mathcal{M}_0$. In all these markings, there are colors on the same set of places P'. If $f' \Rightarrow \mathcal{R}_{P'}$ then we end this branch. We then extend $\mathcal{R}_{P'}$ to $\mathcal{R}_{P'} \vee f'$.

By repeating this procedure in breadth-first order, we build a tree that symbolically represents all reachable markings. This tree should correspond to the complete finite prefix of the symbolic

unfolding of the net to which you added a shared resource (in form of a new place) that every transition consumes and recreates. This makes the system sequential without changing the sequential semantics. We give here only the idea of the tree, and not a formal definition. In future work we want to further investigate on this idea. We can then compare the complete finite prefix of the symbolic unfolding to the symbolic reachability tree.

5. Checking cut-offs symbolically

We show how to check whether a high-level event e is a cut-off* event symbolically in finite time. By definition, this means checking whether $\mathcal{M}([e]) \subseteq \bigcup_{C \prec [e]} \mathcal{M}(C)$. However, since the cut of a configuration can represent infinitely many markings, when applying the adapted algorithm we cannot simply store the set $\mathcal{M}(C)$ for every $C \in \mathcal{C}(Fin)$. Instead, we now define constraints that symbolically describe the markings represented by a configuration's cut. Checking the inclusion above then reduces to checking an implication of these constraints. Since we consider high-level Petri nets with guards written in a decidable theory, such implications can be checked in finite time.

At the end we see that this method can be easily adapted to symbolically check whether, in a prefix of the symbolic unfolding of a net $N \in \mathbf{N_F}$, an event e is a cut-off event in the sense of Def. 2.7. This method is also used in the implementation described in Sec. 6.1.

For the rest of this section, let $N = (P, T, F, \iota, \mathcal{M}_0) \in \mathbf{N_{SC}}$ with symbolic unfolding $\Upsilon(N) = (U, \pi) = (B, E, H, \iota, \mathcal{K}_0, \pi)$.

We first define for every condition b a new predicate $pred^{\odot}(b)$ by

$$pred^{\odot}(b) := pred(\mathbf{e}(b)) \wedge (\pi(b) = \mathbf{v_e}(b)).$$

This predicate now has (in an abuse of notation) an extra variable, named after its label $\pi(b)$. The remaining variables in $pred^{\odot}(b)$ are coming from $pred(\mathbf{e}(b))$ and given by $Var_{[\mathbf{e}(b)] \cup \{\perp\}}$. As we know, $pred(\mathbf{e}(b))$ evaluates to *true* under an assignment $\theta : Var_{[\mathbf{e}(b)] \cup \{\perp\}} \to Col$ if and only if a concurrent execution of $[\mathbf{e}(b)]$ with the assigned modes is possible (i.e., under every instantiation of $[\mathbf{e}(b)]$). In such an execution, $\theta(\mathbf{v_e}(b)) \in Col$ is placed on b. The predicate $pred^{\odot}(b)$ therefore can only be true if $\pi(b)$ is assigned a color that can be placed on b.

We now define for a co-set $B' \subseteq B$ of high-level conditions the constraint on B' as an expression with free variables $\pi(B') = \{\pi(b) \mid b \in B'\}$, describing which color combinations can lie on the places represented by the high-level conditions. We build the conjunction over all predicates $pred^{\odot}(b)$ for $b \in B'$ and quantify over all appearing variables v_e: the *constraint on B'* is defined by

$$\kappa(B') := \exists_{\bigcup_{b \in B'} Var_{[\mathbf{e}(b)] \cup \{\perp\}}} : \bigwedge_{b \in B'} pred^{\odot}(b).$$

We denote by $\Xi(B')$ the set of variable assignments $\vartheta : \pi(B') \to Col$ that satisfy $\kappa(B')[\vartheta] \equiv true$.

For a configuration C, we have that $B' = \mathrm{cut}(C)$ is a co-set, $\pi(B') = \pi(\mathrm{cut}(C))$ describes the set of places occupied in every marking in $\mathcal{M}(C)$. Note that in this case, we have $\bigcup_{b \in \mathrm{cut}(C)} Var_{[\mathbf{e}(b)]} = Var_C$, i.e., the bounded variables in $\kappa(\mathrm{cut}(C))$ are exactly the variables appearing in predicates in C.

For every instantiation θ of C we define a variable assignment $\vartheta_\theta : \pi(\mathrm{cut}(C)) \to Col$ by setting $\forall \pi(b) \in \pi(\mathrm{cut}(C)) : \vartheta_\theta(b) = \theta(\mathbf{v_e}(b))$. Instantiations of a configuration and the constraint on its cut are related as follows.

Lemma 5.1. Let $C \in \mathcal{C}(\Upsilon(N))$. Then $\Xi(\mathrm{cut}(C)) = \{\vartheta_\theta \mid \theta \in \Theta(C)\}$.

Proof:
The proof follows by construction of $pred^\odot$ and ϑ_θ: Let $\vartheta \in \Xi(\mathrm{cut}(C))$. Then $true \equiv \kappa(\mathrm{cut}(C))[\vartheta]$. Thus, there exists $\theta : Var_{C \cup \{\perp\}} \to Col$ s.t. $\left(\bigwedge_{b \in \mathrm{cut}(C)} pred^\odot(b) \right)[\vartheta][\theta] \equiv true$ and therefore

$$\left(\bigwedge_{b \in \mathrm{cut}(C)} pred(\mathbf{e}(b))[\theta] \right) \wedge \left(\bigwedge_{b \in \mathrm{cut}(C)} \vartheta(\pi(b)) = \theta(\mathbf{v_e}(b)) \right) \quad \equiv \quad true. \tag{1}$$

From the inductive definition of $pred$ then follows that $\forall e \in C \cup \{\perp\} : pred(e)[\theta] \equiv true$. Thus, θ is an instantiation of C, and $\vartheta_\theta = \vartheta$, as shown by the posterior conjunction in (1).

Let on the other hand $\theta \in \Theta(\mathrm{cut}(C))$. Then directly, by the definition of $pred^\odot(b)$ and ϑ_θ, we get $\left(\bigwedge_{b \in \mathrm{cut}(C)} pred^\odot(b) \right)[\vartheta_\theta][\theta] \equiv true$ and by the definition of $\kappa(\mathrm{cut}(C))$ that $\kappa(\mathrm{cut}(C))[\vartheta_\theta] \equiv true$, i.e., $\vartheta_\theta \in \Xi(\mathrm{cut}(C))$. $\qquad \square$

From the definition of $\mathcal{K}(C)$ and $\mathcal{M}(C)$ we get:

Corollary 5.2. Let $C \in \mathcal{C}(\Upsilon(N))$. Then $\mathcal{K}(C) = \{\{(b, \vartheta(\pi(b))) \mid b \in \mathrm{cut}(C)\} \mid \vartheta \in \Xi(\mathrm{cut}(C))\}$ and $\mathcal{M}(C) = \{\{\!\!\{ (\pi(b), \vartheta(\pi(b))) \mid b \in \mathrm{cut}(C) \}\!\!\} \mid \vartheta \in \Xi(\mathrm{cut}(C))\}$.

We now show how to check whether an event is a cut-off* event via the constraints defined above. For that, we first look at general configurations in Theorem 5.3, and then explicitly apply this result to cone configurations $[e]$ in Corollary 5.4.

Theorem 5.3. Let $C, C_1, \ldots, C_n$ be finite configurations in the symbolic unfolding of a safe high-level Petri net s.t. $\forall 1 \le i \le n : \pi(\mathrm{cut}(C)) = \pi(\mathrm{cut}(C_i))$. Then

$$\mathcal{M}(C) \subseteq \bigcup_{i=1}^n \mathcal{M}(C_i) \quad \text{if and only if} \quad \kappa(\mathrm{cut}(C)) \Rightarrow \bigvee_{i=1}^n \kappa(\mathrm{cut}(C_i)).$$

Proof:
Assume $\mathcal{M}(C) \subseteq \bigcup_{i=1}^n \mathcal{M}(C_i)$ and let $\vartheta \in \Xi(\mathrm{cut}(C))$. We have that $M_\vartheta := \{(\pi(b), \vartheta(\pi(b))) \mid b \in \mathrm{cut}(C)\} \in \mathcal{M}(C)$ by Cor. 5.2. Thus, $\exists 1 \le i \le n : M_\vartheta \in \mathcal{M}(C_i)$. This, again by Cor. 5.2, means $\exists \vartheta_i \in \Xi(\mathrm{cut}(C_i)) :$
$$M_\vartheta = \{(\pi(b'), \vartheta_i(\pi(b'))) \mid b' \in \mathrm{cut}(C_i)\}$$
This shows that $\vartheta = \vartheta_i$. Thus, $\kappa(\mathrm{cut}(C_i))[\vartheta] \equiv true$, which proves the implication.

Assume on the other hand $\kappa(\mathrm{cut}(C)) \Rightarrow \bigvee_{i=1}^n \kappa(\mathrm{cut}(C_i))$. Let $M \in \mathcal{M}(C)$. Then $\exists \vartheta \in \Xi(\mathrm{cut}(C)) : M = \{(\pi(b), \vartheta(\pi(b))) \mid b \in \mathrm{cut}(C)\}$. Thus, $\exists 1 \le i \le n : \kappa(\mathrm{cut}(C_i))[\vartheta] \equiv true$. Let $\vartheta_i = \vartheta$. Then $\vartheta_i \in \Xi(\mathrm{cut}(C_i))$, and $M = \{(\pi(b'), \vartheta_i(\pi(b'))) \mid b' \in \mathrm{cut}(C_i)\} \in \mathcal{M}(C_i)$, which completes the proof. $\qquad \square$

The following Corollary now gives us a characterization of cut-off* events in a symbolic branching process. It follows from Theorem 5.3 together with the facts that $\mathcal{M}(C_1) \cap \mathcal{M}(C_2) \neq \emptyset \Rightarrow \pi(\mathrm{cut}(C_1)) = \pi(\mathrm{cut}(C_2))$, and that $\prec[e]$ is finite.

Corollary 5.4. Let β be a symbolic branching process and e an event in β. Then e is a cut-off* event in β if and only if

$$\kappa(\mathrm{cut}([e])) \Rightarrow \bigvee_{\substack{C \prec [e] \\ h(\mathrm{cut}(C))=h(\mathrm{cut}([e]))}} \kappa(\mathrm{cut}(C)).$$

Thus, we showed how to decide for any event e added to a prefix of the unfolding whether it is a cut-off* event, namely, by checking the above implication in Cor. 5.4. Note that we can also check whether e is a *cut-off* event (w.r.t. Def. 2.7) by the implication in Cor. 5.4 when we replace all occurrences of "C" by "$[e']$" .

6. Implementation and experimental results

In this section, we delve into the implementation details of the generalized ERV-algorithm and discuss the results of our experiments. We give a concise overview of the technical decisions made during implementation and provide an evaluation of its performance across four novel benchmark families. We identify a property called "mode-determinism" that offers an indicator for whether it is faster to construct (a complete finite prefix of) the symbolic unfolding or the low-level unfolding.

6.1. Implementation specifics

Other tools designed for generating (prefixes of) P/T Petri net unfoldings include MOLE [19], CUNF [20, 21], and PUNF [22]. However, as these tools are specifically optimized for their intended purpose and do not cater to high-level Petri nets, we opted not to integrate the new algorithms into any of these frameworks. Furthermore, we refrain from conducting a speed comparison between our implementation and the aforementioned tools. The objective of Section 6 is to provide a comparison between two approaches: calculating a respective complete finite prefix of the low-level or the symbolic unfolding.

We have devised a prototype implementation called COLORUNFOLDER [23] written in the Java programming language. It serves a dual purpose as an implementation of the low-level approach as a base for comparisons and the novel symbolic approach. It can calculate a finite complete prefix of the low-level unfolding for a given high-level Petri net, combining the concepts from [8] (complete finite prefixes) and [9] (generating the low-level unfolding without expansion). Additionally, it is capable of computing a complete finite prefix of the net's symbolic unfolding, utilizing a modified version of Alg. 1. Since we want to compare the low-level with the high-level case, we restricted ourselves to nets from the class $\mathbf{N_F}$ to guarantee that the low-level unfolding exists.

Both, the generalized (Alg. 1) and the original ([8]) ERV-algorithm create possible extensions that are structurally dependent cut-off events, whereas in the implementation a cut-off event never triggers the calculation of possible extension. With the same idea, conditions in the postset of cut-off events

are never considered for finding co-sets. This leaves the finite complete prefix unmodified, as it only eliminates unnecessary work.

More importantly, the tool operates on a modified unfolding since an implementation using the predicates defined here turned out to be very slow. It rewrites the predicates in the unfolding and modifies arc labels to drastically reduce the number of variables. We achieve this by associating with each condition just one variable, the *internal variable*. Every condition then has this internal variable on all outgoing edges, as on the ingoing edge. We choose the variable name such that it uniquely identifies the event that chooses the color of the token. We say an event *chooses* a color if its transition has a variable on an outgoing edge that is on no ingoing edge. This is in contrast to the case where the variable on the outgoing edge was also on an ingoing edge. In that case the event only forwards the color choice of a previous event. For example, in the unfolding from Figure 2b, COLORUNFOLDER replaces the four variables by a single variable.

As a consequence of this renaming, we might need to modify guards in order to preserve the behavior of the original net. A transition in the original net then can have two ingoing edges with the same variable, but the corresponding event has distinct internal variables in those positions. In that case we add a guard that requires equality of those internal variables. After a finite complete prefix of the modified unfolding is found, the result can be easily transformed into the expected result with barely any overhead.

This optimization yields a significant speed up. However, when working on the symbolic unfolding, in our experiments still more than 99 percent of the time is spent evaluating the satisfiability of predicates to identify cut-off events using Cor. 5.4, and to detect when to discard event candidates because of a color conflict. For this task we chose the CVC5 SMT solver [24]. It performed best in the relevant category (non-linear arithmetic with equality and quantifiers) of the Satisfiability Modulo Theories Competition 2023 (SMT-COMP 2023)[2].

6.2. Benchmark families

In this section, we present four new benchmark families on which we tested the calculation (resp. verification on) of the symbolic unfolding and compared it to the calculation of (resp. verification on) the low-level unfolding.

6.2.1. Fork And Join

The simplest of our benchmark families is called *Fork And Join*. In the initial marking, a token lies on place p_0. A transition t takes this token from p and places an arbitrary color on each of its output places. A transition ε then takes these colors from all places, ending the nets execution. We have two parameters: the first parameter, $m \in \mathbb{N}$, determines the set of colors $Col = \{0, \ldots, m\}$. The second parameter, $n \in \mathbb{N}$, determines the number of output places of t. Fig. 5 shows the independent diamonds for $n = 2$ in (a) and for $n = 4$ in (b).

---

[2]`https://smt-comp.github.io/2023/results/equality-nonlineararith-single-query`
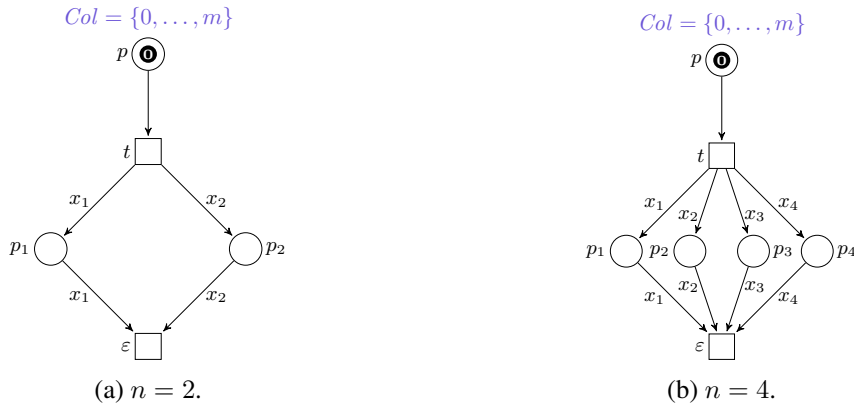
Figure 5: Fork And Join for $n = 2$ in (a) and $n = 4$ in (b).

The symbolic unfolding of a Fork And Join has $n + 3$ nodes as it is structurally equal to the net itself. The low-level unfolding of the expansion has $(n + 2)(m + 1)^n + 1$ nodes (since t is fireable in $(m + 1)^n$ modes), cp. App. A.2.

6.2.2. The water pouring puzzle

This benchmark family generalizes the following logic puzzle (cf., e.g., [25]):

"You have an infinite supply of water and two buckets. One holds 5 liters, the other holds 3 liters. Measure exactly 4 liters of water in one bucket."

In our generalization we have two parameters. The first parameter is a finite list $n = [n_1, \ldots, n_k]$ of natural numbers. Each entry n_i represents an available bucket holding n_i liters. The second parameter, $m \in \mathbb{N}$ is the amount of water that should be measured.
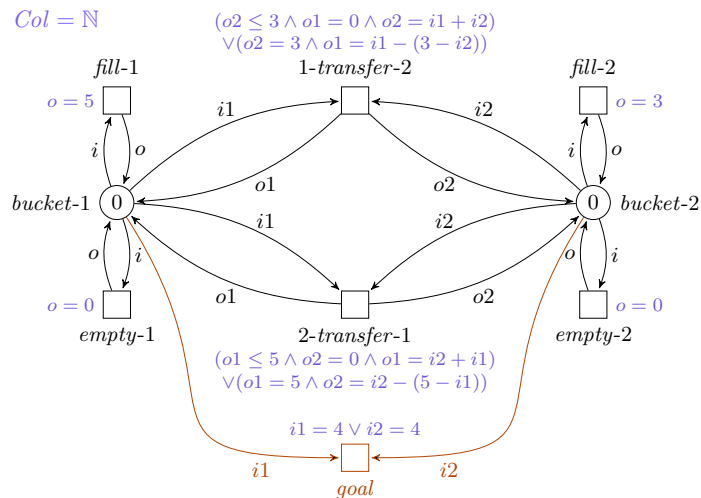

Figure 6: The Water Pouring Puzzle with 2 buckets holding 5 resp. 3 liters.

Fig. 6 shows the high-level Petri net corresponding to the parameters $n = [3, 5]$ and $m = 4$, corresponding to the puzzle above. Independently of the parameters, we have $Col = \mathbb{N}$. The current fill level of each bucket i is represented by a place $bucket\text{-}i$, with an initial color 0, and two attached transitions $fill\text{-}i$ and $empty\text{-}i$, that, when fired, replace the color on $bucket\text{-}i$ by n_i or 0, respectively. Additionally, for each pair of buckets i, j, there are two transitions $i\text{-}transfer\text{-}j$ and $j\text{-}transfer\text{-}i$ that transfer as much water as possible from one bucket to the other without overflowing it. When at least one bucket contains m liters, the goal transition can fire. We include a prefix the symbolic unfolding of the net from Fig. 6 in App. A.2.

6.2.3. Hobbits And Orcs

The *Hobbits And Orcs* problem[3] is another logic puzzle (in particular one of many "river crossing problems", cf., e.g., [26, 27]) and goes as follows:

> "Three Hobbits and three Orcs must cross a river using a boat which can carry at most two passengers. For both river banks, if there are Hobbits present on the bank, they cannot be outnumbered by orcs (if they were, the Orcs would attack the Hobbits). The boat cannot cross the river by itself with no one on board."

We generalize this problem by introducing two parameters. The first parameter, $m \in \mathbb{N}$ is the number of both Hobbits and Orcs. We always have equally many of the two parties. The second parameter, $n \in \mathbb{N}$ is the number of passengers the boat can carry. We additionally assume that also on the boat, if there are Hobbits present, they cannot be outnumbered by Orcs.

Figure 7a shows an illustration of the original puzzle presented above, with three of both, Hobbits and Orcs, and a boat fitting two passengers. Figure 7b shows the corresponding high-level Petri game. The colors are given by $Col = \mathbb{N} \times \mathbb{N}$, where a tuples describes the number of Hobbits and Orcs at a location – the left bank, the boat, or the right bank. We start with three Hobbits and three Orcs on the left bank, indicated by the tuple $(3, 3)$ on the place $bank\text{-}\ell$. The four center places describe the current state of the boat, being either empty and docked on a bank, or loaded and on the river. Initially, there is a tuple $(0, 0)$ on $docked\text{-}\ell$, indicating an empty boat the left bank. Via the transitions $load$ and $unload$ left and right, the boat can be loaded or unloaded, with the guards ensuring all the conditions from the riddle regarding the number of Hobbits and Orcs on both banks and on the boat. When all creatures are on the right bank, the goal transition can fire, ending the net's execution.

6.2.4. Mastermind

The last benchmark family models a generalization of the classic code-breaking game *Mastermind* developed in the early Seventies and completely solved in 1993 [28, 29]. The game is played between two players. The *code maker* secretly chooses an ordered, four digit color code, with six available colors. The *code breaker* then guesses the code. The code maker evaluates the guess by a number of *red pins* indicating how many colors in the guess are in the correct position, and a number of *white*

---

[3]The problem is also known as "Missionaries and Cannibals", and is a variation of the "Jealous Husbands" problem.
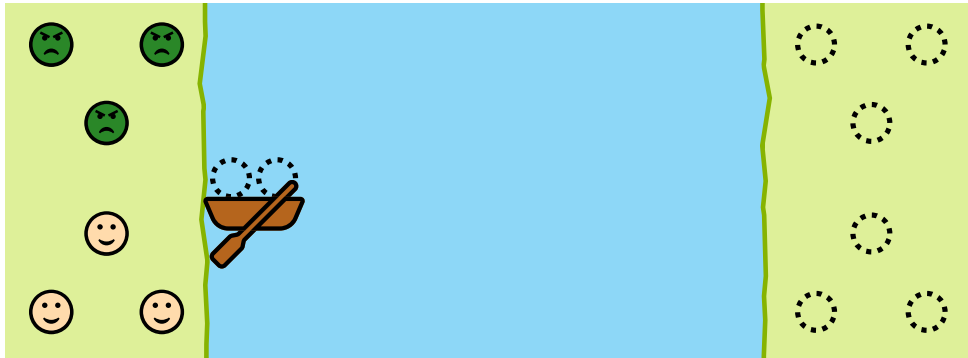
(a) An illustration of the "Hobbits And Orcs" problem.[4]
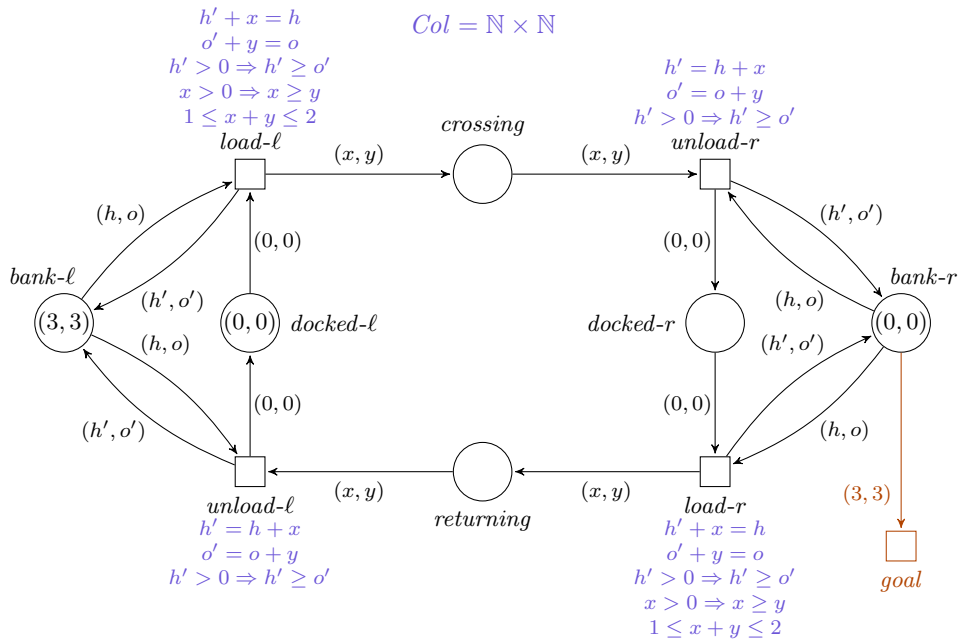
(b) The problem modeled as a high-level Petri net.

Figure 7: The Hobbits And Orcs problem with 3 Hobbits, 3 Orcs, and a boat fitting 2 passengers.

pins indicating how many colors in the guess appear at a different position in the code. Using this knowledge, the code breaker makes the next guess, with up to twelve attempts.

In our generalization we have three parameters. The first parameter, $m \in \mathbb{N}$, describes the number of available colors. The second parameter, $n \in \mathbb{N}$, describes the length of the code. The third pa-

---

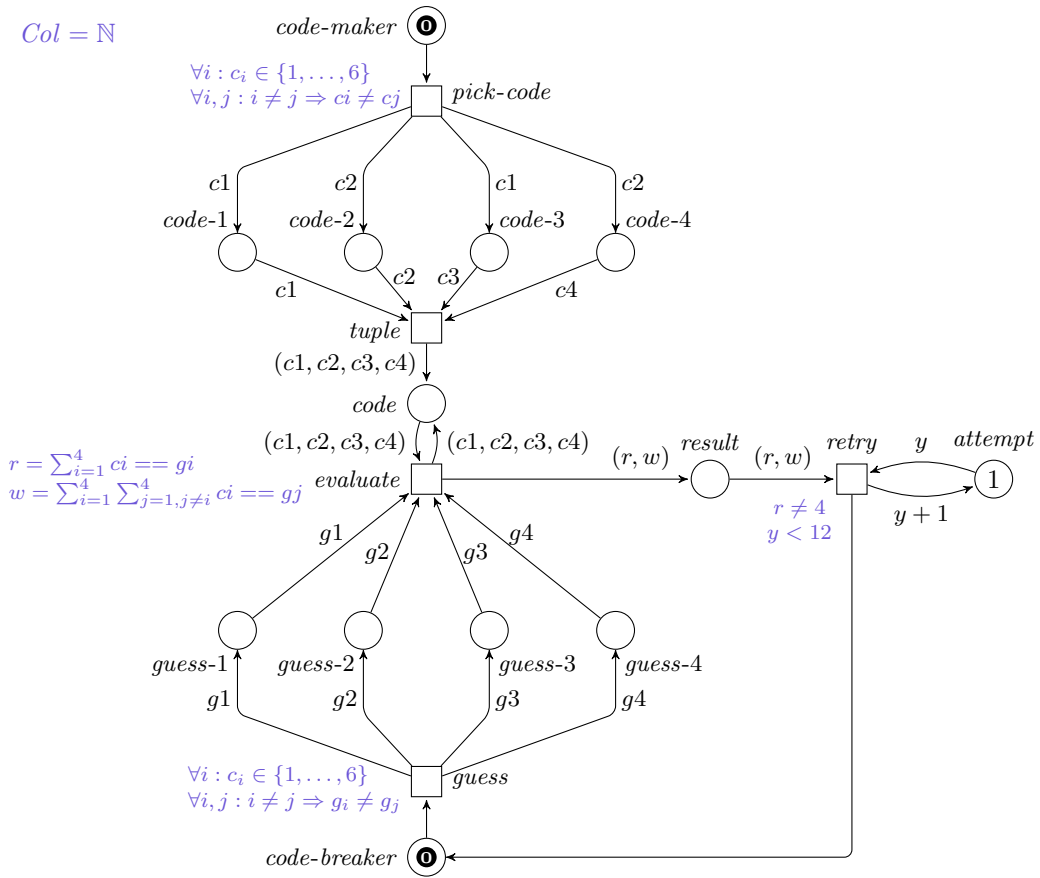[4]The "Boat" icon used in Figure 7a is by DinosoftLabs from Noun Project, `https://thenounproject.com/dinosoftlab/`.

Figure 8: The Mastermind game with a code of length 4, 6 possible colors, and 12 attempts.

rameter, $k \in \mathbb{N}$, describes the number of guesses the code breaker can make. To simplify the net, we restricted the allowed codes to not contain any color twice.

Fig. 8 shows the high-level Petri net with $m = 6$ available colors, a code of length $n = 4$, and $k = 12$ possible attempts, i.e., the scenario described above. The code maker places one color on each of the four places *code-i* by firing *pick-code*. Transition *tuple* puts these colors into a tuple on place *code*. The purpose of this (for the model unimportant) transition is to make the symbolic unfolding (cp. App. A.2) resemble an actual game board of Mastermind. The code breaker, analogously to the code maker, concurrently guesses a code via transition *guess*. Transition *evaluate* compares this guess to the code and places the result, i.e., the corresponding number of red and white pins, on *result*. From there, the code breaker either wins if the guessed code was correct, or resets with transition *retry*, provided that he has another attempt left.

6.3. Mode-deterministic high-level Petri nets

In the experiments presented in the next section we identified an important indicator for whether the symbolic approach for a finite complete prefix presented in this paper is expected to outperform, the complete finite prefix of the low-level unfolding, combining the concepts of [8] and [9].

The identified net property is that in every reachable marking of N, every transition can fire in at most one mode. We call a high-level Petri net with this property *mode-deterministic*, formally:

Definition 6.1. A high-level Petri net N with transitions T is called *mode-deterministic* iff

$$\forall M \in \mathcal{R}(N) \, \forall t \in T \, \exists^{\leq 1} \sigma \in \Sigma(t) : M[t, \sigma\rangle.$$

In the case of a mode-deterministic net N, the *skeleton* of N's symbolic unfolding (essentially, the core structure of the high-level occurrence net, devoid of arc labels and guards, and interpreted as a P/T Petri net) is equivalent in structure to the low-level unfolding of N's expansion. This implies that the high-level abstraction does not offer any computational advantage in the unfolding process. To the best of our knowledge, this property has not been studied elsewhere.

We borrow terminology from "regular" determinism and say that a high-level Petri net is *mode-nondeterministic* if it does not satisfy the above property, and implicitly describe by a high or low "degree" of mode-nondeterminism that there are many resp. few transition-mode combinations making the net mode-nondeterministic.

An illustrative example of a family of mode-deterministic nets is the benchmark family Water Pouring Puzzle introduced in Sec. 6.2.2. Although this property may not be immediately apparent, it is true that in every state of the system, all transitions can fire in at most one mode. Specifically, transitions *fill-i* and *fill-i* replace the color on *bucket-i* with a predetermined one. Every transition *i-transfer-j* emulates the transfer of *all* water from bucket i that can fit into bucket j.

A condition that guarantees N to be a mode-deterministic net is as follows: "For any conceivable assignment of variables on input arcs, there exists at most one possible mode that completes this assignment to all variables." The nets in the Water Pouring Puzzle family fulfill this criterion, since all output variables (o) are either predetermined or derived from the input variables (i).

An example for a highly mode-nondeterministic nets is the benchmark family Fork And Join from Sec. 6.2.1. There are only two transitions in the net, but from the initial marking, t can fire in *every* of its m^n modes. This structural pattern of Fork And Join can also be observed in the Mastermind benchmark family (Sec. 6.2.4), making these nets also mode-nondeterministic.

The Hobbits and Orcs family (Sec. 6.2.3), on the other hand, falls in between and is notably contingent on the parameters involved. When there are only two seats on the boat, there are merely five *potential* modes of *load-ℓ* from the initial marking (reflecting the possible ways to occupy one or both seats with two types of creatures). However, in the scenario where there are n seats on the boat, along with m Hobbits and Orcs, and $m > n$, the total number of possibilities is $\sum_{i=1}^{n} i + 1 = \frac{1}{2}n(n + 3)$. It's worth noting that approximately half of these possibilities would result in more orcs than hobbits on the boat, rendering them non-modes. Hence, given the condition $m > n$, the "degree" of mode-determinism remains unaffected by an increase in m, but solely varies with the parameter n.

In the subsequent section, we empirically validate the hypothesis that the level of mode-determinism is inversely proportional to the benefit gained from employing symbolic unfolding compared to low-level unfolding.

6.4.   Experiments and results

We now present the experimental results of applying our implementation to the four benchmark families presented above. COLORUNFOLDER can check the reachability of a (set of) marking(s) by adding a respective transition to the net. It then executes Alg. 1 but stops when an instance of this transition is added to the prefix under construction, which means that the respective (set of) marking(s) is/are reachable. When the algorithm terminates without such an instance, the completeness of the prefix implies that the marking is not reachable.

For the nets from the Fork And Join benchmark family, the complete unfolding (being its own smallest finite and complete prefix) is calculated. For the benchmark families Water Pouring Puzzle and Hobbits And Orcs, the reachability of the goal state of the riddle is checked. This is done by adding the respective *goal* transition that is depicted in each of the two figures Fig. 6 and Fig. 7. This transition is not part of the input net. Finally, for nets in the Mastermind benchmark family, reachability of a marking with the result of $n-1$ red pins and 1 white pin is checked. Such a marking is never reachable, so always the complete (but finite) unfolding is calculated.

The tasks described above are checked twice, once using the symbolic unfolding and once using the low-level unfolding as described in Sec. 6.1. The experiments are calculated with commit 124b1735 of COLORUNFOLDER [23] on an otherwise idle system with an Intel i7-6700K CPU at 4.0 GHz and 16 GB RAM.
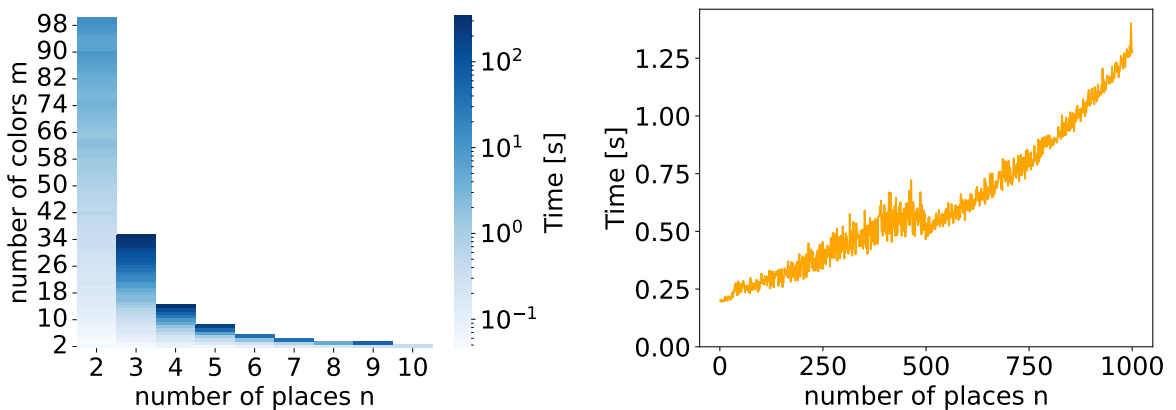
(a) Low-level, for growing number of places on x-axis, color class on y-axis and time as heatmap intensity.

(b) Symbolic, for growing number of places on x-axis and time on y-axis. Behavior is independent of the color class. This graph is for $Col = \mathbb{N}$.

Figure 9: Time needed to calculate unfolding (a) and symbolic unfolding (b) of Fork & Join.

Fork & Join. The results for Fork & Join are shown in Fig. 9.

In Fig. 9a, the elapsed time for calculating the low-level unfolding for the instance with n places (x-axis) and m colors (y-axis) is indicated by the heatmap intensity of the respective cell. Empty cells exceeded a 5-minute timeout. We see that the low-level approach is only viable if at least one parameter is very small. It takes exponentially more time with a growing color class and with growing number of places.

Since the symbolic approach outperforms the low-level approach by a wide margin, we present it in its own figure, Fig. 9b. For all the cases from Fig. 9a, the symbolic approach takes around 200 ms to complete the symbolic unfolding. This is independent of the color class $\{0, \ldots, m\}$. This corresponds to the fact that the symbolic unfolding of Fork & Join is, for a fixed number of places n, independent of the color class $Col = \{0, \ldots, m\}$. This even does not change for $Col = \mathbb{N}$. Since cvc5 (the tool used for checking satisfiability, cp. Sec. 6.1) can handle such infinite color domains, we fix $Col = \mathbb{N}$, and Fig. 9b shows the elapsed time (y-axis) for calculating the symbolic unfolding of the Fork & Join instance with n places (x-axis). This approach is very fast (but not linear in the number of places).

The symbolic approach is faster for all choices of the parameters. Only for the smallest choices are both approaches equally fast within the margin of error. This behavior was expected since the Petri nets are "highly non mode-deterministic", as explained above, and the low-level unfolding is much broader than the symbolic unfolding.

Water Pouring Puzzle. For the Water Pouring Puzzle we cannot get interesting results by varying one parameter while holing the others fixed, because the complexity of the solution is highly volatile and dependent on the combination of all parameters. Since the nets are mode-deterministic, the low-level and symbolic unfolding, as well as their prefixes, are isomorphic.

Table 1: Results of the Water Pouring Puzzle benchmark.

buckets n	target m	solvable	$|B|$	$|E|$	time low-level	time symbolic
$3, 5$	4	yes, 6 steps	90	75	**95 ms**	1.2 s
$15, 17$	10	yes, 18 steps	258	195	**220 ms**	36 s
$57, 73$	51	yes, 92 steps	1294	935	**1.7 s**	> 1 h
$9, 12$	4	no	106	74	**130 ms**	1.8 s
$10, 16$	5	no	190	134	**140 ms**	11 s
$8, 14, 17$	2	yes, 4 steps	411	642	**300 ms**	26 s
$14, 26, 27$	14	yes, 15 steps	21635	15279	**7.7 s**	> 1 h
$12, 15, 18$	10	no	2391	1442	**550 ms**	20 min
$12, 21, 27$	8	no	4029	2444	**1.0 s**	88 min

Table 1 presents the results for this benchmark. The original puzzle with $n = [3, 5]$ and $m = 4$ is included in the first line. Additionally, we randomly selected eight parameter sets such that we consider four scenarios involving 2 buckets and four scenarios involving 3 buckets.

We indicate in each scenario whether the puzzle is solvable, and in the positive case, how many steps a minimal solution has. We compare the time to check the reachability of a goal state (fireability of the *goal* transition, cp. above) of the low-level and symbolic approach. The faster approach is highlighted with bold font. Additionally, we notate the number of conditions ($|B|$) and events ($|E|$) of the generated prefix. Each of these two numbers coincides between the two approaches.

When we pick the parameters at random the low-level approach generally is much faster. We can, however, construct examples in which the symbolic approach outspeeds the low-level approach. In these constructed cases we have a large color domain but can reach the goal quite quickly, e.g., with two buckets of capacity 10^6 and $10^6 + 1$, and a target of $m = 1$.

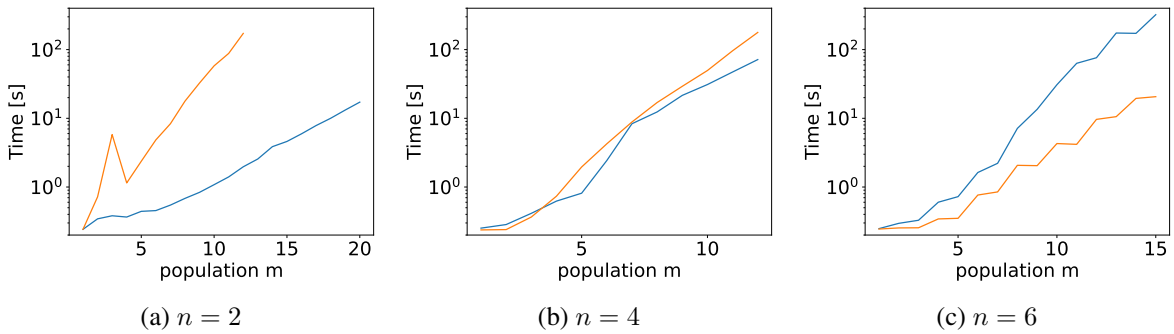

(a) $n = 2$ (b) $n = 4$ (c) $n = 6$

Figure 10: Results for Hobbits And Orcs problem for a fixed boat capacity n in (a), (b), (c), and population with m of each Hobbits and Orcs on the x-axis. The orange line shows the time needed by the symbolic approach and the blue line shows the low-level approach.

Hobbits And Orcs. For the Hobbits And Orcs benchmark we vary the and the capacity n of the boat fixed in Figures 10a, 10b, 10c, and number the of each Hobbits and Orcs, m, plotted on the x-axis. We indicate the time needed by the low-level approach in blue and by the symbolic approach in orange, with a timeout of 3 minutes.

We find that the low-level approach performs better when the boat capacity is smaller. For $n = 2$, the symbolic approach gets a timeout at $m = 13$ while the low-level approach can calculate the prefix up to $m = 33$. We expected this behavior, because a smaller the boat capacity yields a higher degree of mode-determinism in the net. When the capacity increases, the net is 'less' mode-deterministic. For $n \geq 6$ the symbolic approach is faster for all population sizes. Independent of the boat size n, both approaches take exponentially more time with growing population size.

Mastermind. We present some results for the Mastermind benchmark in Table 2. We compare the time needed by the low-level and symbolic approach to generate the unfolding, with a timeout of two minutes. The table also shows the number of nodes (conditions $|B|$ resp. $|B|$, and events $|E|$ resp. $|E|$) in the unfolding. In each row the faster time is highlighted in bold font.

Table 2: Results of the Mastermind benchmark.

k	n	m	Low-level			Symbolic		
			time	$|B|$	$|E|$	time	$|B|$	$|E|$
12	4	6	$> 2\,\mathrm{min}$	-	-	**1 min**	149	36
1	3	3	$86\,\mathrm{ms}$	219	48	**62 ms**	14	3
1	3	5	$1.8\,\mathrm{s}$	18363	3720	**54 ms**	14	3
1	3	6	$18\,\mathrm{s}$	72723	14640	**61 ms**	14	3
1	3	7	$> 2\,\mathrm{min}$	-	-	**54 ms**	14	3
1	3	1000	$> 2\,\mathrm{min}$	-	-	**53 ms**	14	3
1	4	4	$1\,\mathrm{s}$	3651	624	**60 ms**	17	3
1	4	5	$> 2\,\mathrm{min}$	-	-	**67 ms**	17	3
1	5	1000	$> 2\,\mathrm{min}$	-	-	**77 ms**	20	3
1	28	1000	$> 2\,\mathrm{min}$	-	-	**1 s**	89	3
2	3	3	$215\,\mathrm{ms}$	1719	438	**206 ms**	24	6
2	3	4	$3.3\,\mathrm{s}$	110115	27672	**120 ms**	24	6
2	3	5	$111\,\mathrm{s}$	1726443	432060	**124 ms**	24	6
2	3	6	$> 2\,\mathrm{min}$	-	-	**119 ms**	24	6
2	3	1000	$> 2\,\mathrm{min}$	-	-	**127 ms**	29	6
2	4	4	$5.5\,\mathrm{s}$	137235	27672	**1.3 s**	29	6
2	4	5	$> 2\,\mathrm{min}$	-	-	**116 ms**	29	6
2	4	1000	$> 2\,\mathrm{min}$	-	-	**162 ms**	29	6
2	10	1000	$> 2\,\mathrm{min}$	-	-	**1 s**	59	6
3	4	4	$> 2\,\mathrm{min}$	-	-	**35 s**	41	9
4	3	3	$2.6\,\mathrm{s}$	46719	12138	**803 ms**	44	12
5	3	3	$39\,\mathrm{s}$	234219	60888	**1.3 s**	54	15
6	3	3	$> 2\,\mathrm{min}$	-	-	**1.8 s**	64	18

The first line shows the original problem with $k = 12$ attempts, a code length of $n = 4$, and $m = 6$ colors. We observe that in the low-level approach, time and size grow exponentially with respect to the number m of colors, whereas the high-level approach remains constant. When we increase the parameter n controlling the length of the code, both approaches grow exponentially in time. Interestingly the high-level approach only grows linear in *size* (number of nodes). This is due to the size of the guard of the *evaluate* transition increasing exponentially (n choose 2).

Overall, the symbolic approach is the clear winner of the Mastermind benchmark. The low-level approach can only barely compete for the smallest instances of the problem.

We also observe that, in comparison to other parameters, the performance of the high-level approach drops when $n = m$. We currently have no explanation for this. This could be a quirk of the SMT solvers, or alternatively, the formulas may be inherently more challenging to solve in this particular case. The low-level approach is not impacted negatively by this case, but still much slower than the symbolic approach, as seen in lines with parameters $(1, 3, 3)$, $(1, 4, 4)$, $(2, 3, 3)$, $(2, 4, 4)$, and in the last four lines.

7. Conclusions and outlook

We introduced the notion of complete finite prefixes of symbolic unfoldings of high-level Petri nets. We identified a class of 1-safe high-level nets generalizing 1-safe P/T nets, for which we generalized the well-known algorithm by Esparza et al. to compute such a finite and complete prefix. This constitutes a consolidation and generalization of the concepts of [8, 13, 4, 11]. While the resulting symbolic prefix has the same depth as a finite and complete prefix of the unfolding of the represented P/T net, it can be significantly smaller due to less branching. In the case of infinitely many reachable markings (where the original algorithm is not applicable) we identified the class of so-called *symbolically compact* nets for which an adapted version of the generalized algorithm effectively computes a finite complete prefix of the symbolic unfolding. For that, we showed how to check an adapted cut-off criterion by symbolically describing sets of markings. We implemented the generalized algorithm and tested it against four novel benchmark families. This experimentation validated an indicator for whether the symbolic approach is expected to outperform the low-level approach. This indicator relies on the concept of a net property we call "mode-determinism".

Future works include the construction of a symbolic reachability graph for symbolically compact nets and a comparison with the complete finite prefix, as outlined in Sec. 4.4. Additionally, a generalization for k-bounded high-level Petri nets seems possible. Furthermore, we want to apply the results to *high-level Petri games* [30, 31] to find "symbolic strategies" with techniques similar to [32] or [33, 34], employing a successively increasing bound on size of the considered prefix of the symbolic unfolding.

References

[1] Würdemann N, Chatain T, Haar S. Taking Complete Finite Prefixes to High Level, Symbolically. In: Proc. PETRI NETS 2023, LNCS 13929. Springer, 2023 pp. 123–144. doi:10.1007/978-3-031-33620-1_7.

[2] Reisig W. Understanding Petri Nets - Modeling Techniques, Analysis Methods, Case Studies. Springer, 2013. ISBN 978-3-642-33277-7. doi:10.1007/978-3-642-33278-4.

[3] Jensen K. Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use - Volume 1, Second Edition. Monographs in Theoretical Computer Science. An EATCS Series. Springer, 1996. ISBN 978-3-642-08243-6. doi:10.1007/978-3-662-03241-1.

[4] Chatain T, Fabre E. Factorization Properties of Symbolic Unfoldings of Colored Petri Nets. In: Lilius J, Penczek W (eds.), Proc. PETRI NETS 2010, LNCS 6128. Springer, 2010 pp. 165–184. doi:10.1007/978-3-642-13675-7_11.

[5] Nielsen M, Plotkin GD, Winskel G. Petri Nets, Event Structures and Domains, Part I. *Theor. Comput. Sci.*, 1981. **13**:85–108. doi:10.1016/0304-3975(81)90112-2.

[6] Engelfriet J. Branching Processes of Petri Nets. *Acta Informatica*, 1991. **28**(6):575–591. doi:10.1007/BF01463946.

[7] McMillan KL. A Technique of State Space Search Based on Unfolding. *Formal Methods Syst. Des.*, 1995. **6**(1):45–65. doi:10.1007/BF01384314.

[8] Esparza J, Römer S, Vogler W. An Improvement of McMillan's Unfolding Algorithm. *Formal Methods Syst. Des.*, 2002. **20**(3):285–310. doi:10.1023/A:1014746130920.

[9] Khomenko V, Koutny M. Branching Processes of High-Level Petri Nets. In: Proc. TACAS 2003, LNCS 2619. Springer, 2003 pp. 458–472. doi:10.1007/3-540-36577-X_34.

[10] Ehrig H, Hoffmann K, Padberg J, Baldan P, Heckel R. High-Level Net Processes. In: Formal and Natural Computing, LNCS 2300. Springer, 2002 pp. 191–219. doi:10.1007/3-540-45711-9_12.

[11] Chatain T, Jard C. Symbolic Diagnosis of Partially Observable Concurrent Systems. In: Proc. FORTE 2004, LNCS 3235. Springer, 2004 pp. 326–342. doi:10.1007/978-3-540-30232-2_21.

[12] Chatain T, Jard C. Complete Finite Prefixes of Symbolic Unfoldings of Safe Time Petri Nets. In: Proc. ICATPN 2006, LNCS 4024. Springer, 2006 pp. 125–145. doi:10.1007/11767589_8.

[13] Chatain T. Symbolic Unfoldings of High-Level Petri Nets and Application to Supervision of Distributed Systems. Ph.D. thesis, Universit é de Rennes, 2006. URL https://www.sudoc.fr/246936924.

[14] Presburger M. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In: Proc. Comptes-rendus du I Congrés des Mathématiciens des Pays Slaves, Varsovie 1929. 1930 pp. 92–101.

[15] Abdulla PA, Iyer SP, Nylén A. Unfoldings of Unbounded Petri Nets. In: Proc. CAV 2000, LNCS 1855. Springer, 2000 pp. 495–507. doi:10.1007/10722167_37.

[16] Desel J, Juhás G, Neumair C. Finite Unfoldings of Unbounded Petri Nets. In: Proc. PETRI NETS 2004, LNCS 3099. Springer, 2004 pp. 157–176. doi:10.1007/978-3-540-27793-4_10.

[17] Herbreteau F, Sutre G, Tran TQ. Unfolding Concurrent Well-Structured Transition Systems. In: Proc. TACAS 2007, LNCS 4424. Springer, 2007 pp. 706–720. doi:10.1007/978-3-540-71209-1_55.

[18] Schmidt K. Parameterized Reachability Trees for Algebraic Petri Nets. In: Proc. PETRI NETS 1995, LNCS 935. Springer, 1995 pp. 392–411. doi:10.1007/3-540-60029-9_51.

[19] Schwoon S. MOLE. URL http://www.lsv.ens-cachan.fr/~schwoon/tools/mole/.

[20] Rodríguez C. CUNF. URL https://github.com/cesaro/cunf.

[21] Rodríguez C, Schwoon S. Cunf: A Tool for Unfolding and Verifying Petri Nets with Read Arcs. In: Proc. ATVA 2013, volume 8172 of *LNCS*. Springer, 2013 pp. 492–495. doi:10.1007/978-3-319-02444-8_42.

[22] Khomenko V. PUNF. URL homepages.cs.ncl.ac.uk/victor.khomenko/tools/punf/.

[23] Panneke L. COLORUNFOLDER. URL https://github.com/Selebrator/ColorUnfolder.

[24] Barbosa H, Barrett CW, Brain M, Kremer G, Lachnitt H, Mann M, Mohamed A, Mohamed M, Niemetz A, Nötzli A, Ozdemir A, Preiner M, Reynolds A, Sheng Y, Tinelli C, Zohar Y. cvc5: A Versatile and Industrial-Strength SMT Solver. In: Fisman D, Rosu G (eds.), Proc. TACAS 2022, LNCS 13243. Springer, 2022 pp. 415–442. doi:10.1007/978-3-030-99524-9_24.

[25] Atwood ME, Polson PG. A process model for water jug problems. *Cognitive Psychology*, 1976. **8**(2):191–216. doi:https://doi.org/10.1016/0010-0285(76)90023-2.

[26] Jeffries R, Polson PG, Razran L, Atwood ME. A process model for Missionaries-Cannibals and other river-crossing problems. *Cognitive Psychology*, 1977. **9**(4):412–440. doi:https://doi.org/10.1016/0010-0285(77)90015-9.

[27] Pressman I, Singmaster D. The jealous husbands and The missionaries and cannibals. *The Mathematical Gazette*, 1989. **73**(464):73–81. doi:10.2307/3619658.

[28] Knuth DE. The Computer as Master Mind. *J. Recreational Mathematics*, 1976–77. **9**(1):1–6.

[29] Koyama K, Lai TW. An optimal Mastermind Strategy. *J. Recreational Mathematics*, 1993. **25**(4):251–256.

[30] Gieseking M, Olderog E. High-Level Representation of Benchmark Families for Petri Games. In: Model Checking, Synthesis, and Learning, LNCS 13030. Springer, 2021 pp. 115–137. doi:10.1007/978-3-030-91384-7_7.

[31] Gieseking M, Olderog E, Würdemann N. Solving high-level Petri games. *Acta Informatica*, 2020. **57**(3-5):591–626. doi:10.1007/s00236-020-00368-5.

[32] Finkbeiner B. Bounded Synthesis for Petri Games. In: Correct System Design, LNCS 9360. Springer, 2015 pp. 223–237. doi:10.1007/978-3-319-23506-6_15.

[33] Chatain T, Haar S, Jezequel L, Paulevé L, Schwoon S. Characterization of Reachable Attractors Using Petri Net Unfoldings. In: Proc. CMSB 2014, LNCS 8859. Springer, 2014 pp. 129–142. doi:10.1007/978-3-319-12982-2_10.

[34] Aguirre-Samboní GK, Haar S, Paulevé L, Schwoon S, Würdemann N. Avoid One's Doom: Finding Cliff-Edge Configurations in Petri Nets. In: Proc. GandALF 2022, EPTCS 370. 2022 pp. 178–193. doi:10.4204/EPTCS.370.12.

A. Appendix

A.1. Examples of adequate orders

We show that the adequate order used in [7], as well as the orders $\prec_E$ and $\prec_F$ treated in [8], when lifted to the symbolic unfolding, are still adequate orders. In particular we show that $\prec_F$ is a total adequate order on the symbolic unfolding, limiting the size of the later constructed finite prefix. The definition of these orders does not change, so we take most of the following notation directly from [8].

The orders $\prec_M$ and $\prec_E$. The order $\prec_M$ used in [7] is defined by $C_1 \prec_M C_2 :\Leftrightarrow |C_1| < |C_2|$. It is trivial to see that $\prec_M$ satisfies i) and ii) from Def. 2.5. Since $\varphi_{1,D}^2$ is a injective, we have $|\varphi_{1,D}^2(D)| = |D|$, which yields iii).

For a high-level Petri net N, let $\ll$ be an arbitrary total order on the transitions of N. Given a set E' of events in the unfolding of N, let $\mathbf{p}(E')$ be that sequence of transitions which is ordered according to $\ll$, and contains each transition t as often as there are events in E' with label t. We say $\mathbf{p}(E_1) \ll \mathbf{p}(E_2)$ if $\mathbf{p}(E_1)$ is lexicographically smaller than $\mathbf{p}(E_2)$ with respect to the order $\ll$.

The order $\prec_E$ is then defined as follows: let C_1, C_2 be two configurations of the symbolic unfoldings of a high-level Petri net. $C_1 \prec_E C_2$ holds if either $|C_1| < |C_2|$, or $|C_1| = |C_2|$ and $\mathbf{p}(C_1) \ll \mathbf{p}(C_2)$. The proof that $\prec_E$ is an adequate order works exactly as in [8]:

It is easy to show that $\prec_E$ is a well-founded partial order implied by inclusion. We now show that $\prec_E$ is preserved by finite extensions. As already mentioned above, $|D| = |\varphi^2_{1,D}(D)|$. Additionally, we have $\mathbf{p}(D) = \mathbf{p}(\varphi^2_{1,D}(D))$, since $\varphi^2_{1,D}$ preserves the labeling of events.

Assume $C_1 \prec_E C_2$. If $|C_1| < |C_2|$, then $|C_1 \oplus D| < |C_2 \oplus \varphi^2_{1,D}(D)|$. If $|C_1| = |C_2|$ and $\mathbf{p}(C_1) \ll \mathbf{p}(C_2)$, then $|C_1 \oplus D| = |C_2 \oplus \varphi^2_{1,D}(D)|$ and, by the properties of the lexicographic order, $\mathbf{p}(C_1 \oplus D) \ll \mathbf{p}(C_2 \oplus \varphi^2_{1,D}(D))$.

The Total Adequate Order $\prec_F$. The *Foata normal form FC* of a configuration C is obtained by starting with FC empty, and iteratively deleting the set $Min(C)$ from C and appending it to FC, until C is empty.

Given two configurations C_1, C_2, we can compare their Foata normal forms $FC_1 = C_{11} \ldots C_{1n_1}$ and $FC_2 = C_{21} \ldots C_{2n_2}$ with respect to the order $\ll$ by saying $FC_1 \ll FC_2$ if there exists $i \le i \le n_1$ such that $\mathbf{p}(C_{1j}) = \mathbf{p}(C_{2j})$ for every $1 \le j < i$, and $\mathbf{p}(C_{1i}) \ll \mathbf{p}(C_{2i})$.

Definition A.1. (Order $\prec_F$)
let C_1 and C_2 be two configurations of the symbolic unfolding of a high-level Petri net. $C_1 \prec_F C_2$ holds if

- $|C_1| < |C_2|$, or
- $|C_1| = |C_2|$ and $\mathbf{p}(C_1) \ll \mathbf{p}(C_2)$, or
- $\mathbf{p}(C_1) = \mathbf{p}(C_2)$ and $FC_1 \ll FC_2$.

We prove that $\prec_F$ is a total adequate order. In the proof, (a) – (c) are taken directly from [8], with small adaptations due to the high-level formalism. While the ideas from (d) also come directly from [8], we have work with the monomorphism $\varphi^2_{1,D}$ instead of the isomorphism I^2_1, and the new definition of adequate order. This is where the only deviation from [8] happens.

Let $\beta = (O, h)$ be the symbolic unfolding of $N = (\mathcal{N}, \mathcal{M}_0)$.

(a) $\prec_F$ is a well-founded partial order.

This follows immediately from the fact that $\prec_E$ is a well-founded partial order as is the lexicographic order on transition sequences of some fixed length.

(b) $C_1 \subset C_2$ implies $C_1 \prec_F C_2$.

This is obvious, since $C_1 \subset C_2$ implies $|C_1| < |C_2|$.

(c) $\prec_F$ is total.

Assume that C_1 and C_2 are two incomparable configurations under $\prec_F$, i.e., $|C_1| = |C_2|$, $\mathbf{p}(C_1) = \mathbf{p}(C_2)$, and $\mathbf{p}(FC_1) = \mathbf{p}(FC_2)$. We prove $C_1 = C_2$ by induction on the common size $k = |C_1| = |C_2|$.

The base case $k = 0$ gives $C_1 = C_2 = \emptyset$, so assume $k > 0$.

We first prove $Min(C_1) = Min(C_2)$. Aiming a contradiction, assume w.l.o.g. that $e_1 \in Min(C_1) \setminus Min(C_2)$. Since $\mathbf{p}(Min(C_1)) = \mathbf{p}(Min(C_2))$, $Min(C_2)$ contains an event e_2 s.t.

$h(e_1) = h(e_2)$. Since $\to Min(C_1)$ and $\to Min(C_2)$ are subsets of B_0, and all conditions of B_0 carry different labels, we have $\to e_1 = \to e_2$, and thus, $pre(e_1) = pre(e_2)$. This contradicts the definition of symbolic branching processes.

Since $Min(C_1) = Min(C_2)$, both $C_1 \setminus Min(C_1)$ and $C_2 \setminus Min(C_1)$ are configurations of the branching process $\Uparrow Min(C_1)$ of $(\mathcal{N}, \mathcal{M}(Min(C_1)))$, and they are incomparable under $\prec_F$ by construction. Since the common size of $C_1 \setminus Min(C_1)$ and $C_2 \setminus Min(C_1)$ is strictly smaller than k, we can apply the induction hypothesis and conclude $C_1 = C_2$.

(d) $\prec_F$ is preserved by finite extensions.

Take two finite configurations C_1 and C_2, let D be a finite suffix of C_1, and let $M \in \mathcal{M}(C_1) \cap \mathcal{M}(C_2)$ such that $C_1 \llbracket M \rrbracket D$. We have to show that $C_1 \prec C_2$ implies $C_1 \oplus D \prec C_2 \oplus \varphi_{1,D}^2(D)$.

First, notice that we can assume $D = \{e\}$: For $e \in Min(D)$ we have from $C_1 \llbracket M \rrbracket D$ that $\exists \theta \in \Theta(C_1 \oplus D) : \text{mark}(C_1.\theta|_{Var_{C_1 \cup \{\bot\}}}) = M$. Thus, for M' s.t. $M[h(e).\sigma\rangle M'$ with $\sigma = \theta \circ [v \mapsto v_e]_{v \in Var(e)}$, we have that $M' \in \mathcal{M}(C_1 \oplus \{e\}) \cap \mathcal{M}(C_2 \oplus \{\varphi_{1,D}^2(e)\})$ and $(C_1 \oplus \{e\}) \llbracket M' \rrbracket (D \setminus \{e\})$.

Second, the cases $|C_1| < |C_2|$ and $C_1 \prec_E C_2$ in (i), (and the respective cases $|C_2| < |C_1|$ and $C_2 \prec_E C_1$ in (ii)) are easy (shown above). Hence, assume $|C_1| = |C_2|$ and $\mathbf{p}(C_1) = \mathbf{p}(C_2)$.

Third, we show that under these two assumptions e is a minimal event of $C_1' := C_1 \cup \{e\}$ if and only if $\varphi_{1,D}^2(e)$ is a minimal event of $C_2' := C_2 \cup \{\varphi_{1,D}^2(e)\}$. Let e be minimal in C_1', i.e., the transition $h(e)$ can be fired in a mode in one initial marking. Let $p \in \to h(e)$; then no condition in $\to C \cup C \to$ is labeled p, since these conditions would be concurrent to the p-labeled condition in $\to e$, contradicting that $(\mathcal{N}, \mathcal{M}_0)$ is safe. Thus, C_1 contains no event e' with $p \in \to h(e')$, and the same holds for C_2, since $\mathbf{p}(C_1) = \mathbf{p}(C_2)$. Therefore, the conditions in $\text{cut}(C_2)$ with label in $\to h(e)$ are minimal conditions of β, and $\varphi_{1,D}^2(e) = e$ is minimal in C_2'. The reverse implication holds analogously, since about C_1 and C_2 we have only used the hypothesis $\mathbf{p}(C_1) = \mathbf{p}(C_2)$.

With this knowledge, we now show the implication. Assume $C_1 \prec_F C_2$. We show $C_1' \prec_F C_2'$.

If $Min(C_1) \prec_E Min(C_2)$, then we now see $Min(C_1') \prec_E Min(C_2')$, hence $\mathbf{p}(FC_1') \ll \mathbf{p}(FC_2')$ and so we are done. If $\mathbf{p}(Min(C_1)) = \mathbf{p}(Min(C_2))$ and $e \in Min(C_1')$, then

$$C_1' \setminus Min(C_1') = C_1 \setminus Min(C_1) \prec_F C_2 \setminus Min(C_2) = C_2' \setminus Min(C_2'),$$

hence $C_1' \prec_F C_2'$. Finally, if $\mathbf{p}(Min(C_1)) = \mathbf{p}(Min(C_2))$ and $e \notin Min(C_1')$, we again argue that $Min(C_1) = Min(C_2)$ and that, hence, $C \setminus Min(C_1)$ and $C_2 \setminus Min(C_1)$ are configurations of the branching process $\Uparrow Min(C_1)$ of $(\mathcal{N}, \mathcal{M}(Min(C_1)))$. With an inductive argument we get $C_1' \setminus Min(C_1') \prec_F C_2' \setminus Min(C_2')$ and are also done in this case.

A.2.   More symbolic and low-level unfoldings

In this appendix we present unfoldings omitted in the main body of the paper.

Low-level Unfolding of Fork And Join.   Since the symbolic unfolding of any Fork And Join is structurally equal to the net itself we do not display it here. Fig. 11 shows the low-level unfolding with $(n + 2)(m + 1)^n + 1$ nodes for a Fork And Join.
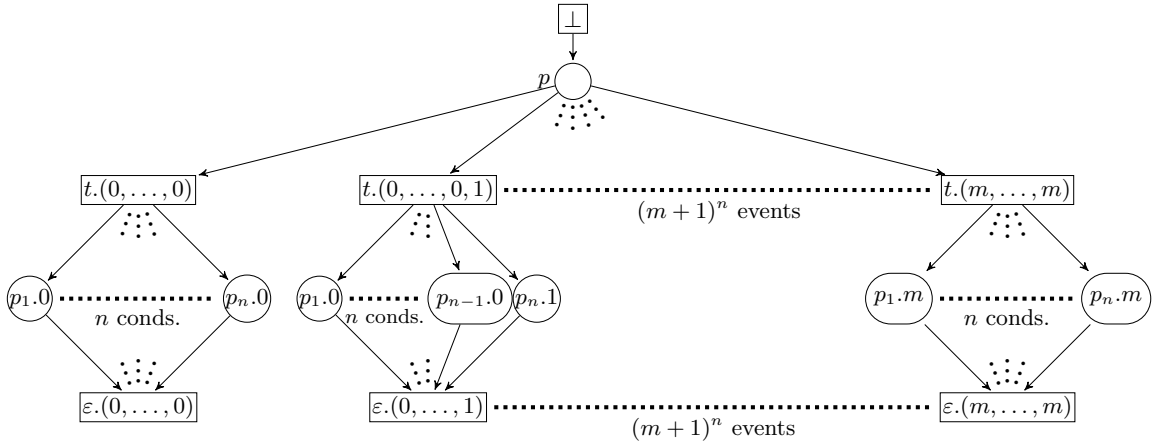

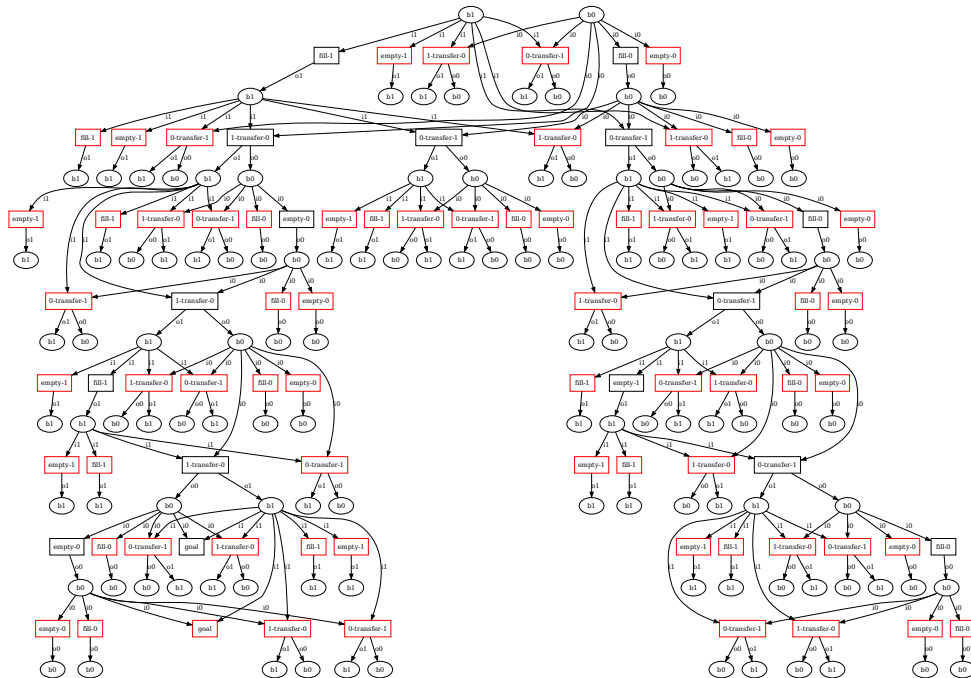Figure 11: The symbolic unfolding of a Fork And Join, depending on the parameters m and n.

Figure 12: A prefix of the symbolic unfolding of the Water Pouring Puzzle for $n = [3, 5]$ and $m = 4$, produced by COLORUNFOLDER.

Symbolic Unfolding of Water Pouring Puzzle. Figure 12 shows the complete finite prefix of the symbolic unfolding of the Water Pouring Puzzle from Fig. 6 with $n = [5, 3]$ and $m = 4$, calculated by our implementation COLORUNFOLDER [23] of the generalized ERV-algorithm, Alg. 1. The figure is automatically produced from the output of COLORUNFOLDER. Cut-off events are marked by a red border. Additionally marked by a red border are the events representing the added *goal* transition that checks the target states. Since the net is mode-deterministic, the symbolic unfolding's skeleton coincides with the low-level unfolding. Thus, the skeleton of the complete finite prefix in Figure 12 coincides with the complete finite prefix of the low-level unfolding generated by the original ERV-algorithm from [8].

Symbolic Unfolding of Mastermind. Figure 13 shows a prefix of the symbolic unfolding of the Mastermind net from Fig. 8 with 6 available colors, a code of length 4, and 12 possible attempts. Combinatorial arguments give that the low-level unfolding has more than 10^{37} nodes, so we do not present it here.
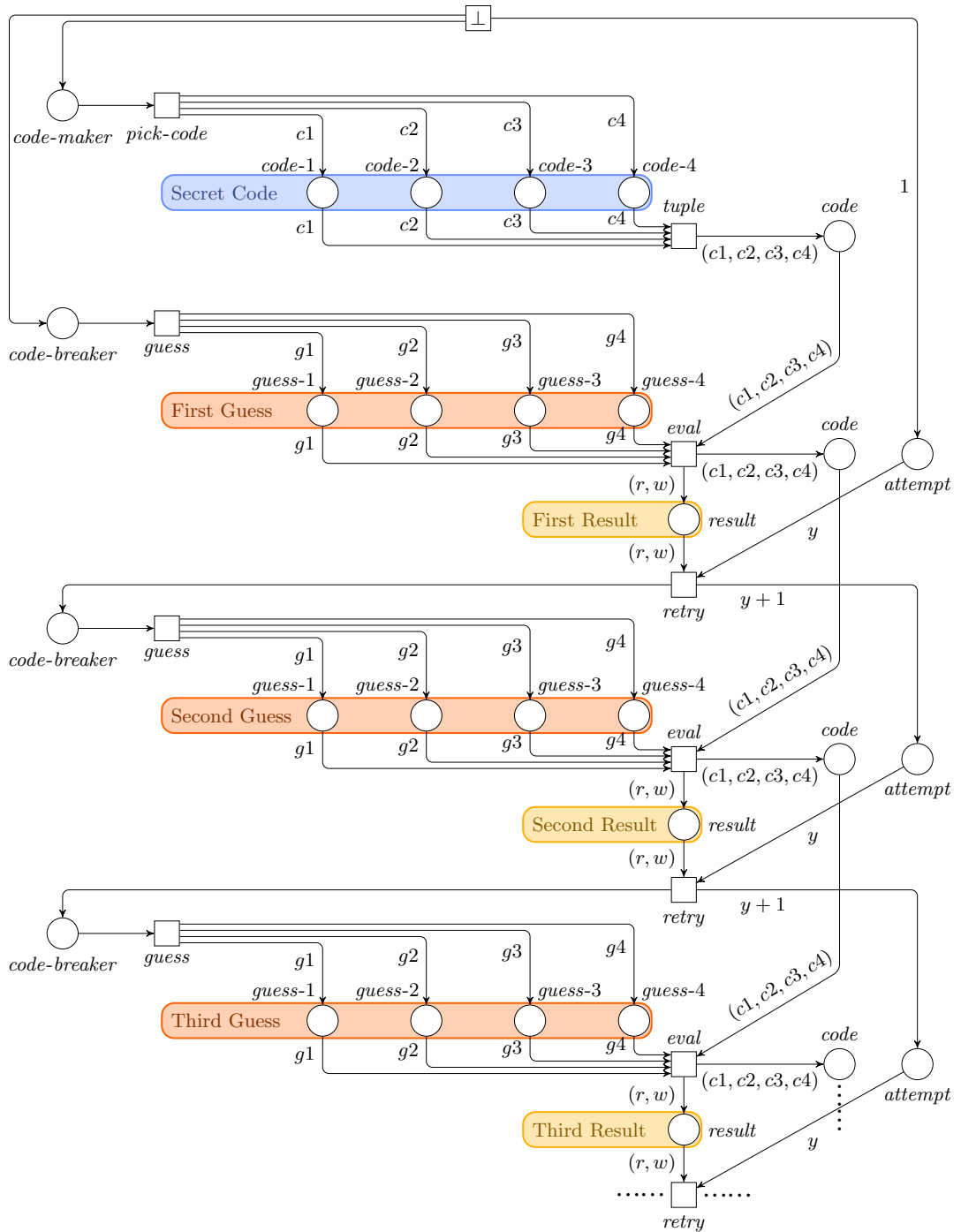
Figure 13: A prefix of the symbolic unfolding of the Mastermind net from Fig. 8.