Regular Separability of Well-Structured Transition Systems

Wojciech Czerwiński
Sławomir Lasota
University of Warsaw
wczerwin,sl@mimuw.edu.pl

Roland Meyer
Sebastian Muskalla
Prakash Saivasan
TU Braunschweig
roland.meyer,s.muskalla,p.saivasan@tu-bs.de

K Narayan Kumar
CMI
kumar@cmi.ac.in

Abstract

We investigate the languages recognized by well-structured transition systems (WSTS) with upward and downward compatibility. Our first result shows that, under very mild assumptions, every two disjoint WSTS languages are regularly separable: There is a regular language containing one of them and being disjoint from the other. As a consequence, if a language as well as its complement are both recognized by WSTS, then they are necessarily regular. Our second result shows that, for the case of Petri nets, the complexity of the backwards coverability algorithm yields a bound on the size of the regular separator. We complement it by a lower bound construction.

Keywords regular separability, well-structured transition systems, coverability languages, regularity problem, Petri nets

1 Introduction

We study the languages recognized by well-structured transition systems (WSTS) [1, 5, 23, 24, 27]. WSTS form an abstract framework subsuming several widely-studied models, like Petri nets [22] and their extensions with transfer [21], data [53], and time [4], graph rewriting systems [34], depth-bounded systems [20, 46, 56], ad-hoc networks [3], process algebras [13], lossy channel systems (LCS) [5], and programs running under weak memory models [6, 7]. Besides their applicability, the importance of WSTS stems from numerous decidability results. Finkel showed the decidability of termination and boundedness [23, 24]. Abdulla came up with a backward algorithm for coverability [5], for which a matching forward procedure was found only much later [29]. Several simulation and equivalence problems are also decidable for WSTS [27]. The work on WSTS even influenced algorithms for regular languages [57] and recently led to the study of new complexity classes [54].

Technically, a WSTS is a transition system equipped with a quasi order on the configurations that satisfies two properties. It is a well quasi order and it is (upward or downward) compatible with the transition relation in the sense that it forms a simulation relation. For our language-theoretic study, we assume the transitions to be labeled and to be given sets of initial and final configurations. The set of final configurations is supposed to be upward or downward closed wrt. the quasi order of the WSTS. When specialized to VAS, this yields the so-called *covering languages*.

For WSTS languages, we study the problem of regular separability. Given two languages $\mathcal{L}$ and $\mathcal{K}$ over the same alphabet, a *separator* is a language $\mathcal{R}$ that contains one of the languages and is disjoint from the other, $\mathcal{L} \subseteq \mathcal{R}$ and $\mathcal{R} \cap \mathcal{K} = \emptyset$. The separator is regular if it is a regular language. Separability has recently attracted considerable attention. We discuss the related work in a moment.

Disjointness is clearly necessary for regular separability. We show that for most WSTS, disjointness is also sufficient. Our main result is the following:

Any two disjoint WSTS languages are regularly separable.

The only assumption we need is that, in the case of upward-compatible WSTS, one of the WSTS is finitely branching. For the case of downward-compatible WSTS, one should be deterministic.

The proof links inductive invariants from verification to separability in formal languages and finally ideal completions from lattice theory. We proceed in two steps. In a first step, we show that any inductive invariant (of the product of the given systems) gives rise to a regular separator — provided it can be finitely represented. Interestingly, we do not even need WSTS here but only upward compatibility. An inductive invariant is a set of configurations that contains the initial ones, is closed under the transition relation, and is disjoint from the final configurations. In a second step, we show that finitely-represented invariants always exist. The insight is that, in a WSTS, any inductive invariant can be finitely represented by its ideal decomposition. This ideal decomposition yields states in the ideal completion of the WSTS, and the first step applies.

The result has theoretical as well as practical applications. On the theoretical side, recall the following about Petri nets from [47, 48]: Every two Petri net covering languages that are complements of each other are necessarily regular. The result not only follows from ours, but the same applies to other classes of WSTS, for instance to the languages of LCS, and actually to *all* WSTS languages fulfilling the above-mentioned assumptions. For instance, if the covering language of a Petri net is the complement of the language of an LCS, they are necessarily regular; and if the languages are just disjoint, they are regularly separable.

The result is also important in verification. In 2016 and 2017, the Software Verification Competition was won by so-called language-theoretic algorithms [31]. These algorithms replace the classical state space search by proofs of language disjointness (between a refinement of the control-flow language and the language of undesirable behavior). Regular separators are precisely what is needed to prove disjointness. In this setting, regular separators seem to play the role that inductive invariants play for safety verification. Indeed, our results establishes a first link between the two.

We accompany our main result by two more findings. The first ones are determinization results that broaden the applicability of our results. For upward compatibility, we show that every finitely branching WSTS can be determinized. For downward compatiblity, we show that every WSTS can be determinized where the quasi order is even an ω^2-wqo. In fact all examples from the literature are ω^2-WSTS, hence they determinize, and in consequence satisfy the assumptions of our results.

Our second accompanying result is on the size of regular separators for Petri nets. We show how to construct a regular separator

in the form of a non-deterministic automaton of size triply exponential in the given nets. With the main result at hand, the result amounts to giving a bound on the size of a finite representation of an inductive invariant. As inductive invariant, we use the complement of the configurations backward reachable from the final ones. The estimation starts from a result on the size of a basis for the backward reachable configurations [39] and reasons about the complementation. There is a matching lower bound, but only for deterministic automata.

Outline. Section 2 recalls the basics on WSTS. The determinization results can be found in Section 3. They prepare the main result in Section 4. The state complexity of separators for Petri nets is in Section 5. Section 6 concludes the paper.

Related Work. Separability is a widely-studied problem in Theoretical Computer Science. A classical result says that every two co-recursively enumerable languages are recursively separable, i.e. separable by a recursive language [28]. In the area of formal languages, separability *of* regular languages by subclasses thereof was investigated most extensively as a decision problem: Given two regular languages, decide whether they are separable by a language from a fixed subclass. For the following subclasses, among others, the separability problem of regular languages is decidable: The piecewise-testable languages, shown independently in [18] and [50], the locally and locally threshold-testable languages [49], the languages definable in first-order logic [52], and the languages of certain higher levels of the first-order hierarchy [51].

Regular separability of classes larger than the regular languages attracted little attention until recently. As a remarkable example, already in the 70s, the undecidability of regular separability of context-free languages has been shown [55] (cf. also a later proof [32]); then the undecidability has been strengthened to visibly pushdown languages [37] and to languages of one-counter automata [17].

An intriguing problem, to the best of our knowledge still open, is the decidability of regular separability of Petri net languages, under the proviso that acceptance is by *reaching* a distinguished final configuration. As for now, positive answers are known only for subclasses of VAS languages: PSPACE-completeness for one-counter nets (i.e. one-dimensional vector addition systems with states) [17], and elementary complexity for languages recognizable by Parikh automata (or, equivalently, by integer vector addition systems) [14]. Finally, regular separability of *commutative closures of* VAS languages has been shown to be decidable in [15]. As a consequence of this paper, regular separability of two VAS languages reduces to disjointness of the same two VAS languages (and is thus trivially decidable), given that acceptance is by *covering* a distinguished final configuration.

Languages of upward-compatible WSTS were investigated e.g. in [30], where interesting closure properties have been shown, including a natural pumping lemma. Various subclasses of languages of WSTS have been considered, e.g. in [2, 19, 44].

2 Well structured transition systems

Well Quasi Orders. A quasi order $(X, \leq)$, i.e. a set X equipped with a reflexive and transitive binary relation $\leq$, is called *well quasi order* (wqo) if for every infinite sequence $x_1, x_2, \ldots \in X$ there are indices $i < j$ such that $x_i \leq x_j$. It is folklore that $(X, \leq)$ is wqo iff it admits

neither an infinite descending sequence (i.e. it is well-founded) nor an infinite antichain (i.e. it has the finite antichain property).

We will be working either with wqos, or with ω^2-wqos, a strengthening of wqos. We prefer not to provide the standard definition of ω^2-wqo (which can be found, e.g in [43]), as it is technical and would not serve our aims in this paper. Instead, we take the characterization provided by Lemma 2.2 below as a working definition. The class of ω^2-wqos provides a framework underlying the forward WSTS analysis developed in [25, 26]. Both classes, namely wqos and ω^2-wqos, are stable under various operations like taking the Cartesian product; also the natural liftings of a wqo (resp. ω^2-wqo) to finite multisets (multiset embedding) or finite sequences (Higman ordering) are wqos (resp. ω^2-wqos) again.

A subset $U \subseteq X$ is *upward closed* with respect to $\leq$ if $u \in U$ and $u' \geq u$ implies $u' \in U$. Similarly, one defines *downward closed* sets. Clearly, U is upward closed iff $X \setminus U$ is downward closed. The *upward* and *downward closure* of a set $U \subseteq X$ are defined as:

$$\uparrow U = \{x \in X \mid \exists u \in U, \ x \geq u\} \ \text{ and}$$
$$\downarrow U = \{x \in X \mid \exists u \in U, \ x \leq u\} \ .$$

The family of all upward closed subsets of X we denote by $\mathcal{P}^\uparrow(X)$, and the family of all downward closed subsets of X by $\mathcal{P}^\downarrow(X)$. If $(X, \leq)$ is a wqo then every upward closed set is the upward closure of a finite set, namely of the set of its minimal elements. This is not the case for downward closed set; we thus distinguish a subfamily $\mathcal{P}^\downarrow_{\text{fin}}(X) \subseteq \mathcal{P}^\downarrow(X)$ of *finitary* downward closed subsets of X, i.e. downward closures of finite sets. In general, these are not necessarily finite sets (e.g. consider the set $\mathbb{N} \cup \{\omega\}$ with ω bigger than all natural numbers, and the downward closure of $\{\omega\}$). The set $\mathcal{P}^\downarrow_{\text{fin}}(X)$, ordered by inclusion, is a wqo whenever $(X, \leq)$ is:

Lemma 2.1. $\left(\mathcal{P}^\downarrow_{\text{fin}}(X), \subseteq\right)$ *is a wqo iff* $(X, \leq)$ *is a wqo.*

This property does not necessarily extend to the whole set $\mathcal{P}^\downarrow(X)$ of all downward closed subsets of X. As shown in [33]:

Lemma 2.2. $\left(\mathcal{P}^\downarrow(X), \subseteq\right)$ *is a wqo iff* $(X, \leq)$ *is an* ω^2-*wqo.*

As a matter of fact, [33] considers the reverse inclusion order on upward closed sets, which is clearly isomorphic to the inclusion order on downward closed sets.

Labeled Transition Systems. In the sequel we always fix a finite alphabet Σ. A labeled transition system (LTS) $\mathcal{W} = (S, T, I, F)$ over Σ consists of a set of *configurations* S, a set of *transitions* $T \subseteq S \times \Sigma \times S$, and subsets $I, F \subseteq S$ of *initial* and *final* configurations. We write $s \xrightarrow{a} s'$ instead of $(s, a, s') \in T$. A path from configuration s to configuration s' over a word $w = a_0 \cdots a_{k-1}$ is a sequence of configurations $s = s_0, s_1, \ldots, s_{k-1}, s_k = s'$ such that $s_i \xrightarrow{a_i} s_{i+1}$ for all $i \in \{0, \ldots, k-1\}$. We write $s \xrightarrow{w} s'$. For a subset $X \subseteq S$ of configurations and a word $w \in \Sigma^*$ we write

$$\text{Reach}_{\mathcal{W}}(X, w) = \{s \in S \mid \exists x \in X \colon x \xrightarrow{w} s\}$$
$$\text{Reach}^{-1}_{\mathcal{W}}(X, w) = \{s \in S \mid \exists x \in X \colon s \xrightarrow{w} x\}$$

for the set of all configurations reachable (resp. reversely reachable) from X along w. Note that we have $\text{Reach}_{\mathcal{W}}(X, \varepsilon) = X = \text{Reach}^{-1}_{\mathcal{W}}(X, \varepsilon)$. Important special cases will be the set of all a-successors (resp. a-predecessors) for $a \in \Sigma$, i.e. configurations reachable along a one-letter word a, and

the configurations reachable from the initial configurations I (resp. reversely reachable from the final configurations F):

$$\text{Succ}_W(X, a) = \text{Reach}_W(X, a) \qquad \text{Reach}_W(w) = \text{Reach}_W(I, w)$$

$$\text{Pred}_W(X, a) = \text{Reach}^{-1}_W(X, a) \qquad \text{Reach}^{-1}_W(w) = \text{Reach}^{-1}_W(F, w)$$

satisfying the following equalities for all $w \in \Sigma^*$ and $a \in \Sigma$:

$$\text{Reach}_W(w.a) = \text{Succ}_W(\text{Reach}_W(w), a) \qquad (1)$$

$$\text{Reach}^{-1}_W(a.w) = \text{Pred}_W(\text{Reach}^{-1}_W(w), a) . \qquad (2)$$

For future use we establish the notation for the whole set of (reversely) reachable configurations:

$$\text{Reach}_W = \bigcup_{w \in \Sigma^*} \text{Reach}_W(w) \qquad \text{Reach}^{-1}_W = \bigcup_{w \in \Sigma^*} \text{Reach}^{-1}_W(w) .$$

An LTS $W = (S, T, I, F)$ is *finitely branching* if I is finite and for every configuration $s \in S$ and each $a \in \Sigma$ there are only finitely many configurations $s' \in S$ such that $s \xrightarrow{a} s'$. Furthermore, W is *deterministic* if it has exactly one initial configuration, and for every $s \in S$ and each $a \in \Sigma$ there is exactly one $s' \in S$ such that $s \xrightarrow{a} s'$. If W is deterministic, we write $s' = \text{Succ}_W(s, a)$ (resp. $s' = \text{Reach}_W(w)$) instead of $\{s'\} = \text{Succ}_W(s, a)$ (resp. $\{s'\} = \text{Reach}_W(w)$).

The language recognized by W, denoted $\mathcal{L}(W)$, is the set of words which occur on some path starting in an initial configuration and ending in a final one, i.e.

$$\mathcal{L}(W) = \{w \in \Sigma^* \mid \exists i \in I, f \in F \colon i \xrightarrow{w} f\} .$$

We call two LTS W, W' are *equivalent* if their languages are the same. They are *reverse-equivalent* if $\mathcal{L}(W) = \{\text{rev}(w) \mid w \in \mathcal{L}(W')\}$ with $\text{rev}(a_1 \ldots a_k) = a_k \ldots a_1$.

Note that we did not allow for ε-steps in transition systems. Even if ε-steps can be easily eliminated by pre-composing and post-composing every transition $s \xrightarrow{a} s'$ with the reflexive-transitive closure of $\xrightarrow{\varepsilon}$, this transformation does not necessarily preserve finite branching.

Synchronized Products. Consider LTS $W = (S, T, I, F)$ and $W' = (S', T', I', F')$. Their *synchronized product* is the LTS $W \times W' = (S_\times, T_\times, I_\times, F_\times)$ defined as follows: The configurations are tuples of configurations, $S_\times = S \times S'$, and the initial and final configurations are $I_\times = I \times I'$ and $F_\times = F \times F'$, respectively. The transition relation is defined by

$$(s, s') \xrightarrow{a} (r, r') \text{ in } W \times W' \quad \text{if} \quad s \xrightarrow{a} r \text{ in } W$$
$$\text{and } s' \xrightarrow{a} r' \text{ in } W' .$$

It is immediate from the definition that the language of the product is the intersection of the languages, i.e. $\mathcal{L}(W \times W') = \mathcal{L}(W) \cap \mathcal{L}(W')$. If W and W' both are finitely branching, then so is their product.

Upward-Compatible Well-Structured Transition Systems. Now we define a labeled version of *well-structured transition systems* as described in [27], here called upward-compatible well structured transition system (UWSTS). We start by defining the more general notions of quasi ordered LTS and weak UWSTS.

By a *quasi-ordered LTS* $W = (S, T, \le, I, F)$ we mean an LTS (S, T, I, F) extended with a quasi order $\le$ on configurations.

A *weak UWSTS* is a quasi-ordered LTS such that the set F of final configurations F is upward closed[1] with respect to $\le$, and the following upward compatibility[2] is satisfied: whenever $s \le s'$ and $s \xrightarrow{a} r$, then $s' \xrightarrow{a} r'$ for some $r' \in S$ such that $r \le r'$.

$$
\begin{array}{ccc}
s' & \xdashrightarrow{\;a\;} & r' \,(\exists) \\
\rotatebox{90}{$\ge$}\!| & & \rotatebox{90}{$\ge$}\!| \\
s & \xrightarrow{\;\;a\;\;} & r
\end{array}
$$

In other words, $\le$ is a simulation relation. We observe that upward compatibility extends to words:

Lemma 2.3. *Let $w \in \Sigma^*$. Whenever $s \le s'$ and $s \xrightarrow{w} r$, then $s' \xrightarrow{w} r'$ for some $r' \in S$ such that $r \le r'$.*

If the order $(S, \le)$ in a weak UWSTS $W = (S, T, \le, I, F)$ is a wqo, we call W a UWSTS.

As F is upward closed, W is equivalent to its downward closure $\downarrow W$, obtained from W by replacing the set I by its (not necessarily finite) downward closure $\downarrow I$ with respect to $\le$, and by extending the transition relation as follows: $s \xrightarrow{a} r$ in $\downarrow W$ if $s \xrightarrow{a} r'$ in W for some $r' \ge s$. Note that with respect to the extended transition relation, $\text{Succ}_{\downarrow W}(X, a)$ is downward closed for every $X \subseteq S$. One easily checks that $\downarrow W$ still satisfies upward compatibility, and every word accepted by W is also accepted by $\downarrow W$. The converse implication follows by the following simulation of $\downarrow W$ by W:

Lemma 2.4. *Let $w \in \Sigma^*$. Whenever $s \le s'$ and $s \xrightarrow{w} r$ in $\downarrow W$, then $s' \xrightarrow{w} r'$ in W for some $r' \in S$ such that $r \le r'$.*

The synchronized product of two weak UWSTS $(S, T, \le, I, F)$ and $(S', T', \le', I', F')$ is still a weak UWSTS with respect to the product order $\le_\times$ defined by $(x, x') \le_\times (y, y')$ iff $x \le y$ and $x' \le' y'$. Indeed, $F \times F'$ is upward closed wrt. $\le_\times$ and the transition relation satisfies upward compatibility. Since the product order of two wqos is again a wqo, the the synchronized product of two UWSTS is a UWSTS.

When $\le$ is a ω^2-wqo, the UWSTS W is called ω^2-*UWSTS*. When the LTS (S, T, I, F) is finitely branching (resp. deterministic), the UWSTS W is called *finitely-branching UWSTS* (resp. *deterministic UWSTS*). In the sequel we speak shortly of UWSTS-languages (resp. ω^2-UWSTS-languages, finitely-branching UWSTS-languages, deterministic UWSTS languages, etc.).

Downward-Compatible Well-Structured Transition Systems. A downward-compatible well-structured transition system (DWSTS) is defined like its upward-compatible counterpart, with two modifications. First, we assume the set of final configurations F to be downward closed, instead of being upward closed. Second, instead of upward compatibility, we require its symmetric variant, namely *downward compatibility*: Whenever $s' \le s$ and $s \xrightarrow{a} r$, then $s' \xrightarrow{a} r'$ for some $r' \in S$ such that $r' \le r$. In other words, the inverse of $\le$ is a simulation relation. Downward compatibility extends to words, which can been shown similar to Lemma 2.3:

Lemma 2.5. *Let $w \in \Sigma^*$. Whenever $s' \le s$ and $s \xrightarrow{w} r$, then $s' \xrightarrow{w} r'$ for some $r' \in S$ such that $r' \le r$.*

---

[1]Languages defined by upwards closed sets of final configurations are called *coverability languages* in the literature.

[2]In the terminology of [27], this is strong compatibility.

Symmetrically to the downward closure of a UWSTS, we may define the upward closure $\uparrow W$ of a DWSTS W that recognizes the same language.

As above, we also speak of finitely-branching DWSTS, or ω^2-DWSTS. We jointly call UWSTS and DWSTS just WSTS.

Examples of WSTS. Various well known and intensively investigated models of computation happen to be either an UWSTS or DWSTS. The list of natural classes of systems which are UWSTS contains, among the others: vector addition systems (VAS) resp. Petri nets and their extensions (e.g. with reset arcs or transfer arcs); lossy counter machines [10]; string rewriting systems based on context-free grammars; lossy communicating finite state machines (aka lossy channel systems, LCS) [12]; and many others. In the first two models listed above the configurations are ordered by the multiset embedding, while in the remaining two ones the configurations are ordered by Higman's subsequence ordering. The natural examples of UWSTS, including all models listed above, are ω^2-UWSTS and, when considered without ε-transitions, finitely-branching.

DWSTS are less common. A natural source of examples is *gainy* models, like gainy counter system machines or gainy communicating finite state machines. For an overview, see e.g. page 31 of [27].

3 Expressibility

In this section we compare language expressivity of different subclasses of WSTS, as summarized in Theorem 3.1 below.

For classes of WSTS, we use the symbol $\subseteq$ to say that the language of every WSTS from one class is also the language of a WSTS from another class; and we use the symbol $\subseteq_{\mathrm{rev}}$ to say that the language of every WSTS from one class is the reverse of the language of a WSTS from another class.

Theorem 3.1. *The following relations hold between the WSTS language classes:*

$$\begin{aligned}
\omega^2\text{-}UWSTS \;\subseteq\;\; &\text{deterministic } UWSTS \\
=\;\; &\text{finitely-branching } UWSTS \;\subseteq\; \text{all } UWSTS\,, \\
\omega^2\text{-}DWSTS \;\subseteq\;\; &\text{deterministic } DWSTS \\
\subseteq\;\; &\text{finitely-branching } DWSTS \;=\; \text{all } DWSTS\,, \\
\omega^2\text{-}UWSTS \;\subseteq_{\mathrm{rev}}\;\; &\text{deterministic } DWSTS\,, \\
\omega^2\text{-}DWSTS \;\subseteq_{\mathrm{rev}}\;\; &\text{deterministic } UWSTS\,.
\end{aligned}$$

In short words: ω^2-UWSTSes and ω^2-DWSTSes determinize, and reverse-determinize; finitely branching UWSTSes determinize too; and (unrestricted) DWSTSes are equivalent to finitely branching DWSTSes. In the rest of this section, we formulate and prove a series of lemmata which jointly prove Theorem 3.1.

Lemma 3.2. *Every ω^2-UWSTS is equivalent to a deterministic UWSTS.*

Proof. Let $W = (S, T, \preceq, I, F)$ be an arbitrary ω^2-UWSTS. Wlog. we assume that $W = \downarrow W$, i.e. $\mathrm{Succ}_W(X, a)$ is downward closed for every X and a, and $\mathrm{Reach}_W(w)$ is downward closed for every w.

We define a deterministic UWSTS $\overline{W} = (\bar{S}, \bar{T}, \subseteq, \bar{I}, \bar{F})$ essentially as a powerset construction on W. Let the configurations $\bar{S} = \mathcal{P}^{\downarrow}(S)$ be the downward closed subsets of S, ordered by inclusion $\subseteq$; this is a wqo by Lemma 2.2. Let the only initial configuration of $\overline{W}$ be

the set I itself, i.e. let $\bar{I} = \{I\}$. (Recall that I is downward closed in $\downarrow W$.) Let the accepting configurations $\bar{F} = \{X \in \bar{S} \mid X \cap F \neq \emptyset\}$ be those downward closed subsets $X \in \bar{S}$ which contain at least one accepting configuration from F. The transition relation $\bar{T}$ is defined by the direct image,

$$X \xrightarrow{a} \mathrm{Succ}_W(X, a) \tag{3}$$

which is well-defined as $\mathrm{Succ}_W(X, a) \subseteq S$ is always downward closed. The equality of the languages of W and $\overline{W}$ follows directly from the following claim:

Claim: For every $w \in \Sigma^*$, $\mathrm{Reach}_{\overline{W}}(w) = \{\mathrm{Reach}_W(w)\}$.

The claim is shown by induction on the length of w. The base case is $\bar{I} = \{I\}$, and the step follows by (1) combined with the equality $\mathrm{Succ}_{\overline{W}}(X, a) = \{\mathrm{Succ}_W(X, a)\}$, a reformulation of (3). $\quad\square$

Lemma 3.3. *Every ω^2-DWSTS is equivalent to a deterministic DWSTS.*

This can be proven similarly to Lemma 3.2. Also similarly, but using the finitary downward closed subsets, we prove the following result.

Lemma 3.4. *Every finitely-branching UWSTS is equivalent to a deterministic UWSTS.*

Lemma 3.5. *Every DWSTS is equivalent to a finitely branching DWSTS.*

Proof. Given a DWSTS $W = (S, T, \preceq, I, F)$ we define a finitely-branching one $\overline{W} = (\bar{S}, \bar{T}, \preceq, \bar{I}, \bar{F})$. Wlog. we assume, $W = \uparrow W$.

The configurations of $\overline{W}$ are the same as those of W, $\bar{S} = S$. The transition relation $\bar{T}$ of $\overline{W}$ is a subset of T, where only minimal successors wrt. $\preceq$ are allowed:

$$\mathrm{Succ}_{\overline{W}}(x, a) \stackrel{\mathrm{def}}{=} \min(\mathrm{Succ}_W(x, a)). \tag{4}$$

The initial configurations of $\overline{W}$ are minimal initial configurations of W, $\bar{I} = \min(I)$. As $(S, \preceq)$ is a wqo, all upward closed sets have a minimal basis. Since in $\uparrow W$, the set of successors and the set of initial states are upward closed, $\bar{T}$ is finitely branching and $\bar{I}$ is finite. Finally, we put $\bar{F} = F$. The equality of the languages of W and $\overline{W}$ is due to the following equality for all $w \in \Sigma^*$:

$$\mathrm{Reach}_{\overline{W}}(w) = \min(\mathrm{Reach}_W(w)). \tag{5}$$

Indeed, as F is downward closed, we have that $\mathrm{Reach}_{\overline{W}}(w)$ contains a configuration from F if and only if $\mathrm{Reach}_W(w)$ does.

Finally, the equality (5) itself is shown by induction on the length of w. The base case is $\bar{I} = \min(I)$. The induction step follows by (1) and (4). We start with

$$X \stackrel{\mathrm{def}}{=} \mathrm{Reach}_{\overline{W}}(wa) \stackrel{(1)}{=} \mathrm{Succ}_{\overline{W}}(\mathrm{Reach}_{\overline{W}}(w), a)$$
$$\stackrel{(4)}{=} \min(\mathrm{Succ}_W(\mathrm{Reach}_{\overline{W}}(w), a))$$

and use the induction hypothesis to derive $X = \min(\mathrm{Succ}_W(\min(\mathrm{Reach}_W(w)), a))$. Finally, using downward-compatibility and the assumption that the transition relation in W is upward closed we observe that for every upward closed set $Y \subseteq S$,

$$\mathrm{Succ}_W(\min(Y), a) = \mathrm{Succ}_W(Y, a),$$

which allows us to complete the induction step:
$$X = \min(\mathrm{Succ}_W(\mathrm{Reach}_W(w), a)) \stackrel{(1)}{=} \min(\mathrm{Reach}_W(wa)) \,. \quad\square$$

By using similar powerset constructions, we can show the following two results.

Lemma 3.6. *Every ω^2-UWSTS is reverse-equivalent to a deterministic DWSTS.*

Lemma 3.7. *Every ω^2-DWSTS is reverse-equivalent to a deterministic UWSTS.*

4 Regular Separability

We now show our first main results: Under mild assumptions, disjoint DWSTS resp. disjoint UWSTS are regularly separable. Both theorems follow from a technical result that establishes a surprising link between verification and formal language theory: Every inductive invariant (of a suitable product WSTS) that has a finite representation can be turned into a regular separator. With this, the proofs of regular separability are invariant constructions.

Main Results. We say that two languages $\mathcal{L}$ and $\mathcal{K}$ over the same alphabet are *regularly separable* if there is a regular language $\mathcal{R}$ that satisfies $\mathcal{L} \subseteq \mathcal{R}$ and $\mathcal{R} \cap \mathcal{K} = \emptyset$. For two WSTS $\mathcal{W}$ and $\mathcal{W}'$, we say that they are regularly separable if so are their languages. Disjointness is clearly necessary for regular separability. Our first main results show that for most WSTS disjointness is also sufficient:

Theorem 4.1. *Every two disjoint DWSTS, one of them deterministic, are regularly separable.*

Theorem 4.2. *Every two disjoint UWSTS, one of them finitely branching, are regularly separable.*

For a class of languages C, we call a language *doubly C*, if the language as well as its complement are in C. We obtain the following corollary, generalizing earlier results for Petri nets [47, 48].

Corollary 4.3. *Every doubly ω^2-UWSTS resp. doubly ω^2-DWSTS resp. doubly finitely branching UWSTS language is regular.*

The rest of the section is devoted to the proofs. We will use that the product of two disjoint WSTS is again a WSTS with the empty language. Whenever the language of a WSTS is empty, we can find an inductive invariant, a downward-closed set of configurations separating the reachability set from the final configurations. Given a finite representation for such an invariant, we show how to turn it into a regular separator, provided one of the WSTS is deterministic. This is our key technical insight, formulated as Theorem 4.6 below.

The proof of Theorem 4.1 follows directly from this result. For Theorem 4.2, we consider the ideal completion of an UWSTS, an extended system in which every downward-closed set has a finite representation. This in particular applies to inductive invariants, as we show in the form of Proposition 4.16: Any inductive invariant in the original UWSTS induces an inductive invariant in the ideal completion that has a finite representation. Combining this result with Theorem 4.6 yields the desired proof.

Turning Inductive Invariants into Regular Separators. Inductive invariants are a standard tool in the safety verification of programs [42]. Technically, an inductive invariant (of a program for a safety property) is a set of program configurations that includes the initial ones, is closed under the transition relation, and is disjoint from the set of undesirable states. The following definition lifts the notion to WSTS, where it is natural to require inductive invariants to be downward-closed.

Definition 4.4. *An* inductive invariant *for a weak UWSTS $\mathcal{W}$ with configurations S is a downward-closed set $X \subseteq S$ with the following three properties:*

$$I \subseteq X \,, \tag{6}$$

$$F \cap X = \emptyset \,, \tag{7}$$

$$\text{SUCC}_{\mathcal{W}}(X, a) \subseteq X \text{ for all } a \in \Sigma \,. \tag{8}$$

An inductive invariant X is finitely-represented *if $X = \downarrow Q$ for a finite set $Q \subseteq S$.*

By (6) and (8), the invariant has to contain the whole reachability set. By (7) and (8), it has to be disjoint from the predecessors of the final configurations:

$$\text{REACH}_{\mathcal{W}} \subseteq X \,, \qquad \text{REACH}_{\mathcal{W}}^{-1} \cap X = \emptyset \,.$$

This means every inductive invariant shows language emptiness. Even more, inductive invariants are complete for proving emptiness, like inductive invariants for programs are (relatively) complete for proving safety [16].

Lemma 4.5. *Consider a weak UWSTS $\mathcal{W}$. We have $\mathcal{L}(\mathcal{W}) = \emptyset$ iff there is an inductive invariant for $\mathcal{W}$.*

For completeness, observe that $X = \downarrow\text{REACH}_{\mathcal{W}}$ is an inductive invariant. It is the least one wrt. inclusion. There is also a greatest inductive invariant, namely the complement of $\text{REACH}_{\mathcal{W}}^{-1}$. Note that, due to upward compatibility, $\text{REACH}_{\mathcal{W}}^{-1}$ is always upward-closed.

Other invariants may have the advantage of being easier to represent. We will be particularly interested in invariants that are finitely-represented in the sense that they form the downward closure of a finite set.

Here is the core result. Consider two disjoint UWSTS. Any finitely-represented inductive invariant for the product WSTS can be turned into a regular separator. We will comment on the assumed determinism in a moment.

Theorem 4.6. *Let $\mathcal{W}$ and $\mathcal{W}'$ be disjoint weak UWSTS, one of them deterministic, such that $\mathcal{W} \times \mathcal{W}'$ admits a finitely-represented inductive invariant $\downarrow Q$. Then $\mathcal{W}$ and $\mathcal{W}'$ are regularly separable by the language of a finite automaton with states Q.*

For the definition of the separator, let $\mathcal{W} = (S, T, \preceq, I, F)$ be an arbitrary weak UWSTS and let $\mathcal{W}' = (S', T', \preceq', I', F')$ be a deterministic one such that their languages are disjoint. Let

$$\mathcal{W}_\times = \mathcal{W} \times \mathcal{W}' = (S_\times, T_\times, \preceq_\times, I_\times, F_\times)$$

be their synchronized product. By the disjointness of $\mathcal{W}$ and $\mathcal{W}'$ we know that $\mathcal{L}(\mathcal{W}_\times) = \emptyset$. Let $Q \subseteq S_\times$ be a finite set such that $\downarrow Q$ is an inductive invariant.

We define a finite automaton $\mathcal{A}$ with states Q whose language will contain $L(\mathcal{W})$ while being disjoint from $L(\mathcal{W}')$. The idea is to over-approximate the configurations of $\mathcal{W}_\times$ by the elements available in Q. The fact that $\text{REACH}_{\mathcal{W}_\times} \subseteq \downarrow Q$ guarantees that every configuration $(s, s') \in S_\times$ has such a representation. Since we seek to approximate the language of $\mathcal{W}$, the final states only refer to the $\mathcal{W}$-component. Transitions are approximated existentially.

Definition 4.7. *We define the* separating automaton induced by Q *to be $\mathcal{A} = (Q, \rightarrow, Q_I, Q_F)$. A state is initial if it dominates some initial configuration of $\mathcal{W}_\times$:*

$$Q_I = \{(s, s') \in Q \mid (i, i') \preceq_\times (s, s') \text{ for some } (i, i') \in I_\times\} \,.$$
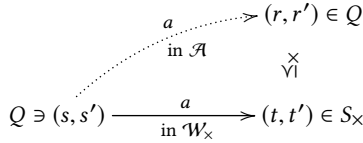
As final states we take pairs whose $\mathcal{W}$-component is final,

$$Q_F = \{(s, s') \in Q \mid s \in F\} \ .$$

Finally, the transition relation in $\mathcal{A}$ is an over-approximation of the transition relation in $\mathcal{W}_\times$:

$$(s, s') \xrightarrow{a} (r, r') \text{ in } \mathcal{A} \quad \text{if } (s, s') \xrightarrow{a} (t, t') \text{ in } \mathcal{W}_\times$$
$$\text{for some } (t, t') \preceq_\times (r, r') \ .$$

The following diagram illustrates the construction:

To show separation, we need to prove $\mathcal{L}(\mathcal{W}) \subseteq \mathcal{L}(\mathcal{A})$ and $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{W}') = \emptyset$. We begin with the former. As $\mathcal{W}'$ is deterministic, $\mathcal{W}_\times$ contains all computations of $\mathcal{W}$. Due to upward compatibility, $\mathcal{A}$ over-approximates the computations in $\mathcal{W}_\times$. Combining these two insights, which are summarized in the next lemma, yields the result.

Lemma 4.8. *(1) For every $s \in \text{REACH}_{\mathcal{W}}(w)$ there is some $(s, s') \in \text{REACH}_{\mathcal{W}_\times}(w)$. (2) For every $(s, s') \in \text{REACH}_{\mathcal{W}_\times}(w)$ there is some $(r, r') \in \text{REACH}_{\mathcal{A}}(w)$ with $(s, s') \preceq_\times (r, r')$.*

Proposition 4.9. $\mathcal{L}(\mathcal{W}) \subseteq \mathcal{L}(\mathcal{A})$.

It remains to prove disjointness of $\mathcal{L}(\mathcal{A})$ and $\mathcal{L}(\mathcal{W}')$. The key observation is that, due to determinism, $\mathcal{W}'$ simulates the computations of $\mathcal{A}$ — in the following sense: If upon reading a word $\mathcal{A}$ reaches a state (s, s'), then the unique computation of $\mathcal{W}'$ will reach a configuration dominated by s'.

Lemma 4.10. *For every $w \in \Sigma^*$ and every $(s, s') \in \text{REACH}_{\mathcal{A}}(w)$ we have $\text{REACH}_{\mathcal{W}'}(w) \preceq' s'$.*

With this lemma we can show disjointness. Towards a contradiction, suppose some word w satisfies $w \in \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{W}')$. As $w \in \mathcal{L}(\mathcal{A})$, there is a configuration $(s, s') \in \text{REACH}_{\mathcal{A}}(w)$ with $s \in F$. As $w \in \mathcal{L}(\mathcal{W}')$, the unique configuration $\text{REACH}_{\mathcal{W}'}(w)$ belongs to F'. With the previous lemma and the fact that F' is upward-closed, we conclude $s' \in F'$. Together, $(s, s') \in F_\times$, which contradicts the fact that $\downarrow Q$ is an inductive invariant, Property (7).

Proposition 4.11. $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{W}') = \emptyset$.

Together, Proposition 4.9 and 4.11 show Theorem 4.6.

Proof of Theorem 4.1. We apply Theorem 4.6 to prove our first main result. Consider an arbitrary DWSTS $\mathcal{W} = (S, T, \preceq, I, F)$ and a deterministic one $\mathcal{W}' = (S', T', \preceq', I', F')$. We start with the observation that the inversed versions of $\mathcal{W}$ and $\mathcal{W}'$, namely with the orders $\preceq^{-1}$ and $(\preceq')^{-1}$ and denoted by $\mathcal{W}^{-1}$ and $(\mathcal{W}')^{-1}$, are weak UWSTS. We claim that these weak UWSTS satisfy the assumptions of Theorem 4.6. The language of $\mathcal{W}_\times^{-1} = \mathcal{W}^{-1} \times (\mathcal{W}')^{-1}$ is empty since the language of $\mathcal{W}_\times = \mathcal{W} \times \mathcal{W}'$ is empty and inversion does not change the language, $\mathcal{L}(\mathcal{W}) = \mathcal{L}(\mathcal{W}^{-1})$ and similar for $\mathcal{W}'$. Inversion also does not influence determinism.

It remains to find an inductive invariant of $\mathcal{W}_\times^{-1}$ that is finitely represented. We claim that

$$X = \downarrow_{-1} \text{REACH}_{\mathcal{W}_\times^{-1}}$$

is a suitable choice. The subscript indicates that the downward closure is computed relative to the quasi order of $\mathcal{W}_\times^{-1}$. The proof of Lemma 4.5 shows that X is an inductive invariant. For the finite representation, note that inversion does not change the transition relation. Hence, $\mathcal{W}_\times$ and $\mathcal{W}_\times^{-1}$ reach the same configurations,

$$\text{REACH}_{\mathcal{W}_\times^{-1}} = \text{REACH}_{\mathcal{W}_\times} = Z \ .$$

With the definition of inversion, $X = \downarrow_{-1} Z = \uparrow Z$ holds. Moreover, $\uparrow Z = \uparrow \min(Z)$, with minimum and upward closure computed relative to $\mathcal{W}_\times$. Since the configurations of $\mathcal{W}_\times$ are well quasi ordered, $\min(Z)$ is finite. Another application of inversion yields $X = \uparrow \min(Z) = \downarrow_{-1} \min(Z)$. Hence, X is a finitely-represented downward-closed subset of $\mathcal{W}_\times^{-1}$.

By Theorem 4.6, the languages of $\mathcal{W}^{-1}$ and $(\mathcal{W}')^{-1}$ are regularly separable and so are the languages of $\mathcal{W}$ and $\mathcal{W}'$. □

Ideal Completions of UWSTS. The proof of regular separability for UWSTS is more involved. Here, we need the notion of ideal completions [8, 26]. We show that any invariant for a WSTS yields a finitely-represented invariant for the corresponding ideal completion. Theorem 4.2 follows from this.

An *ideal* in a wqo $(X, \preceq)$ is a non-empty downward-closed subset $Z \subseteq X$ which is directed: for every $z, z' \in Z$ there is a $z'' \in Z$ with $z \preceq z''$ and $z' \preceq z''$. Every downward-closed set decomposes into finitely many ideals. In fact, the finite antichain property is sufficient and necessary for this.

Lemma 4.12 ([26, 35, 40]). *In a wqo, every downward-closed set is a finite union of ideals.*

We use $\text{ID-DEC}_X(Z)$ to denote the set of inclusion-maximal ideals in Z. By the above lemma, $\text{ID-DEC}_X(Z)$ is always finite and

$$Z = \bigcup \text{ID-DEC}_X(Z) \ . \tag{9}$$

We will also make use of the fact that ideals are irreducible in the following sense.

Lemma 4.13 ([26, 35, 40]). *Let $(X, \preceq)$ be a wqo. If $Z \subseteq X$ is downward-closed and $I \subseteq Z$ is an ideal, then $I \subseteq J$ for some $J \in \text{ID-DEC}_X(Z)$.*

The *ideal completion* $(\overline{X}, \subseteq)$ of $(X, \preceq)$ has as elements all ideals in X. The order is inclusion. The ideal completion $\overline{X}$ can be seen as extension of X; indeed, every element $x \in X$ is represented by $\downarrow\{x\} \in \overline{X}$, and inclusion among such representations coincides with the original quasi order $\preceq$. Later, we will also need general ideals that may not be the downward closure of a single element.

Technically, the ideal completion of $\mathcal{W} = (S, T, \preceq, I, F)$ is a weak UWSTS $\overline{\mathcal{W}}$, where the given wqo is replaced by its ideal completion. The initial configurations are the ideals in the decomposition of $\downarrow I$. The transition relation is defined similarly, by decomposing $\downarrow \text{Succ}_{\mathcal{W}}(X, a)$, with X an ideal. The final configurations are the ideals that intersect F.

Definition 4.14 ([8, 26]). *For an UWSTS $\mathcal{W} = (S, T, \preceq, I, F)$, we define its ideal completion $\overline{\mathcal{W}} = (\overline{S}, \overline{T}, \subseteq, \overline{I}, \overline{F})$, where $(\overline{S}, \subseteq)$ is the ideal completion of $(S, \preceq)$, the transition relation is defined by*

$$\text{Succ}_{\overline{\mathcal{W}}}(X, a) = \text{ID-DEC}_S(\downarrow \text{Succ}_{\mathcal{W}}(X, a)) \ ,$$

$\overline{I} = \text{ID-DEC}_S(\downarrow I)$, and $\overline{F} = \{X \in \overline{S} \mid X \cap F \neq \emptyset\}$.

Using upward compatibility in $\mathcal{W}$, language equivalence indeed holds and determinism is preserved.

Lemma 4.15. *The ideal completion $\overline{\mathcal{W}}$ of an UWSTS $\mathcal{W}$ is a weak UWSTS. We have $\mathcal{L}(\overline{\mathcal{W}}) = \mathcal{L}(\mathcal{W})$. If $\mathcal{W}$ is deterministic, so is $\overline{\mathcal{W}}$.*

As a matter of fact, $\overline{\mathcal{W}}$ is even finitely branching, but we do not need this property.

The purpose of using ideal completions is to make it easier to find inductive invariants that are finitely represented. Assume the given UWSTS $\mathcal{W}$ has an inductive invariant X, not necessarily finitely represented. By definition, X is downward-closed. Thus, by Lemma 4.12, X is a *finite* union of ideals. These ideal are configurations of the ideal completion $\overline{\mathcal{W}}$. To turn ID-DEC$_S(X)$ into an inductive invariant of $\overline{\mathcal{W}}$, it remains to take the downward closure of the set. As the order among ideals is inclusion, this does not add configurations. In short, an inductive invariant for $\mathcal{W}$ induces a finitely-represented inductive invariant for $\overline{\mathcal{W}}$.

Proposition 4.16. *If $X \subseteq S$ is an inductive invariant of $\mathcal{W}$, $\downarrow$ID-DEC$_S(X)$ is a finitely-represented inductive invariant of $\overline{\mathcal{W}}$.*

Theorem 4.6 expects invariants for UWSTS of a particular shape, namely products $\mathcal{W} \times \mathcal{W}'$. We now show that the operation of ideal completion commutes with taking products of UWSTS, a fact that will be key to the proof of Theorem 4.2. We start by recalling that the ideals in a product wqo $X \times Y$ are precisely the products of the ideals in X and in Y.

Lemma 4.17 ([26, 35, 40]). *A set $Z \subseteq X \times Y$ is an ideal iff $Z = I \times J$, where $I \subseteq X$ and $J \subseteq Y$ are ideals.*

Lemma 4.17 yields the mentioned commutativity.

Lemma 4.18. *For two UWSTSes $\mathcal{W}$ and $\mathcal{W}'$, the weak UWSTS $\overline{\mathcal{W}} \times \overline{\mathcal{W}'}$ and $\overline{\mathcal{W} \times \mathcal{W}'}$ are isomorphic.*

Proof of Theorem 4.2. We are prepared to apply Theorem 4.6 once more in order to establish our second main result. Consider two disjoint UWSTS $\mathcal{W} = (S, T, \leq, I, F)$ and $\mathcal{W}' = (S', T', \leq', I', F')$ such that $\mathcal{W}'$ is finitely branching. By Lemma 3.4 we can assume wlog. that $\mathcal{W}'$ is deterministic.

We would like to construct a finitely-represented inductive invariant in the synchronized product of the ideal completions $\overline{\mathcal{W}} \times \overline{\mathcal{W}'}$ and then apply Theorem 4.6. Indeed, by Lemma 4.15 we know that the ideal completions are disjoint weak UWSTS, and that the latter one is still deterministic, so they satisfy the assumptions.

Relying on Lemma 4.18 we prefer to show the existence of a finitely-represented inductive invariant in

$$\overline{\mathcal{W} \times \mathcal{W}'}.$$

Using Proposition 4.16, it is sufficient to find any inductive invariant in $\mathcal{W} \times \mathcal{W}'$, it does not have to be finitely-represented. We know that such an inductive invariant exists by Lemma 4.5, since we assume $\mathcal{L}(\mathcal{W} \times \mathcal{W}') = \mathcal{L}(\mathcal{W}) \cap \mathcal{L}(\mathcal{W}') = \emptyset$. □

5 Separator Size: The Case of Petri Nets

The UWSTS associated to Petri nets are always finitely branching. Hence, Theorem 4.2 applies. Whenever the covering languages of two Petri nets are disjoint, they are regularly separable. We now show how to construct a triply-exponential non-deterministic finite automaton (NFA) separating two such languages, provided they are disjoint. Moreover, for deterministic finite automata (DFA), we show that this size cannot be avoided.

Theorem 5.1. *Let N_1 and N_2 be Petri nets. If their covering languages are disjoint, there is an NFA $\mathcal{A}$ of size triply exponential in $|N_1| + |N_2|$ such that $\mathcal{L}(A)$ separates $\mathcal{L}(N_1)$ and $\mathcal{L}(N_2)$.*

Theorem 5.2. *Covering languages cannot be separated by DFA of less than triply-exponential size.*

Instead of invoking Theorem 4.2, which uses Lemma 3.4 to determinize, we directly show how to construct an equivalent input in which one of the nets is deterministic. In this setting, the upper bound is proven as Proposition 5.4. The construction of the automaton combines Theorem 4.6 with a size estimation for an ideal decomposition. Proposition 5.7 then shows how to handle non-determinism and proves Theorem 5.1.

The lower bound is proven as Proposition 5.10. We combine a classical result from automata theory, showing that minimal DFA may have exponentially many states, with a Petri net construction due to Lipton. The result is a family of pairs of languages that are essentially complements of each other and cannot be separated by DFA of less than triply-exponential size.

Petri Nets. We introduce Petri nets the way we need them for our development. A *Petri net* over the alphabet Σ is a tuple $N = (P, T, F, \lambda, M_0, M_f)$ where P is a finite set of places, T is a finite set of transitions with $P \cap T = \emptyset$, $F: (P \cup T) \times (P \cup T) \to \mathbb{N}$ is a flow function, and $\lambda: T \to \Sigma$ is a labeling of the transitions. The runtime behavior of Petri nets is defined in terms of so-called *markings* from $M \in \mathbb{N}^d$ with $d = |P|$. If $M(p) = k > 0$, we also say place p carries k tokens or that place p is marked. We assume to be given an initial and a final marking, $M_0, M_f \in \mathbb{N}^d$.

Markings are changed by firing transitions. Formally, a transition $t \in T$ is *enabled* in marking $M \in \mathbb{N}^d$, if $M(p) \geq F(p, t)$ for all places p. An enabled transition can be *fired* leading to the marking M' with $M'(p) = M(p) - F(p, t) + F(t, p)$, denoted $M[t\rangle M'$. Note that enableness and firing are upward compatible with the componentwise ordering $\leq$ on markings, in the following sense. If $M_1 \leq M_2$ and $M_1[t\rangle M_1'$, then $M_2[t\rangle M_2'$ with $M_1' \leq M_2'$.

With this compatibility, we define the *UWSTS induced by N* to be $\mathcal{W}_N = (\mathbb{N}^P, T', \leq, \{M_0\}, \uparrow M_f)$. The transition relation is defined via firing, $(M, a, M') \in T'$ if there is a transition $t \in T$ such that $M[t\rangle M'$ and $\lambda(t) = a$. The language of $\mathcal{W}_N$ is also called the *(covering) language of N*, and denoted by $\mathcal{L}(N)$. If the language of a finite automaton $\mathcal{L}(A)$ separates $\mathcal{L}(\mathcal{W}_{N_1})$ and $\mathcal{L}(\mathcal{W}_{N_2})$, we also say that A *separates N_1 and N_2*. Similarly, we say N_1 and N_2 are *disjoint* if so are their languages. We call N *deterministic* if $\mathcal{W}_N$ is.

We will make use of a *product operation* on Petri nets. Consider $N_i = (P_i, T_i, F_i, \lambda_i, M_{0,i}, M_{f,i})$, $i = 1, 2$. The product Petri net is obtained by putting the places of N_1 and N_2 side by side and creating a new transition for all pairs of transitions in $T_1 \times T_2$ that carry the same label. Formally, $N_1 \times N_2 = (P, T, F, \lambda, M_0, M_f)$ with $P = P_1 \uplus P_2$, $T = \{(t_1, t_2) \in T_1 \times T_2 \mid \lambda(t_1) = \lambda(t_2)\}$. We have $\lambda(t_1, t_2) = \lambda(t_1) = \lambda(t_2)$. The flow function is defined by the flow functions of the component Petri nets, $F(p, (t_1, t_2)) = F_x(p, t_x)$ and $F((t_1, t_2), p) = F_x(t_x, p)$, where $x = i$ if $p \in P_i$. We have $M_0(p) = M_{0,i}(p)$ for $p \in P_i$, and similar for M_f. The product operation on Petri nets coincides with the product on UWSTS. This allows us to identify a Petri net with its induced UWSTS and carry out product constructions as required.

Lemma 5.3. $\mathcal{W}_{N_1 \times N_2}$ *is isomorphic to* $\mathcal{W}_{N_1} \times \mathcal{W}_{N_2}$.

We will need the size of a Petri net. It is defined using a binary encoding of the values in the range of the flow function and in the markings. Define the *infinity norm* of a vector $M \in \mathbb{N}^d$ to be $\|M\|_\infty = \max_{p \in P} M(p)$. We extend this notion to matrices, sets of vectors, and functions by taking the maximum over all entries, elements, and elements in the range, respectively. The *size* of the Petri net N is now

$$|N| = |P| \, |T| \, (1 + \lceil \log_2(1 + \|F\|_\infty) \rceil) + |M_0| + |M_f| \, .$$

The size of a marking M is $|M| = |P|(1 + \lceil \log_2(1 + \|M\|_\infty) \rceil)$.

We start by proving the upper bound in a special case.

An Upper Bound Assuming Determinism. Theorem 4.6 assumes that one of the UWSTS is deterministic. We now show that for Petri nets, in this case, the regular separator is (an NFA of size) at most doubly exponential in the size of the input Petri nets. From this NFA, a triply exponential DFA can be obtained by the powerset construction.

Proposition 5.4. *Let N_1 be an arbitrary Petri net and let N_2 be deterministic. If N_1 and N_2 are disjoint, they can be separated by an NFA of size doubly exponential in $|N_1| + |N_2|$.*

To prove the result, we show how a size estimation for the basis of $\text{REACH}_{\mathcal{W}}^{-1}$ with $\mathcal{W} = \mathcal{W}_{N_1 \times N_2}$ can be turned into a size estimation for the ideal decomposition of the complement. The size estimation of the basis is the following result. It is obtained by inspecting Abdulla's backward search [1].

Theorem 5.5 (Bozzelli & Ganty [11]). *Consider a Petri net N with final marking M_f. Then $\text{REACH}_{\mathcal{W}_N}^{-1} = \uparrow\{v_1, \ldots, v_k\}$, where k as well as $\|\{v_1, \ldots, v_k\}\|_\infty$ are upper bounded by*

$$g = \left(|T| \cdot \left(\|F\|_\infty + \|M_0\|_\infty + \|M_f\|_\infty + 2 \right) \right)^{2^{O(|P| \cdot \log|P|)}} .$$

By Lemma 4.5, $\mathbb{N}^d \setminus \text{REACH}_{\mathcal{W}}^{-1}$ is an inductive invariant of $\mathcal{W}$ (provided the language is empty). We can now apply Lemma 4.12 to finitely represent this set by its ideal decomposition. To represent this ideal decomposition in turn, we have to explicitly represent ideals in $\mathbb{N}^d$. The following lemma gives such a representation.

Let $\mathbb{N}_\omega$ denote $\mathbb{N}$ extended by a new top element ω. Every ideal in $\mathbb{N}^d$ is the downward closure $\downarrow u$ of a single vector $u \in \mathbb{N}_\omega^d$. The lemma moreover shows that the intersection of two ideals $\downarrow u_1$ and $\downarrow u_2$ is an ideal represented by the minimum (taken componentwise) of u_1 and u_2. Finally, the ideal decomposition of the complement of $\uparrow v$ is the union of the ideals $\downarrow u_{<v(j)}, j \in [1, d]$, where $\downarrow u_{<v(j)}$ is smaller than v in component j and ω everywhere else.

Lemma 5.6 (see e.g. [39]). *(1) The ideals in $\mathbb{N}^d$ have the shape $\downarrow u$ for $u \in \mathbb{N}_\omega^d$. (2) For two ideals $\downarrow u_1, \downarrow u_2$ of $\mathbb{N}^d$, the intersection is $\downarrow u_1 \cap \downarrow u_2 = \downarrow u$ with $u(i) = \min\{u_1(i), u_2(i)\}$. (3) For $v \in \mathbb{N}^d$, we have*

$$\text{ID-DEC}(\mathbb{N}^d \setminus \uparrow v) = \{\downarrow u_{<v(j)} \mid j \in [1..d]\} \, ,$$

where $u_{<v(j)}(j) = v(j) - 1$ and $u_{<v(j)}(i) = \omega$ for $i \neq j$.

We extend the infinity norm to $\mathbb{N}_\omega^d$ by treating ω-components as zero. Note that $\|u\|_\infty \leq \|\{u_1, u_2\}\|_\infty$ and $\|u_{<v(j)}\|_\infty \leq \|v\|_\infty$.

We are now prepared to prove the upper bound.

Proof of Proposition 5.4. Let N_1 and N_2 be the given Petri nets with a total of $d \in \mathbb{N}$ places. Let $\mathcal{W} = \mathcal{W}_{N_1 \times N_2}$. Since $\mathcal{L}(\mathcal{W})$ is empty,

$X = \mathbb{N}^d \setminus \text{REACH}_{\mathcal{W}}^{-1}$ is an inductive invariant of $\mathcal{W}$ by the proof of Lemma 4.5. By Proposition 4.16, $\downarrow Y$ with $Y = \text{ID-DEC}_{\mathbb{N}^d}(X)$ is a finitely-represented inductive invariant in the ideal completion $\overline{\mathcal{W}}$. By Lemma 5.3 and Lemma 4.18, we have

$$\overline{\mathcal{W}_{N_1 \times N_2}} \overset{iso}{=} \overline{\mathcal{W}_{N_1} \times \mathcal{W}_{N_2}} \overset{iso}{=} \overline{\mathcal{W}_{N_1}} \times \overline{\mathcal{W}_{N_2}} \, .$$

Hence, as N_2 is deterministic, we can construct a separating finite automaton with states Y by Theorem 4.6.

It remains to prove that the cardinality of Y is at most doubly-exponential. To this end, we invoke Theorem 5.5 and consider a representation $\text{REACH}_{\mathcal{W}}^{-1} = \uparrow\{v_1, \ldots, v_k\}$. We have

$$X = \mathbb{N}^d \setminus \uparrow\{v_1, \ldots, v_k\} = \mathbb{N}^d \setminus \bigcup_{i \in [1..k]} \uparrow v_i$$

$$= \bigcap_{i \in [1..k]} \left(\mathbb{N}^d \setminus \uparrow v_i \right) = \bigcap_{i \in [1..k]} \bigcup_{j_i \in [1..d]} \downarrow u_{<v_i(j_i)} \, ,$$

where the ideal representatives $u_{<v_i(j_i)}$ are constructed as in Lemma 5.6(3). We use distributivity to rewrite this expression as

$$\bigcap_{i \in [1..k]} \bigcup_{j_i \in [1..d]} \downarrow u_{<v_i(j_i)} = \bigcup_{\vec{j} \in [1..d]^k} \bigcap_{i \in [1..k]} \downarrow u_{<v_i(\vec{j}(i))} \, .$$

By Lemma 5.6(1) and (2), applied inductively, we obtain that each intersection

$$\bigcap_{i=1}^{k} \downarrow u_{<v_i(\vec{j}(i))} \quad \text{is an ideal} \quad \downarrow u_{\vec{j}} \, .$$

This means the ideal decomposition of X consists of at most $d^k \leq d^g$ many ideals, with g as defined in Theorem 5.5.

This bound is triply-exponential. We improve it by observing that $\|u_{\vec{j}}\|_\infty \leq \|\{v_1, \ldots, v_k\}\|_\infty \leq g$ for all $\vec{j}$. Consequently, all non-ω components are bounded by g, and we only have $(g+2)^d$ many such vectors in $\mathbb{N}_\omega^d$.

The ideal decomposition of X thus consists of at most

$$h = \left(\left(|T| \left(\|F\|_\infty + \|M_0\|_\infty + \|M_f\|_\infty + 2 \right) \right)^{2^{O(d \cdot \log d)}} + 2 \right)^d$$

many ideals. Note that even if $\|F\|_\infty + \|M_0\|_\infty + \|M_f\|_\infty$ are exponential in $|N_1| + |N_2|$ (due to the binary encoding of values), h is still doubly exponential in the size of the given Petri nets. $\square$

A General Upper Bound. The previous result yields a doubly-exponential separator in the case where N_2 is deterministic. We now show how to get rid of this assumption and construct a separator in the general case.

Proposition 5.7. *Let N_1 and N_2 be disjoint Petri nets. Then they are separable by an NFA of size triply exponential in $|N_1| + |N_2|$.*

The proof transforms N_1 and N_2 into $N_{-\lambda}$ and N_{det} so that N_{det} is deterministic, invokes Proposition 5.4, and then turns the resulting separator for $N_{-\lambda}$ and N_{det} into a separator for N_1 and N_2. The approach is inspired by [14].

Let N_2 be non-deterministic with labeling function $\lambda \colon T_2 \to \Sigma$. We define N_{det} to be a variant of N_2 that is labeled by the identity function. This means N_{det} is a Petri net over the alphabet T_2. We have $\mathcal{L}(N_2) = \lambda(\mathcal{L}(N_{det}))$, where we see λ as a homomorphism on words. We furthermore define $N_{-\lambda}$ to be the T_2-labeled Petri net obtained from N_1 as follows. For each a-labeled transition t_1 of N_1 and each a-labeled transition t of N_2, $N_{-\lambda}$ contains a t-labeled

copy t_1^t of t_1. The copy has the same input-output-behavior as t_1. Transition t_1 itself is removed. Intuitively, $N_{-\lambda}$ determines all transitions in N_2 that t_1 can synchronize with (over Σ). With this observation, we can synchronize $N_{-\lambda}$ and N_{det} (over T_2), and then apply λ — and still obtain the product language.

Lemma 5.8. $\mathcal{L}(N_1 \times N_2) = \lambda(\mathcal{L}(N_{-\lambda} \times N_{det}))$.

With this lemma, and since N_1 and N_2 are disjoint, $N_{-\lambda}$ and N_{det} have to be disjoint. As moreover N_{det} is deterministic, we can apply Proposition 5.4 and obtain a separator for $N_{-\lambda}$ and N_{det}. Let $\mathcal{A}$ be the doubly-exponential NFA over the alphabet T_2 with $\mathcal{L}(N_{-\lambda}) \subseteq \mathcal{L}(\mathcal{A})$ and $\mathcal{L}(N_{det}) \cap \mathcal{L}(\mathcal{A}) = \emptyset$.

We show how to turn $\mathcal{A}$ into a separator for N_1 and N_2. The first step is to determine the complement automaton $\mathcal{A}^C$. Note that it satisfies $\mathcal{L}(N_{det}) \subseteq \mathcal{L}(\mathcal{A}^C)$ and $\mathcal{L}(N_{-\lambda}) \cap \mathcal{L}(\mathcal{A}^C) = \emptyset$. The second step is to apply λ to $\mathcal{A}^C$. Let $\mathcal{B} = \lambda(\mathcal{A}^C)$ be the automaton obtained from $\mathcal{A}^C$ by relabeling each t-labeled transition to $\lambda(t)$. The following lemma shows that $\mathcal{B}$ is a separator. The observation that the size of $\mathcal{A}^C$ and hence the size of $\mathcal{B}$ is at most exponential in the size of $\mathcal{A}$ concludes the proof of Proposition 5.7

Lemma 5.9. $\mathcal{L}(N_2) \subseteq \mathcal{L}(\mathcal{B})$ and $\mathcal{L}(N_1) \cap \mathcal{L}(\mathcal{B}) = \emptyset$.

Note that $\lambda(\mathcal{A})$ is not a separator, as it may contain words from $\mathcal{L}(N_2)$. There might be $u \in \mathcal{L}(A)$, $u \notin \mathcal{L}(N_{det})$ s.t. there is a $u' \in \mathcal{L}(N_{det})$ with $\lambda(u) = \lambda(u')$. Thus, $\lambda(u) \in \lambda(\mathcal{L}(\mathcal{A})) \cap \mathcal{L}(N_2)$.

A Lower Bound. We now consider separation by *deterministic finite automata* (DFA). In this case, we can show a triply-exponential lower bound on the size of the separator.

Proposition 5.10. *For all $n \in \mathbb{N}$, there are disjoint Petri nets $N_0(n)$ and $N_1(n)$ of size polynomial in n such that any separating DFA has size at least triply exponential in n.*

Our proof relies on the classical result that for each $x \in \{0, 1\}$ and each $k \in \mathbb{N}$, the minimal DFA for the language

$$\mathcal{L}_{x@k} = \{w \in \{0, 1\}^{\geq k} \mid \text{ the } k\text{-last letter in } w \text{ is } x\}$$

needs at least 2^k states [38].

To obtain the desired lower bound, we will show how to generate $\mathcal{L}_{x@k}$ for a doubly-exponential number k by a polynomially-sized Petri net. To this end, we make use of Lipton's proof of EXPSPACE-hardness for coverability [41]. We will not need the precise construction, the following lemma gives a specification of Lipton's Petri nets that is enough for our purposes.

Lemma 5.11 (Lipton [41]). *For every $n \in \mathbb{N}$,*
- *there is a Petri net $N_{inc}(n)$ of size polynomial in n with places $p_{haltinc}$ and p_{out} such that any computation leading to a marking M with $M(p_{haltinc}) = 1$ has $M(p_{out}) = 2^{2^n}$.*
- *there is $N_{dec}(n)$ of size polynomial in n with places $p_{haltdec}$ and p_{in} such that there is a computation leading to a marking M with $M(p_{haltdec}) = 1$ if and only if $M_0(p_{in}) \geq 2^{2^n}$.*

We can assume all transitions in these Petri nets carry label b.

Using the lemma, we create for $x \in \{0, 1\}$ a new Petri net $N_x(n)$ over the alphabet $\{0, 1, b, c\}$ whose language is essentially $\mathcal{L}_{x@2^{2^n}}$.

A computation of $N_x(n)$ consists of four successive phases. Four so-called control places $p_1, \ldots, p_4$ indicate the current phase of the computation, and all transitions of a specific phase check that the corresponding control place is marked.

1. In the first phase, $N_{inc}(n)$ is used to create 2^{2^n} tokens on p_{out} and one token on $p_{haltinc}$. The phase ends by firing a c-labeled transition that checks for a token on $p_{haltinc}$ and moves the token from control place p_1 to p_2. Note that all transitions are labeled b up to this point.
2. In the second phase, there are two transitions labeled by 0 resp. 1 which only check that the control place is marked. They create an arbitrary sequence in $\{0, 1\}^*$, corresponding to the part of the word before the last 2^{2^n} letters.
x. The second phase ends with an x-labeled transition that moves the token from control place p_2 to p_3. It consumes one token from p_{out} and generates one token on p_{in}.
3. In the third phase, there are two transitions labeled by 0 resp. 1, each moving one token from p_{out} to p_{in}. The phase ends by firing a c-labeled transition that moves the token from p_3 to p_4.
4. In the fourth phase, $N_{dec}(n)$ is used to check that the number of tokens on p_{in} at the beginning of the phase is at least 2^{2^n}.

The initial marking for $N_x(n)$ assigns a token to p_1 as well as the necessary initial tokens to $N_{inc}(n)$ and $N_{dec}(n)$. The final marking requires a token on p_4 and a token on the place $p_{haltdec}$ of $N_{dec}(n)$.

We claim that the language of $N_x(n)$ is

$$\mathcal{L}(N_x(n)) = \mathcal{L}'.c.\mathcal{L}_{x@2^{2^n}}.c.\mathcal{L}'' \qquad \text{with } \mathcal{L}', \mathcal{L}'' \subseteq b^* .$$

With the control places $p_1, \ldots, p_4$, words in $\mathcal{L}(N_x(n))$ clearly have the shape $w_{inc}.c.w.c.w_{dec}$, where w_{inc} and w_{dec} are computations in Lipton's Petri nets. We argue that the 2^{2^n}-last letter in w is x. After running $N_{inc}(n)$, we have 2^{2^n} tokens on p_{out}. This means the x-labeled transition that ends the second phase has to be fired at most 2^{2^n} letters before the end of w, because any transition fired in the third phase consumes a token from p_{out}. Since any transition fired during the third phase also produces a token on p_{in}, and we check for 2^{2^n} tokens on this place in phase four, the x-labeled transition has to be fired at least 2^{2^n} letters before the end of w.

Since $N_{inc}(n)$ and $N_{dec}(n)$ are of size polynomial in n, and in $N_x(n)$ we only add a constant number of transitions and places, and a polynomial number of entries to the flow function, it is clear that $N_x(n)$ is also polynomially-sized.

It remains to argue that $\mathcal{L}(N_0(n))$ and $\mathcal{L}(N_1(n))$ cannot be separated by a DFA of less than triply-exponential size. Intuitively, the separator has to incorporate a DFA for the language $\mathcal{L}_{0@2^{2^n}}$. We make this precise in the form of Proposition D.1.

6 Conclusion

We have shown that, under mild assumptions, disjointness of WSTS languages implies their regular separability. In particular, we have shown that if one of two disjoint upward-compatible WSTS is finitely branching, they are regularly separable. Using our expressibility results, it is also sufficient if the underlying order for one of the two is an ω^2-wqo. A similar result holds for downward-compatible WSTS assuming that one of them is deterministic or the underlying order is an ω^2-wqo. As WSTSes are typically ω^2-WSTS (e.g. Petri nets and their extensions, LCS), our result already implies the decidability of regular separability for almost all WSTS of practical relevance.

Our work brings together research on inductive invariants and regular separability. We show that a finite representation of an inductive invariant for the product system can be transformed into

a regular separator. For Petri nets, one may use any representation of the coverability set, e.g. obtained from the coverability graph [36]. As we show, it is beneficial in terms of the worst-case size, to use an inductive invariant obtained from the backwards coverability algorithm [1]. For LCS, it is known that the coverability set is not computable [45], but one can still obtain a finitely-represented inductive invariant e.g. from the EEC-algorithm [29].

We leave some questions without answer. For instance, it is not clear whether the assumptions of Theorems 4.2 and 4.1 are necessary; we were neither able to drop the assumptions, nor to provide a counterexample. Similarly, we do not know whether the inclusions in Theorem 3.1 are strict. Finally, in the case of Petri nets, closing the gap between the triply-exponential size of the NFA separator and the triply-exponential lower bound for DFA remains an open problem.

As future work, one could consider the *well-behaved transition systems* of [9], a generalization of WSTS where only the finite antichain property is required. Our proofs do not work for WBTS, as they are not closed under Cartesian product.

References

[1] P. A. Abdulla, K. Cerans, B. Jonsson, and Y.-K. Tsay. General decidability theorems for infinite-state systems. In *LICS*, pages 313–321, 1996.

[2] P. A. Abdulla, G. Delzanno, and L. Van Begin. Comparing the expressive power of well-structured transition systems. In *CSL*, pages 99–114, 2007.

[3] P. A. Abdulla, G. Delzanno, O. Rezine, A. Sangnier, and R. Traverso. On the verification of timed ad hoc networks. In *FORMATS*, volume 6919 of *LNCS*, pages 256–270. Springer, 2011.

[4] P. A. Abdulla, J. Deneux, and P. Mahata. Multi-clock timed networks. In *LICS*, pages 345–354. IEEE, 2004.

[5] P. A. Abdulla and B. Jonsson. Verifying programs with unreliable channels. In *LICS*, pages 160–170. IEEE, 1993.

[6] M. F. Atig, A. Bouajjani, S. Burckhardt, and M. Musuvathi. On the verification problem for weak memory models. In *POPL*, pages 7–18. ACM, 2010.

[7] M. F. Atig, A. Bouajjani, S. Burckhardt, and M. Musuvathi. What's decidable about weak memory models? In *ESOP*, volume 7211 of *LNCS*, pages 26–46. Springer, 2012.

[8] M. Blondin, A. Finkel, and P. McKenzie. Handling infinitely branching WSTS. In *ICALP*, pages 13–25, 2014.

[9] M. Blondin, A. Finkel, and P. McKenzie. Well behaved transition systems. *Logical Methods in Computer Science*, 13(3), 2017.

[10] A. Bouajjani and R. Mayr. Model checking lossy vector addition systems. In *STACS*, pages 323–333, 1999.

[11] L. Bozzelli and P. Ganty. Complexity analysis of the backward coverability algorithm for VASS. In *RP*, pages 96–109, 2011.

[12] D. Brand and P. Zafiropulo. On communicating finite-state machines. *J. ACM*, 30(2):323–342, 1983.

[13] N. Busi, M. Gabbrielli, and G. Zavattaro. Comparing recursion, replication, and iteration in process calculi. In *ICALP*, volume 3142 of *LNCS*, pages 307–319. Springer, 2004.

[14] L. Clemente, W. Czerwiński, S. Lasota, and C. Paperman. Regular separability of Parikh automata. In *ICALP*, pages 117:1–117:13, 2017.

[15] L. Clemente, W. Czerwiński, S. Lasota, and C. Paperman. Separability of reachability sets of vector addition systems. In *STACS 2017*, pages 24:1–24:14, 2017.

[16] S. A. Cook. Soundness and completeness of an axiom system for program verification. *SIAM J. Comput.*, 7(1):70–90, 1978.

[17] W. Czerwiński and S. Lasota. Regular separability of one counter automata. In *LICS*, pages 1–12, 2017.

[18] W. Czerwiński, W. Martens, and T. Masopust. Efficient separability of regular languages by subsequences and suffixes. In *ICALP*, pages 150–161, 2013.

[19] G. Delzanno and F. Rosa-Velardo. On the coverability and reachability languages of monotonic extensions of Petri nets. *Theoretical Computer Science*, 467:12 – 29, 2013.

[20] E. D'Osualdo. *Verification of Message Passing Concurrent Systems*. PhD thesis, University of Oxford, 2015.

[21] C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset nets between decidability and undecidability. In *ICALP*, volume 1443 of *LNCS*, pages 103–115. Springer, 1998.

[22] J. Esparza. Decidability and complexity of Petri net problems - an introduction. In *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets*, pages 374–428. Springer, 1998.

[23] A. Finkel. A generalization of the procedure of Karp and Miller to well structured transition systems. In *ICALP*, pages 499–508, 1987.

[24] A. Finkel. Reduction and covering of infinite reachability trees. *Inf. Comput.*, 89(2):144–179, 1990.

[25] A. Finkel and J. Goubault-Larrecq. Forward analysis for wsts, part I: completions. In *STACS*, pages 433–444, 2009.

[26] A. Finkel and J. Goubault-Larrecq. Forward analysis for wsts, part II: complete WSTS. *Logical Methods in Computer Science*, 8(3), 2012.

[27] A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1-2):63–92, 2001.

[28] W. Gasarch. A survey of recursive combinatorics. In *Handbook of recursive mathematics, Vol. 2*, volume 139 of *Stud. Logic Found. Math.*, page 1041âĂŞ1176. Amsterdam: North-Holland, 1998.

[29] G. Geeraerts, J.-F. Raskin, and L. Van Begin. Expand, enlarge and check: New algorithms for the coverability problem of wsts. *Journal of Computer and System Sciences*, 72(1):180 – 203, 2006.

[30] G. Geeraerts, J.-F. Raskin, and L. Van Begin. Well-Structured Languages. *Acta Informatica*, 44(3-4):249–288, 2007.

[31] M. Heizmann, J. Hoenicke, and A. Podelski. Nested interpolants. In *POPL*, pages 471–482. ACM, 2010.

[32] H. B. Hunt III. On the decidability of grammar problems. *Journal of the ACM*, 29(2):429–447, 1982.

[33] P. Jancar. A note on well quasi-orderings for powersets. *Inf. Process. Lett.*, 72(5-6):155–160, 1999.

[34] S. Joshi and B. König. Applying the graph minor theorem to the verification of graph transformation systems. In *CAV*, volume 5123 of *LNCS*, pages 214–226. Springer, 2008.

[35] M. Kabil and M. Pouzet. Une extension d'un théorème de p. jullien sur les âges de mots. *ITA*, 26:449–484, 1992.

[36] R. M. Karp and R. E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969.

[37] E. Kopczynski. Invisible pushdown languages. In *LICS*, pages 867–872, 2016.

[38] D. Kozen. *Automata and computability*. Undergraduate texts in Computer Science. Springer, 1997.

[39] R. Lazic and S. Schmitz. The ideal view on rackoff's coverability technique. In *RP*, pages 76–88, 2015.

[40] J. Leroux and S. Schmitz. Demystifying reachability in vector addition systems. In *LICS*, pages 56–67, 2015.

[41] R. J. Lipton. The reachability problem requires exponential space. Technical report, Yale University, Department of Computer Science, 1976.

[42] Z. Manna and A. Pnueli. *Temporal verification of reactive systems - safety*. Springer, 1995.

[43] A. Marcone. Foundations of bqo theory. *Transactions of the American Mathematical Society*, 345(2):641–660, 1994.

[44] M. Martos-Salgado and F. Rosa-Velardo. Dynamic networks of timed Petri nets. In *Petri nets*, pages 294–313, 2014.

[45] R. Mayr. Undecidable problems in unreliable computations. *Theoretical Computer Science*, 1-3(297), 2003.

[46] R. Meyer. On boundedness in depth in the pi-calculus. In *TCS*, volume 273 of *IFIP*, pages 477–489. Springer, 2008.

[47] M. Mukund, K. N. Kumar, J. Radhakrishnan, and M. A. Sohoni. Robust asynchronous protocols are finite-state. In *ICALP*, pages 188–199, 1998.

[48] M. Mukund, K. N. Kumar, J. Radhakrishnan, and M. A. Sohoni. Towards a characterisation of finite-state message-passing systems. In *ASIAN*, pages 282–299, 1998.

[49] T. Place, L. van Rooijen, and M. Zeitoun. Separating regular languages by locally testable and locally threshold testable languages. In *FSTTCS*, pages 363–375, 2013.

[50] T. Place, L. van Rooijen, and M. Zeitoun. Separating regular languages by piecewise testable and unambiguous languages. In *MFCS*, pages 729–740, 2013.

[51] T. Place and M. Zeitoun. Going higher in the first-order quantifier alternation hierarchy on words. In *ICALP*, pages 342–353, 2014.

[52] T. Place and M. Zeitoun. Separating regular languages with first-order logic. *Logical Methods in Computer Science*, 12(1), 2016.

[53] F. Rosa-Velardo and M. Martos-Salgado. Multiset rewriting for the verification of depth-bounded processes with name binding. *Inf. Comput.*, 215:68–87, 2012.

[54] Sylvain Schmitz and Philippe Schnoebelen. Multiply-recursive upper bounds with higman's lemma. In *ICALP*, volume 6756 of *LNCS*, pages 441–452. Springer, 2011.

[55] T. G. Szymanski and J. H. Williams. Noncanonical extensions of bottom-up parsing techniques. *SIAM Journal on Computing*, 5(2):231–250, 1976.

[56] T. Wies, D. Zufferey, and T. A. Henzinger. Forward analysis of depth-bounded processes. In *FOSSACS*, volume 6014 of *LNCS*, pages 94–108. Springer, 2010.

[57] M. De Wulf, L. Doyen, T. A. Henzinger, and J.-F. Raskin. Antichains: A new algorithm for checking universality of finite automata. In *CAV*, volume 4144 of *LNCS*, pages 17–30. Springer, 2006.

A Details for Section 2

Proof of Lemma 2.1. We first assume that $\left(\mathcal{P}_{\text{fin}}^{\downarrow}(X), \subseteq\right)$ is a wqo and prove that $(X, \preceq)$ is a wqo. Consider any infinite sequence $x_1, x_2 \ldots$, where each $x_i \in X$. By our assumption, in the infinite sequence $\downarrow\{x_1\}, \downarrow\{x_2\}, \ldots$ we find $i < j$ such that $\downarrow\{x_i\} \subseteq \downarrow\{x_j\}$. We conclude $x_i \preceq x_j$ as desired.

For the other direction, we will assume that $(X, \preceq)$ is a wqo. Using Higmann's lemma, also $(X^*, \preceq^*)$ is a wqo. Here X^* is set of all finite sequences over X and $\preceq^*$ is the subsequence order, i.e. $w \preceq^* v$ if w is obtained from v by deleting symbols and/or replacing them by smaller symbols with respect to $\preceq$. Now consider any infinite sequence $X_1, X_2, \ldots$ in $\mathcal{P}_{\text{fin}}^{\downarrow}(X)$. By definition, each X_i can be written as $\downarrow\{u_1^i, \ldots, u_{n_i}^i\}$ for appropriately chosen u_j^i. We represent each X_i by $u_1^i \ldots u_{n_i}^i \in X^*$ and consider the sequence

$$u_1^1 \ldots u_{n_1}^1, u_1^2 \ldots u_{n_2}^2, \ldots$$

in X^*. Using the fact that $(X^*, \preceq^*)$ is a wqo, we obtain $i < j$ such that $u_1^i \ldots u_{n_i}^i \preceq^* u_1^j \ldots u_{n_j}^j$. From this, we immediately obtain $X_i \subseteq X_j$. □

Proof of Lemma 2.3. We claim that for all $w \in \Sigma^*$, $s \preceq s'$ and $s \xrightarrow{w} r$ implies $s' \xrightarrow{w} r'$ for some $r' \in S$ with $r \preceq r'$.

We proceed by induction on w and use upward compatibility. In the base case $w = \varepsilon$, there is nothing to prove.

Let us now consider a word $w.a$ and $s \xrightarrow{w} t \xrightarrow{a} r$. Let $s \in S$ with $s \preceq s'$. By induction, there is a t' such that $s' \xrightarrow{w} t'$ and $t \preceq t'$. By the upward compatibility of UWSTS, we get that there is some r' with $t' \xrightarrow{a} r'$ with $r \preceq r'$ as required. □

Proof of Lemma 2.4. We claim that for each $w \in \Sigma^*$, $s \preceq s'$ and $s \xrightarrow{w} r$ in $\downarrow \mathcal{W}$ implies $s' \xrightarrow{w} r'$ in $\mathcal{W}$ for some r' with $r \preceq r'$.

We proceed by induction on w. In the base case, the statement follows from the fact that the initial configurations of $\downarrow \mathcal{W}$ are the configurations in $\downarrow I$.

Now consider a word $w.a$. such that $s \xrightarrow{w} t \xrightarrow{a} r$ in $\downarrow \mathcal{W}$. By induction, there is t' such that $s' \xrightarrow{w} t'$ in $\mathcal{W}$ and $t \preceq t'$. From the definition of transition in $\downarrow \mathcal{W}$, we have that $t \xrightarrow{a} r''$ in $\mathcal{W}$ for some r'' with $r \preceq r''$. Since $t \preceq t'$, we can apply upward compatibility to obtain $t' \xrightarrow{a} r'$ in $\mathcal{W}$ with $r'' \preceq r'$. We conclude $s' \xrightarrow{w} t' \xrightarrow{a} r'$ in $\mathcal{W}$ with $r \preceq r'' \preceq r'$ as desired. □

B Details for Section 3

Proof of Lemma 3.3. Let $\mathcal{W} = (S, T, \preceq, I, F)$ be an ω^2-DWSTS. Wlog. we assume that $\mathcal{W} = \uparrow \mathcal{W}$, i.e., the initial configurations and the transition relation in $\mathcal{W}$ are upward closed. A deterministic DWSTS $\overline{\mathcal{W}} = (\overline{S}, \overline{T}, \supseteq, \overline{I}, \overline{F})$ equivalent to $\mathcal{W}$ can be defined, essentially by a powerset construction on $\mathcal{W}$, similar to the proof of Lemma 3.2. Let the configurations $\overline{S} = \mathcal{P}^{\uparrow}(S)$ be the upward closed subsets of S, ordered by the superset relation $\supseteq$.

Using the fact that $(\mathcal{P}^{\uparrow}(S), \supseteq)$ and $(\mathcal{P}^{\downarrow}(S), \subseteq)$ are isomorphic and Lemma 2.2, we obtain that $(\mathcal{P}^{\uparrow}(S), \supseteq)$ is a wqo. Let the initial configuration of $\overline{\mathcal{W}}$ be the set I, i.e. $\overline{I} - \{I\}$. The accepting configurations are those sets containing an accepting configuration in $\mathcal{W}$, $\overline{F} = \{X \in \overline{S} \mid X \cap F \neq \emptyset\}$.

The transition relation $\overline{T}$ is defined by the direct image:

$$X \xrightarrow{a} \text{Succ}_{\mathcal{W}}(X, a)$$

which is well-defined as $\text{Succ}_{\mathcal{W}}(X, a)$ is upward closed for $X \in \mathcal{P}^{\uparrow}(S)$. The equality of the languages of $\mathcal{W}$ and $\overline{\mathcal{W}}$ follows directly from the following claim:

Claim: For every $w \in \Sigma^*$, $\text{Reach}_{\overline{\mathcal{W}}}(w) = \{\text{Reach}_{\mathcal{W}}(w)\}$.

The claim is shown by induction on the length of w. The induction base is $\overline{I} = \{I\}$, and the inductive step follows by (1) combined with the equality $\text{Succ}_{\overline{\mathcal{W}}}(X, a) = \{\text{Succ}_{\mathcal{W}}(X, a)\}$. □

Proof of Lemma 3.4. Given a finitely-branching UWSTS $\mathcal{W} = (S, T, \preceq, I, F)$ we define a deterministic one $\overline{\mathcal{W}} = (\overline{S}, \overline{T}, \subseteq, \overline{I}, \overline{F})$ by a powerset construction. We proceed similarly as in the proof of Lemma 3.2, but using the finitely-represented downward closed subsets of S instead of all such sets.

Let configurations be downward closures of finite subsets of S, $\overline{S} = \mathcal{P}_{\text{fin}}^{\downarrow}(S)$. They are ordered by inclusion $\subseteq$, which is a wqo by Lemma 2.1. Define the unique initial configuration of $\overline{\mathcal{W}}$ as $\overline{I} = \{\downarrow I\}$, and note that I is finite. The final configurations of $\overline{\mathcal{W}}$ are the sets containing a final configuration from $\mathcal{W}$, $\overline{F} = \{X \in \mathcal{P}_{\text{fin}}^{\downarrow}(S) \mid X \cap F \neq \emptyset\}$. Let the deterministic transition relation be defined by direct image,

$$X \xrightarrow{a} \downarrow\text{Succ}_{\mathcal{W}}(X, a).$$

Note that $X \in \mathcal{P}_{\text{fin}}^{\downarrow}(S)$ implies $\text{Succ}_{\mathcal{W}}(X, a) \in \mathcal{P}_{\text{fin}}^{\downarrow}(S)$, as $\mathcal{W}$ is assumed to be finitely branching and satisfies upward compatibility. The equality of the language of $\mathcal{W}$ and the language of $\overline{\mathcal{W}}$ follows by the following claim and the fact that $\mathcal{W}$ and $\downarrow \mathcal{W}$ are language-equivalent:

Claim: For every $w \in \Sigma^*$, $\text{Reach}_{\overline{\mathcal{W}}}(w) = \left\{\text{Reach}_{\downarrow \mathcal{W}}(w)\right\}$.

We proceed by induction on w.
In the base case, we have $\text{Reach}_{\overline{\mathcal{W}}}(\varepsilon) = \{\downarrow I\} = \left\{\text{Reach}_{\downarrow \mathcal{W}}(w)\right\}$ as desired.

Let us now consider some word $w.a$. Using induction and the definition of the transition relation in $\overline{\mathcal{W}}$, we have

$$\begin{aligned}
\text{Reach}_{\overline{\mathcal{W}}}(w.a) &= \text{Succ}_{\overline{\mathcal{W}}}(\text{Reach}_{\overline{\mathcal{W}}}(w), a) \\
&= \text{Succ}_{\overline{\mathcal{W}}}(\{\text{Reach}_{\downarrow W}(w)\}, a) \\
&= \left\{\downarrow\text{Succ}_{\mathcal{W}}(\text{Reach}_{\downarrow W}(w), a)\right\}
\end{aligned}$$

We claim that we indeed have

$$\downarrow\text{Succ}_{\mathcal{W}}(\text{Reach}_{\downarrow W}(w), a) = \text{Reach}_{\downarrow \mathcal{W}}(w.a).$$

Let $s \in S$ be such that $s \preceq s'$ for some $s' \in \text{Succ}_{\mathcal{W}}(\text{Reach}_{\downarrow W}(w), a)$. This means $s' \xrightarrow{a} t'$ in $\mathcal{W}$ for some $t' \in \text{Reach}_{\downarrow W}(w)$. By the definition of the transition relation of $\downarrow \mathcal{W}$, we conclude $s \xrightarrow{a} t$ in $\downarrow \mathcal{W}$ which implies $s \in \text{Reach}_{\downarrow \mathcal{W}}(w.a)$.

Let $s \in \text{Reach}_{\downarrow \mathcal{W}}(w.a)$, then we have $s \xrightarrow{a} t$ in $\downarrow \mathcal{W}$ for some $t \in \text{Reach}_{\downarrow \mathcal{W}}(w)$. By the definition of the transition relation of $\downarrow \mathcal{W}$, we have that there is some s' with $s \preceq s'$ and $s' \xrightarrow{a} t$ in $\mathcal{W}$. We obtain $s' \in \text{Succ}_{\mathcal{W}}(\text{Reach}_{\downarrow W}(w), a)$ and conclude $s \in \downarrow\text{Succ}_{\mathcal{W}}(\text{Reach}_{\downarrow W}(w), a)$.

This proves the claim, showing that the language of $\overline{\mathcal{W}}$ is equal to the language of $\downarrow \mathcal{W}$, which in turn is equal to the language of $\mathcal{W}$. □

Proof of Lemma 3.6. Given an ω^2-UWSTS $\mathcal{W} = (S, T, \leq, I, F)$, we define a deterministic DWSTS $\overline{\mathcal{W}} = (\overline{S}, \overline{T}, \overline{\leq}, \overline{I}, \overline{F})$ as follows:

Configurations are upward closed subsets of S, i.e. $\overline{S} = \mathcal{P}^{\uparrow}(S)$, ordered by the superset relation, the reverse of inclusion: $U \overline{\leq} V$ iff $U \supseteq V$. This order is isomorphic to the order $(\mathcal{P}^{\downarrow}(S), \subseteq)$ of downward closed subset ordered by inclusion, hence a wqo by Lemma 2.2. There is one initial configuration in $\overline{\mathcal{W}}$, namely $\overline{I} = \{F\}$. An upward closed subset $U \in \mathcal{P}^{\uparrow}(S)$ is final, i.e. $U \in \overline{F}$, if $U \cap I \neq \emptyset$, i.e. if U contains some initial configuration from I. The set $\overline{F}$ is downward closed as required; indeed, if $U \in \overline{F}$ and $U \subseteq V$ then necessarily $V \in \overline{F}$ too. The deterministic transition relation is defined using the pre-image:

$$U \xrightarrow{a} \text{PRED}_{\mathcal{W}}(U, a) .$$

(Note that $\text{PRED}_{\mathcal{W}}(U, a)$ is upward closed whenever U is.) Finally, we verify that $\overline{\leq}$ satisfies the downward compatibility condition: If $U \subseteq V$, then $\text{PRED}_{\mathcal{W}}(U, a) \subseteq \text{PRED}_{\mathcal{W}}(V, a)$.

From the following claim, we deduce that the language of $\mathcal{W}$ is the reverse of the language of $\overline{\mathcal{W}}$.

Claim: For every $w \in \Sigma^*$, $\text{REACH}_{\overline{\mathcal{W}}}(w) = \left\{ \text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w)) \right\}$.

We prove the claim by induction on w. In the base case, we have

$$\text{REACH}_{\overline{\mathcal{W}}}(\varepsilon) = \overline{I} = \{F\} = \left\{ \text{REACH}_{\mathcal{W}}^{-1}(w) \right\} .$$

Now consider a word $w.a$ and note that $\text{rev}(w.a) = a.\text{rev}(w)$. Using induction and the definition of $\overline{T}$, we have

$$\begin{aligned}
\text{REACH}_{\overline{\mathcal{W}}}(w.a) &\overset{(1)}{=} \text{SUCC}_{\overline{\mathcal{W}}}(a, \text{REACH}_{\overline{\mathcal{W}}}(w)) \\
&= \text{SUCC}_{\overline{\mathcal{W}}}(a, \{\text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w))\}) \\
&= \text{PRED}_{\mathcal{W}}(a, \{\text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w))\}) \\
&\overset{(2)}{=} \{\text{REACH}_{\mathcal{W}}^{-1}(a.\text{rev}(w))\} .
\end{aligned}$$

Thus, $w \in \mathcal{L}(\overline{\mathcal{W}})$ iff (by the above claim) $\text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w)) \in \overline{F}$ iff $\text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w)) \cap I \neq \emptyset$ iff $\text{REACH}_{\mathcal{W}}(\text{rev}(w)) \cap F \neq \emptyset$ iff $\text{rev}(w) \in \mathcal{L}(\mathcal{W})$. □

Proof of Lemma 3.7. Let $\mathcal{W} = (S, T, \leq, I, F)$ be the given ω^2-DWSTS, we show how to construct the required deterministic UWSTS $\overline{\mathcal{W}} = (\overline{S}, \overline{T}, \subseteq, \overline{I}, \overline{F})$ as follows: Configurations are downward closed subsets of S, i.e. $\overline{S} = \mathcal{P}^{\downarrow}(S)$, ordered by inclusion $\subseteq$. By Lemma 2.2, $(\mathcal{P}^{\downarrow}(S), \subseteq)$ is a wqo.

The initial configuration is given by $\overline{I} = \{F\}$. Note that by definition F is a downward closed set. A downward closed set $D \in \overline{S}$ is final if it contains an initial configuration, i.e.

$$\overline{F} = \{D \in \mathcal{P}^{\downarrow}(S) \mid D \in \overline{S} \wedge D \cap I \neq \emptyset\}$$

Note that for any $V, U \subseteq V$ for some $U \in \overline{F}$ implies $V \cap I \neq \emptyset$, hence $V \in \overline{F}$. Hence $\overline{F}$ is upward closed as required. The deterministic transition relation is given by

$$U \xrightarrow{a} \text{PRED}_{\mathcal{W}}(U, a)$$

If U is downward closed then $\text{PRED}_{\mathcal{W}}(U, a)$ is also downward closed by downward-compatibility.

Finally, we show that $\overline{\leq}$ satisfies upward compatibility. For this, let U, V be such that $U \subseteq V$. We need to prove that if $U \xrightarrow{a} U'$, then there is a V' with $V \xrightarrow{a} V'$ and $U' \subseteq V'$. This follows from $U' = \text{PRED}_{\mathcal{W}}(U, a) \subseteq \text{PRED}_{\mathcal{W}}(V, a) = V'$.

The equality of the language of $\mathcal{W}$ and the language of $\overline{\mathcal{W}}$ follows by the following claim:

Claim: For every $w \in \Sigma^*$, $\text{REACH}_{\overline{\mathcal{W}}}(w) = \left\{ \text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w)) \right\}$.

The proof of the claim is similar to the one in the proof of Lemma 3.6. Thus, $w \in \mathcal{L}(\overline{\mathcal{W}})$ iff (by the above claim) $\text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w)) \in \overline{F}$ iff $\text{REACH}_{\mathcal{W}}^{-1}(\text{rev}(w)) \cap I \neq \emptyset$ iff $\text{REACH}_{\mathcal{W}}(\text{rev}(w)) \cap F \neq \emptyset$ iff $\text{rev}(w) \in \mathcal{L}(\mathcal{W})$. □

C Details for Section 4

Proof of Lemma 4.8.
(1) We show that for every $s \in \text{REACH}_{\mathcal{W}}(w)$, there is some $(s, s') \in \text{REACH}_{\mathcal{W}_{\times}}(w)$.

Indeed, $s \in \text{REACH}_{\mathcal{W}}(w)$ implies $(s, s') \in \text{REACH}_{\mathcal{W}_{\times}}(w)$ for $s' = \text{REACH}_{\mathcal{W}'}(w)$.
(2) We show that for every $(s, s') \in \text{REACH}_{\mathcal{W}_{\times}}(w)$, there is some $(r, r') \in \text{REACH}_{\mathcal{A}}(w)$ with $(s, s') \leq_{\times} (r, r')$.

We proceed by induction on w. In the base case, we have that any $(s, s') \in I_{\times} = \text{REACH}_{\mathcal{W}_{\times}}(\varepsilon)$ is dominated by some $(r, r') \in Q$ since $I \subseteq \downarrow Q$, Property 6. By the definition of Q_I, we have $(r, r') \in Q_I = \text{REACH}_{\mathcal{A}}(\varepsilon)$.

Now consider $(s, s') \in \text{REACH}_{\mathcal{W}_{\times}}(w.a)$. By definition, there is $(\tilde{s}, \tilde{s}') \in \text{REACH}_{\mathcal{W}_{\times}}(w)$ and $(\tilde{s}, \tilde{s}') \xrightarrow{a} (s, s')$ in $\mathcal{W}_{\times}$. Applying induction, we obtain $(\tilde{r}, \tilde{r}') \in \text{REACH}_{\mathcal{A}}(w)$ with $(\tilde{s}, \tilde{s}') \leq_{\times} (\tilde{r}, \tilde{r}')$. Using the upward-compatibility of $\mathcal{W}_{\times}$, there is a transition $(\tilde{r}, \tilde{r}') \xrightarrow{a} (t, t')$ in $\mathcal{W}_{\times}$ such that $(s, s') \leq_{\times} (t, t')$. Since $(r, r') \in Q$, and $\downarrow Q$ is closed under taking successors in $\mathcal{W}_{\times}$, Property 8, we have that there is $(r, r') \in Q$ such that $(t, t') \leq_{\times} (r, r')$. We have $(s, s') \leq_{\times} (r, r')$ be transitivity. To complete the proof, it remains to argue that there is a transition $(\tilde{r}, \tilde{r}') \xrightarrow{a} (r, r')$ in $\mathcal{A}$. To this end, we instantiate the definition of the transition relation, using $(t, t') \leq_{\times} (r, r')$. □

Proof of Lemma 4.10. We proceed by induction on w. The base case follows by the definition of the initial states in $\mathcal{A}$. For the induction step, consider $(s, s') \in \text{REACH}_{\mathcal{A}}(w.a)$, which means there is $(r, r') \in \text{REACH}_{\mathcal{A}}(w)$ with $(r, r') \xrightarrow{a} (s, s')$ in $\mathcal{A}$. By definition of the transition relation in the automaton, there is (t, t') such that $(r, r') \xrightarrow{a} (t, t')$ in $\mathcal{W}_{\times}$ and $(t, t') \leq_{\times} (s, s')$. By definition of the transition relation in $\mathcal{W}_{\times}$, we have $r' \xrightarrow{a} t'$ in $\mathcal{W}'$.

We apply induction to (r, r') and get $\text{REACH}_{\mathcal{W}'}(w) \leq' r'$. In $\mathcal{W}'$, we have $\text{REACH}_{\mathcal{W}'}(w) \xrightarrow{a} \text{REACH}_{\mathcal{W}'}(w.a)$. Using upward compatibility of $\mathcal{W}'$, r' can simulate this transition. Since $\mathcal{W}'$ is deterministic, it is in fact simulated by $r' \xrightarrow{a} t'$ and we conclude $\text{REACH}_{\mathcal{W}'}(w.a) \leq' t'$. Hence, $\text{REACH}_{\mathcal{W}'}(w.a) \leq' t' \leq' s'$. □

Proof of Lemma 4.15. The ideal completion is a weak UWSTS. Indeed, the set of final configurations $\overline{F}$ is upward-closed since the ideals are ordered by inclusion. The transition relation still satisfies upward compatibility by Lemma 4.13.

The language is preserved due to the following invariant:

Claim: For every $w \in \Sigma^*$, $\bigcup \text{REACH}_{\overline{\mathcal{W}}}(w) = \downarrow \text{REACH}_{\mathcal{W}}(w)$.

We prove the claim by induction on w.

In the base case, we have

$$
\begin{aligned}
\bigcup \text{Reach}_{\overline{\mathcal{W}}}(\varepsilon) &= \bigcup \overline{I} \\
&= \bigcup \text{Id-dec}_S(\downarrow I) \\
&\overset{(9)}{=} \downarrow I \\
&= \downarrow \text{Reach}_{\mathcal{W}}(\varepsilon)
\end{aligned}
$$

as required.

Consider $w.a$. Using upward compatibility, induction, and the fact that unions commute with taking the successor in $\mathcal{W}$, we obtain

$$
\begin{aligned}
\downarrow \text{Reach}_{\mathcal{W}}(w.a) &= \downarrow \text{Succ}_{\mathcal{W}}(\text{Reach}_{\mathcal{W}}(w), a) \\
&= \downarrow \text{Succ}_{\mathcal{W}}(\downarrow \text{Reach}_{\mathcal{W}}(w), a) \\
&= \downarrow \text{Succ}_{\mathcal{W}}(\bigcup \text{Reach}_{\overline{\mathcal{W}}}(w), a) \\
&= \downarrow \text{Succ}_{\mathcal{W}}(\bigcup_{Y \in \text{Reach}_{\overline{\mathcal{W}}}(w)} Y, a) \\
&= \bigcup_{Y \in \text{Reach}_{\overline{\mathcal{W}}}(w)} \downarrow \text{Succ}_{\mathcal{W}}(Y, a) .
\end{aligned}
$$

Using Equation (9), the last expression is equal to

$$
\bigcup_{Y \in \text{Reach}_{\overline{\mathcal{W}}}(w)} \bigcup \text{Id-dec}_S(\downarrow \text{Succ}_{\mathcal{W}}(Y, a)) .
$$

By the above equality and the definition of the transition relation in $\overline{\mathcal{W}}$, we obtain

$$
\begin{aligned}
&\bigcup_{Y \in \text{Reach}_{\overline{\mathcal{W}}}(w)} \bigcup \downarrow \text{Succ}_{\mathcal{W}}(Y, a) \\
&= \bigcup \bigcup_{Y \in \text{Reach}_{\overline{\mathcal{W}}}(w)} \text{Succ}_{\overline{\mathcal{W}}}(Y, a) \\
&= \bigcup \text{Succ}_{\overline{\mathcal{W}}}(\text{Reach}_{\overline{\mathcal{W}}}(w), a) \\
&= \bigcup \text{Reach}_{\overline{\mathcal{W}}}(w.a)
\end{aligned}
$$

as desired.

Suppose now that $\mathcal{W} = (S, T, \leq, I, F)$ is deterministic. In particular, $I = \{i\}$ for some $i \in S$. Then $\downarrow\{i\}$ is the unique initial configuration of $\overline{\mathcal{W}}$. To prove that $\text{Succ}_{\overline{\mathcal{W}}}(X, a)$ contains a unique element, we show that $\downarrow \text{Succ}_{\mathcal{W}}(X, a)$ is already an ideal. To this end, it is sufficient to show that whenever X is directed, $\text{Succ}_{\mathcal{W}}(X, a)$ is directed. Consider $r, r' \in \text{Succ}_{\mathcal{W}}(X, a)$. Then there are $s, s' \in X$ with $s \xrightarrow{a} r$ and $s' \xrightarrow{a} r'$. As X is directed, there is an element $\tilde{s}$ with $s \leq \tilde{s}$ and $s' \leq \tilde{s}$. Using upward compatibility, $\tilde{s}$ can simulate the transitions of s and s'. Since $\mathcal{W}$ is deterministic, there is in fact a unique $\tilde{r} \in \text{Succ}_{\mathcal{W}}(X, a)$ with $\tilde{s} \xrightarrow{a} \tilde{r}$ and $r \leq \tilde{r}$ as well as $r' \leq \tilde{r}$, as desired. □

Proof of Proposition 4.16. Define $Q = \text{Id-dec}_S(X)$. First, note that Q contains all ideals $Y \subseteq X$ that are maximal wrt. inclusion. Consequently, $\downarrow Q$ contains all ideals $Y \subseteq X$. We observe that

$$
X \overset{(9)}{=} \bigcup Q = \bigcup \downarrow Q .
$$

By Lemma 4.12, Q is finite and thus $\downarrow Q$ is finitely-represented. It remains to check that $\downarrow Q$ satisfies the Properties (6), (7), and (8).

We need to show that any maximal ideal contained in $\downarrow I$ belongs to $\downarrow Q$. Since $I \subseteq X$ by Property (6), and since X is downward-closed,

we also have $\downarrow I \subseteq X$. Consequently, any ideal contained in $\downarrow I$ is also an ideal in X, and $\downarrow Q$ contains all such ideals.

Next, we consider Property (7). Towards a contradiction, assume that $\downarrow Q$ contains an ideal Y that is final in $\overline{\mathcal{W}}$. By definition, this means Y contains a final configuration. Since $Y \subseteq X$, this final configuration would also be contained in X, a contradiction to $X \cap F = \emptyset$, Property (7).

It remains to check the inclusion $\text{Succ}_{\overline{\mathcal{W}}}(\downarrow Q, a) \subseteq \downarrow Q$. We pick an ideal $Y \in \downarrow Q$ and show $\text{Succ}_{\overline{\mathcal{W}}}(Y, a) \subseteq \downarrow Q$. Recall the definition $\text{Succ}_{\overline{\mathcal{W}}}(Y, a) = \text{Id-dec}_S(\downarrow \text{Succ}_{\mathcal{W}}(Y, a))$. Thus, any element of $\text{Succ}_{\overline{\mathcal{W}}}(Y, a)$ is an ideal of $\downarrow \text{Succ}_{\mathcal{W}}(Y, a)$.

We have $\text{Succ}_{\mathcal{W}}(X, a) \subseteq X$ by Property (8). This implies $\text{Succ}_{\mathcal{W}}(Y, a) \subseteq X$ as $Y \subseteq X$, and even $\downarrow \text{Succ}_{\mathcal{W}}(Y, a) \subseteq X$ as X is downward-closed. Thus, any ideal in $\downarrow \text{Succ}_{\mathcal{W}}(Y, a)$ is also an ideal in X, and thus contained in $\downarrow Q$. □

Proof of Lemma 4.18. With Lemma 4.17, as an isomorphism between $\overline{\mathcal{W}} \times \overline{\mathcal{W}'}$ and $\overline{\mathcal{W} \times \mathcal{W}'}$ take the function $I, J \mapsto I \times J$ that maps a pair of ideals to their product. This is an isomorphism as the transition relation in the ideal completion of a UWSTS is defined by direct image, and direct image commutes with product. □

D Details for Section 5

Proof of Lemma 5.8. We show that for the Petri nets $N_{-\lambda}$ and N_{det} that we have constructed,

$$
\mathcal{L}(N_1 \times N_2) = \lambda(\mathcal{L}(N_{-\lambda} \times N_{det}))
$$

holds.

Let w be in the left-hand side. Consider the corresponding computations u_1 and u_2 of N_1 and N_2, respectively. u_2 can be seen as a computation of N_{det} with $\lambda(u_2) = w$ as desired. Consider the computation u_1' of $N_{-\lambda}$ that is obtained as follows: If at some position i, u_1 uses transition $t_1 \in T_1$ and u_2 uses transition $t \in T_2$ (where $\lambda_1(t_1) = \lambda(t)$ has to hold), we let u_1' use transition t_1^t. Note that in $N_{-\lambda}$, this transition has label t. Indeed, u_1' synchronizes with u_2 as desired, proving $u_2 \in \mathcal{L}(N_{-\lambda} \times N_{det})$ and thus $\lambda(u_2) = w$ is in the right-hand side.

Let $\lambda(u_2)$ be in the right hand side and let u_1', u_2 be the corresponding computations. u_2 is already a computation of N_2 with labeling $\lambda(u_2)$ as desired. We define the computation u_1 of N_1 as follows: If at some positions i, u_1' uses some transition t_1^t, we define u_1 to use transition t at this position. Note that $\lambda_1(t_1) = \lambda(t)$ has to hold. The computations synchronize as desired, proving that $\lambda(u_1') = \lambda(u_2)$ is in the left-hand side. □

Proof of Lemma 5.9. We have $\mathcal{L}(N_{det}) \subseteq \mathcal{L}(\mathcal{A}^C)$ and hence

$$
\mathcal{L}(N_2) = \lambda(\mathcal{L}(N_{det})) \subseteq \lambda(\mathcal{L}(\mathcal{A}^C)) = \mathcal{L}(\lambda(\mathcal{A}^C)) = \mathcal{L}(\mathcal{B}).
$$

As for disjointness, assume $w \in \mathcal{L}(N_1) \cap \mathcal{L}(\mathcal{B})$. Then there is an accepting computation u in N_1 with $\lambda_1(u) = w$ and $v \in \mathcal{L}(\mathcal{A}^C)$ with $\lambda(v) = w$. We inductively construct a v-labeled accepting computation in $N_{-\lambda}$. This will contradict $\mathcal{L}(N_{-\lambda}) \cap \mathcal{L}(\mathcal{A}^C) = \emptyset$. Whenever u uses some transition t_1 and v uses $t \in T_2$, use the transition t_1^t of $N_{-\lambda}$. Since we have $\lambda_1(t_1) = \lambda(t)$, the transition t_1^t indeed exists in $N_{-\lambda}$. Because the behavior of t_1 and t_1^t is the same, the resulting computation of $N_{-\lambda}$ is still accepting. □

Proposition D.1. *Let $n \in \mathbb{N}$ and consider the Petri nets $N_0(n), N_1(n)$. Any DFA $\mathcal{A}$ such that $\mathcal{L}(N_0(n)) \subseteq \mathcal{L}(\mathcal{A})$ and $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(N_1(n)) = \emptyset$ needs to have at least $2^{2^{2^n}}$ many states.*

Recall that the languages of each $N_x(n)$ is

$$\mathcal{L}'.c.\mathcal{L}_{x@2^{2^n}}.c.\mathcal{L}'' .$$

Since the languages are not distinguishable in their $\mathcal{L}'$ prefix and $\mathcal{L}''$ suffix, the separator has to distinguish $\mathcal{L}_{0@2^{2^n}}$ from $\mathcal{L}_{1@2^{2^n}}$.

These languages partition $\{0,1\}^{\geq 2^{2^n}}$, so the separator has to incorporate a DFA for $\mathcal{L}_{0@2^{2^n}}$. It is a classic result from automata theory that any DFA for $\mathcal{L}_{x@m}$ needs to have at least 2^m states [38]. Intuitively, a DFA for $\mathcal{L}_{x@m}$ cannot guess the end of the word. It always needs to store the last m bits of the input that it has processed so far, for which there are 2^m possibilities. For the sake of completeness, we give a proof.

Proof. For ease of notation, we define $k = 2^{2^n}$. Assume towards a contradiction that $\mathcal{A}$ has strictly less than 2^k states. We consider the set $\mathcal{B} = \{0,1\}^k$ of all sequences over $\{0,1\}$ of length exactly k. We have that $|\mathcal{B}| = 2^k$.

Let $w_{inc} \in b^*$ be a word corresponding to computation for $N_{inc}(n)$ that creates k many tokens on places p_{out} and one token on $p_{haltinc}$. Similarly, let $w_{dec} \in b^*$ be a word corresponding to a computation for $N_{dec}(n)$ that creates a token on $p_{haltdec}$ (assuming that p_{in} contains at least k tokens).

For any $w = w_1 \ldots w_k \in \mathcal{B}$, $w_{inc}.c.w.c.w_{dec}$ is contained in the language of $N_x(n)$ for exactly one $x \in \{0,1\}$, namely for $x = w_1$. For each $w \in \mathcal{B}$, we denote by q_w the unique control state in which the DFA $\mathcal{A}$ is after processing $w_{inc}.c.w$. Since $\mathcal{A}$ has strictly less than 2^k states, but $\mathcal{B}$ has 2^k elements, there are distinct $w, w' \in \mathcal{B}$ such that $q_w = q_{w'}$. Since $w \neq w'$, there is some bit $i \geq 1$ such that $w_i \neq w_i'$. We assume without loss of generality $w_i = 0, w_i' = 1$. Let us define $w_{fill} = 0^{i-1}$. Since $w_{inc}.c.w$ and $w_{inc}.c.w'$ lead to the same state, also $w_{inc}.c.w.w_{fill}.c.w_{dec}$ and $w_{inc}.c.w'.w_{fill}.c.w_{dec}$ lead to the same state. Thus, either both or none of these words is accepted by $\mathcal{A}$. To complete the proof, note that the length $i-1$ of w_{fill} was chosen such that w_i is the k-last bit of $w.w_{fill}$. Therefore, we have $w_{inc}.c.w.w_{fill}.c.w_{dec} \in \mathcal{L}(N_0(n)) \subseteq \mathcal{L}(\mathcal{A})$ since the k-last bit is 0, and $w_{inc}.c.w'.w_{fill}.c.w_{dec} \in \mathcal{L}(N_1(n))$ since the k-last bit is 1. We conclude $w_{inc}.c.w'.w_{fill}.c.w_{dec} \notin \mathcal{L}(\mathcal{A})$ since $\mathcal{L}(N_1(n)) \cap \mathcal{L}(\mathcal{A}) = \emptyset$, a contradiction. $\square$