Branch-Well-Structured Transition Systems and Extensions

Benedikt Bollig[1], Alain Finkel[1,2], and Amrita Suresh[1](✉)

[1] Université Paris-Saclay, ENS Paris-Saclay, CNRS, LMF, Gif-sur-Yvette, France
amrita.suresh@ens-paris-saclay.fr

[2] Institut Universitaire de France, Paris, France

Abstract. We propose a relaxation to the definition of a well-structured transition systems (WSTS) while retaining the decidability of boundedness and termination. In this class, we ease the well-quasi-ordered (wqo) condition to be applicable only between states that are reachable one from another. Furthermore, we also relax the monotony condition in the same way. While this retains the decidability of termination and boundedness, it appears that the coverability problem is undecidable. To this end, we define a new notion of monotony, called cover-monotony, which is strictly more general than the usual monotony and still allows to decide a restricted form of the coverability problem.

Keywords: Verification · Decidability · Coverability · Termination · Well-quasi-ordering

1 Introduction

Well-structured transition systems (WSTS) (initially called structured transition systems in [10]) have decidable termination and boundedness problems. They capture properties common to a wide range of formal models used in model-checking, system verification and concurrent programming [13].

A WSTS is an infinite set X (of states) with a transition relation $\rightarrow \subseteq X \times X$. The set X is quasi-ordered by $\leq$, and $\rightarrow$ fulfills one of various possible monotonies with respect to $\leq$. The quasi-ordering of X is further assumed to be well, i.e. well-founded and with no infinite antichains (see Sect. 2 for precise formal definitions). These two properties lead to a general framework in which it is possible to algorithmically decide verification problems like coverability, termination and boundedness.

This class of systems includes Lossy Channel Systems, Petri Nets and their extensions, among others [1,13]. More recently, the theory of WSTS has been applied to study computational models resulting from a combination of different types of systems like asynchronous systems defined by extending pushdown systems with an external memory [5], cryptographic protocols [7], and others.

Various strengthenings and weakenings of the notion of monotony (of $\rightarrow$ with respect to $\leq$) were introduced, to allow WSTS to capture more models [1,13].

M. R. Mousavi and A. Philippou (Eds.): FORTE 2022, LNCS 13273, pp. 50–66, 2022.
https://doi.org/10.1007/978-3-031-08679-3_4

More recently, [3] showed that the wellness assumption in the definition of WSTS can be relaxed while some decidabilities are retained (notably, the coverability problem is decidable).

Our main contribution is to prove that the monotony and well-quasi-order (wqo) assumptions can further be weakened while some problems remain decidable. More precisely, we introduce a notion of well-structured transition systems, called branch-well-structured transition systems, where the monotony is only applicable to states reachable one from another. Furthermore, we also relax the wqo condition to such states. With this relaxation, it is still possible to retain the decidability of termination and boundedness. Furthermore, for the coverability problem, we introduce a notion of monotony, called cover-monotony, which still allows deciding the coverability problem, even in the absence of strong (or strict or transitive or reflexive) monotony. Indeed, while the usual backward algorithm for coverability relies on well-foundedness, the forward algorithm described in [3] does not require that property.

Outline. Sect. 2 introduces terminology and some well-known results concerning well-quasi-orderings and well-structured transition systems. Section 3 defines branch-WSTS, and shows that both the boundedness and the termination problems are decidable for such systems. Section 4 investigates the coverability problem for WSTS with relaxed conditions. We conclude in Sect. 5. Due to space constraints, some proofs are omitted.

2 Preliminaries

Quasi-Orderings. Let X be a set and $\leq \subseteq X \times X$ be a binary relation over X, which we also write as $(X, \leq)$. We call $\leq$ a *quasi-ordering (qo)* if it is reflexive and transitive. As usual, we call $\leq$ a *partial ordering* if it is a qo and anti-symmetric (if $x \leq y$ and $y \leq x$, then $x = y$).

For the following definitions, we also use the terminology qo for the ordering $\leq$ and its associated set X, i.e. $(X, \leq)$.

We write $x < y$ if $x \leq y$ and $y \not\leq x$. If $\leq$ is a partial ordering, $x < y$ is then equivalent to $x \leq y$ and $x \neq y$.

To any $x \in X$, we associate the sets $\uparrow x \overset{\text{def}}{=} \{y \mid x \leq y\}$ and $\downarrow x \overset{\text{def}}{=} \{y \mid y \leq x\}$. Moreover, for $A \subseteq X$, we let $\uparrow A \overset{\text{def}}{=} \bigcup_{x \in A} \uparrow x$ and $\downarrow A \overset{\text{def}}{=} \bigcup_{x \in A} \downarrow x$. We say that A is *upward-closed* if $A = \uparrow A$. Similarly, A is *downward-closed* if $A = \downarrow A$. A *basis* of an upward-closed set A is a set $B \subseteq X$ such that $A = \uparrow B$.

We call $(X, \leq)$ *well-founded* if there is no infinite strictly decreasing sequence $x_0 > x_1 > \dots$ of elements of X. An *antichain* is a subset $A \subseteq X$ of pairwise incomparable elements, i.e., for every distinct $x, y \in A$, we have $x \not\leq y$ and $y \not\leq x$. For example, consider the alphabet $\Sigma = \{a, b\}$. There exists an infinite antichain $\{b, ab, aab, ...\}$ with respect to the prefix ordering over Σ^*.

An *ideal* is a downward-closed set $I \subseteq X$ that is also *directed*, i.e., it is nonempty and, for every $x, y \in I$, there exists $z \in I$ such that $x \leq z$ and $y \leq z$. The set of ideals is denoted by $\mathsf{Ideals}(X)$.

Well-Quasi-Orderings. When a qo satisfies some additional property, we deal with a well-quasi-ordering:

Definition 1. *A* well-quasi-ordering (wqo) *is a qo* $(X, \leq)$ *such that every infinite sequence* $x_0, x_1, x_2, \ldots$ *over* X *contains an* increasing pair*, i.e., there are* $i < j$ *such that* $x_i \leq x_j$*.*

For example, the set of natural numbers $\mathbb{N}$, along with the standard ordering $\leq$ is a wqo. Moreover, $(\mathbb{N}^k, \leq)$, i.e. the set of vectors of $k \geq 1$ natural numbers with component-wise ordering, is a wqo [6]. On the other hand, the prefix ordering of words over an alphabet Σ, denoted by $\preceq$, is not a wqo since, in the infinite sequence $b, ab, a^2b, a^3b, \ldots a^nb, \ldots$, we have $a^ib \not\preceq a^jb$ for all $i < j$.

In general, for qo, upward-closed sets do not necessarily have a *finite* basis. However, from [14], we know that every upward-closed set in a wqo has a finite basis.

We have the following equivalent characterization of wqos.

Proposition 1 **([9]).** *A qo* $(X, \leq)$ *is a wqo iff every infinite sequence in* X *has an infinite increasing subsequence.*

Moreover, one can prove that a qo is a wqo iff it is well-founded and contains no infinite antichain.

The following proposition is useful to design the forward coverability algorithm that enumerates finite subsets of ideals composing inductive invariants. It shows that the wqo hypothesis is not necessary to decide coverability.

Proposition 2 **([9]).** *A qo* $(X, \leq)$ *contains no infinite antichain iff every downward-closed set decomposes into a finite union of ideals.*

Transition Systems. A *transition system* is a pair $\mathcal{S} = \langle X, \rightarrow \rangle$ where X is the set of states and $\rightarrow \subseteq X \times X$ is the transition relation. We write $x \rightarrow y$ for $(x, y) \in \rightarrow$. Moreover, we let $\xrightarrow{*}$ be the transitive and reflexive closure of the relation $\rightarrow$, and $\xrightarrow{+}$ be the transitive closure of $\rightarrow$.

Given a state $x \in X$, we write $Post_{\mathcal{S}}(x) = \{y \in X \mid x \rightarrow y\}$ for the set of immediate successors of x. Similarly, $Pre_{\mathcal{S}}(x) = \{y \in X \mid y \rightarrow x\}$ denotes the set of its immediate predecessors.

We call $\mathcal{S}$ *finitely branching* if, for all $x \in X$, the set $Post_{\mathcal{S}}(x)$ is finite. The *reachability set* of $\mathcal{S}$ from $x \in X$ is defined as $Post^*_{\mathcal{S}}(x) = \{y \in X \mid x \xrightarrow{*} y\}$. Note that, when $\mathcal{S}$ is clear from the context, we may drop the subscript and write, e.g., $Post^*(x)$. We say that a state y is reachable from x if $y \in Post^*(x)$ (resp. $y \in \downarrow Post^*(x)$).

A *(well-)ordered transition system* is a triple $\mathcal{S} = (X, \rightarrow, \leq)$ consisting of a transition system $\langle X, \rightarrow \rangle$ equipped with a qo (resp., wqo) $(X, \leq)$. An ordered transition system $\mathcal{S} = (X, \rightarrow, \leq)$ is *effective* if $\leq$ and $\rightarrow$ are decidable. We say that a state y is coverable from x if $y \in \downarrow Post^*(x)$.

Definition 2 **([10]).** *A* well-structured transition system (WSTS) *is a well-ordered transition system $\mathcal{S} = (X, \to, \leq)$ that satisfies (general)* monotony*: for all $x, y, x' \in X$, we have: $x \leq y \wedge x \to x' \implies \exists y' \in X$: $x' \leq y' \wedge y \xrightarrow{*} y'$.*

We define other types of monotony. We say that a well-ordered transition system $\mathcal{S} = (X, \to, \leq)$ satisfies *strong monotony* (resp., *transitive monotony*) if, for all $x, y, x' \in X$ such that $x \leq y$ and $x \to x'$, there is $y' \in X$ such that $x' \leq y'$ and $y \to y'$ (resp., $y \xrightarrow{+} y'$). The transition system $\mathcal{S}$ satisfies *strict monotony* if, for all $x, y, x' \in X$ such that $x < y$ and $x \to x'$, there is $y' \in X$ such that $x' < y'$ and $y \to y'$.

Definition 3. *We define the following decision problems. Given an ordered transition system $\mathcal{S} = (X, \to, \leq)$ and an initial state $x_0 \in X$:*

- The non-termination problem*: Is there an infinite sequence of states $x_1, x_2, \ldots$ such that $x_0 \to x_1 \to x_2 \to \ldots$?*
- The boundedness problem*: Is $Post^*_{\mathcal{S}}(x_0)$ finite?*
- The coverability problem*: Given states $x, y \in X$, is y coverable from x?*

It is folklore [10,13] that termination is decidable for finitely branching WSTS with transitive monotony and that boundedness is decidable for finitely branching WSTS $\mathcal{S} = (X, \to, \leq)$ where $\leq$ is a partial ordering and $\to$ is strictly monotone; in both cases, we suppose that the WSTS are effective and that $Post(x)$ is computable, for all $x \in X$.

Recall that, in a wqo $(X, \leq)$, upward-closed sets have a finite basis. Coverability is decidable for a large class of WSTS:

Theorem 1 **([1,13]).** *The coverability problem is decidable for effective WSTS $\mathcal{S} = (X, \to, \leq)$ equipped with an algorithm that, for all finite sets $I \subseteq X$, computes a finite basis $pb(I)$ of $\uparrow Pre(\uparrow I)$.*

Assume $\mathcal{S} = (X, \to, \leq)$ is a WSTS and $x \in X$ is a state. The *backward coverability algorithm* involves computing (a finite basis of) $Pre^*(\uparrow x)$ as the limit of the infinite increasing sequence $\uparrow I_0 \subseteq \uparrow I_1 \subseteq \ldots$ where $I_0 = \{x\}$ and $I_{n+1} \overset{\text{def}}{=} I_n \cup pb(I_n)$. Since there exists an integer k such that $\uparrow I_{k+1} = \uparrow I_k$, the finite set I_k is computable (one may test, for all n, whether $\uparrow I_{n+1} = \uparrow I_n$) and I_k is then a finite basis of $Pre^*(\uparrow x)$ so one deduces that coverability is decidable.

Coverability can be also decided by using the *forward coverability algorithm* that relies on two semi-decision procedures (as described below). It applies to the class of well-behaved transition systems, which are more general than WSTS. A *well-behaved transition system (WBTS)* is an ordered transition system $\mathcal{S} = (X, \to, \leq)$ with monotony such that $(X, \leq)$ contains no infinite antichain. We describe effectiveness hypotheses that allow manipulating downward-closed sets in WBTS.

Definition 4 **([3, Definition 3.4]).** *A class C of WBTS is* ideally effective *if, given $\mathcal{S} = (X, \to, \leq) \in C$,*

- *the set of encodings of* $\mathsf{Ideals}(X)$ *is recursive,*
- *the function mapping the encoding of a state* $x \in X$ *to the encoding of the ideal* $\downarrow x \in \mathsf{Ideals}(X)$ *is computable;*
- *inclusion of ideals of* X *is decidable;*
- *the downward closure* $\downarrow Post(I)$ *expressed as a finite union of ideals is computable from the ideal* $I \in \mathsf{Ideals}(X)$*.*

Theorem 2 ([3]). *The coverability problem is decidable for ideally effective WBTS.*

The proof is done by two semi-decision procedures where downward-closed sets are represented by their finite decomposition in ideals and this is effective. Procedure 1 checks for coverability of y from x_0, by recursively computing $\downarrow x_0$, $\downarrow(\downarrow x_0 \cup Post(\downarrow x_0))$ and so on. This procedure terminates only if y belongs to one of these sets, hence it terminates if y is coverable. Hence, we deduce:

Proposition 3 ([3]). *For an ideally effective WBTS* $\mathcal{S} = (X, \rightarrow, \leq)$*, an initial state* x_0*, and a state* y*, Procedure 1 terminates iff* y *is coverable from* x_0*.*

Procedure 1 : Checks for a coverability certificate of y from x_0

```
input: $\mathcal{S} = (X, \rightarrow, \leq)$ and $x_0, y$
  $D := \downarrow x_0$
  while $y \notin D$ do
    $D := \downarrow(D \cup Post_{\mathcal{S}}(D))$
  end while
  return "$y$ is coverable from $x_0$"
```

Procedure 2 enumerates all downward-closed subsets (by means of their finite decomposition in ideals) in some fixed order $D_1, D_2, \ldots$ such that for all i, $D_i \subseteq X$ and $\downarrow Post(D_i) \subseteq D_i$. This enumeration is effective since $\mathcal{S}$ is ideally effective. If such a set D_i contains x_0, it is an over-approximation of $Post^*(x_0)$. Hence, if there is such a set D_i such that $x_0 \in D_i$ but $y \notin D_i$, it is a certificate of non-coverability. Moreover, this procedure terminates if y is non-coverable because $\downarrow Post^*(x_0)$ is such a set, and hence, will eventually be found.

Proposition 4 ([3]). *For a WBTS* $\mathcal{S} = (X, \rightarrow, \leq)$*, an initial state* x_0 *and a state* y*, Procedure 2 terminates iff* y *is not coverable from* x_0*.*

3 Termination and Boundedness

In this section, we generalize wqo and monotony such that these properties only consider states along a branch in the reachability tree. To define these notions, we use labels on the transitions, hence, we consider labeled transition systems.

Procedure 2 : Checks for non-coverability

```
input: $\mathcal{S} = (X, \rightarrow, \leq)$ and $x_0, y$
  enumerate $D_1, D_2, \ldots$
  $i := 1$
  while $\neg(\downarrow Post(D_i) \subseteq D_i$ and $x_0 \in D_i$ and $y \notin D_i)$ do
    $i := i + 1$
  end while
  return false
```

Labeled Transition Systems. A *labeled transition system (LTS)* is a tuple $\mathcal{S} = (X, \Sigma, \rightarrow, x_0)$ where X is the set of states, Σ is the finite action alphabet, $\rightarrow \subseteq X \times \Sigma \times X$ is the transition relation, and $x_0 \in X$ is the initial state.

Definition 5. *An* (quasi-)ordered labeled transition system (OLTS) *is defined as a tuple* $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ *where* $(X, \Sigma, \rightarrow, x_0)$ *is an LTS and* $(X, \leq)$ *is a qo.*

In the case of an LTS or OLTS, we write $x \xrightarrow{a} x'$ instead of $(x, a, x') \in \rightarrow$. For $\sigma \in \Sigma^*$, $x \xrightarrow{\sigma} x'$ is defined as expected. We also let $x \rightarrow x'$ if $(x, a, x') \in \rightarrow$ for some $a \in \Sigma$, with closures $\xrightarrow{*}$ and $\xrightarrow{+}$.

We call an OLTS $\mathcal{S}$ *effective* if $\leq$ and, for all $a \in \Sigma$, $\xrightarrow{a}$ are decidable.

Remark 1. We can similarly define a labeled WSTS as an OLTS such that the ordering is well and it satisfies the general monotony condition (canonically adapted to take care of the transition labels). Moreover, we lift the decision problems from Definition 3 to OLTS in the obvious way.

Branch-WSTS. Consider an OLTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$. A *run* (or *branch*) of $\mathcal{S}$ is a finite or infinite sequence $\rho = (x_0 \rightarrow x_1)(x_1 \rightarrow x_2)...$ simply written $\rho = x_0 \rightarrow x_1 \rightarrow x_2 \ldots$. We say that ρ is *branch-wqo* if the set of states $\{x_0, x_1, x_2, \ldots\}$ visited along ρ is wqo w.r.t. $\leq$.

Definition 6. *An OLTS* $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ *is branch-wqo if every run of* $\mathcal{S}$ *is branch-wqo.*

Example 1. Consider the FIFO machine (formally defined in Definition 10) $\mathcal{M}_1$ in Fig. 1 with one FIFO channel. In control-state q_0, it makes a loop by sending letter a to the channel. Then, we may go, non-deterministically, to control-state q_1 by sending letter b once, and then we stop. Let us consider the set of states $X_1 = \{q_0, q_1\} \times \{a, b\}^*$ together with the ordering $\leq_p$ defined by $(q, u) \leq_p (q', u')$ if $q = q'$ and u is a prefix of u', i.e., $u \preceq u'$. The reachability set of $\mathcal{M}_1$ from (q_0, ε) is equal to $\{(q_0, w), (q_1, w') \mid w \in a^*, w' \in a^*b\}$. Note that $\leq_p$ is not a wqo since elements of the set $\{(q_1, w) \mid w \in a^*b\}$ form an infinite antichain for $\leq_p$. However, the reachability tree of $\mathcal{M}_1$ is branch-wqo for the initial state (q_0, ε). Hence, there exist branch-wqo OLTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ such that $(X, \leq)$ is not a wqo.

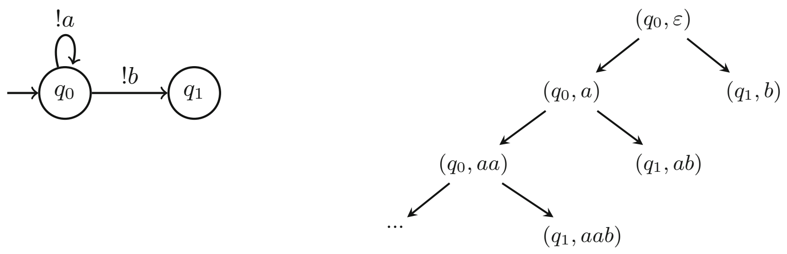

Fig. 1. The FIFO machine $\mathcal{M}_1$ (left), and its corresponding (incomplete) infinite reachability tree (right).

Remark 2. There exist a system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ and $x_0' \in X$ such that $\mathcal{S}$ is branch-wqo but $(X, \Sigma, \rightarrow, \leq, x_0')$ is not branch-wqo (cf. Figure 2).

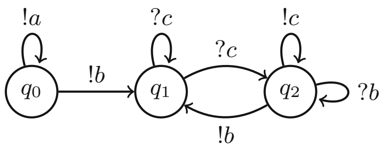

Fig. 2. The FIFO machine shown is branch-wqo if the initial control-state is q_0. If the initial control-state is q_2, then it is not branch-wqo as the states in the set $\{(q_1, w) \mid w \in c^+b\}$, which form an infinite antichain, are reachable from (q_2, ε).

We now look at a generalization of *strong monotony*, which we will refer to as branch-monotony.

Definition 7 (Branch-monotony). *An OLTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ is* branch-monotone *if, for all $x, x' \in X$, $\sigma \in \Sigma^*$ such that $x \xrightarrow{\sigma} x'$ and $x \leq x'$, there exists a state y such that $x' \xrightarrow{\sigma} y$ and $x' \leq y$.*

Remark 3. Let $\mathcal{S}$ be a branch-monotone OLTS and let there be states x, x' such that $x \xrightarrow{\sigma} x'$ and $x \leq x'$, with $\sigma \in \Sigma^*$. Then, for any $n \geq 1$, there exists $y_n \in X$ such that $x \xrightarrow{\sigma^n} y_n$ with $x \leq y_n$.

As in the case of general monotony, *strict* branch-monotony is defined using strict inequalities in both cases.

Example 2. Consider $\mathcal{M}_1$ from Example 1 once again. Note $\mathcal{M}_1$ induces an OLTS by considering the actions on the edges to be the labels. Moreover, $\mathcal{M}_1$ is branch-monotone. For every $x \xrightarrow{\sigma} x'$ such that $x \leq x'$ and $\sigma \in \Sigma^*$, it is necessary that $x = (q_0, a^n)$, $x' = (q_0, a^{n+k})$, for some $n, k \in \mathbb{N}$. Moreover, there always exists a transition from x' such that $x' \xrightarrow{\sigma} y = (q_0, a^{n+k+k})$. Hence, $x' \leq y$. We deduce that $\mathcal{M}_1$ is branch-monotone.

$$\begin{tikzcd}
 & (q_0, \varepsilon) \arrow[dl] \arrow[dr] & \\
(q_0, a) & & (q_1, b) \\
\text{dead} & &
\end{tikzcd}$$

Fig. 3. The reduced reachability tree of $\mathcal{M}_1$ from (q_0, ε). Note that (q_0, a) is dead because it is subsumed by state (q_0, ε). As a matter of fact, we have $(q_0, \varepsilon) \xrightarrow{*} (q_0, a)$ and $(q_0, \varepsilon) \leq_p (q_0, a)$. State (q_1, b) is also dead but it is not subsumed.

We are now ready to extend the definition of WSTS.

Definition 8 (Branch-WSTS). *A* branch-WSTS *is an OLTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ that is finitely branching, branch-monotone, and branch-wqo.*

When we say, without ambiguity, that a machine $\mathcal{M}$ is branch-wqo, WSTS, or branch-WSTS, we mean that the ordered transition system $\mathcal{S}_{\mathcal{M}}$, associated with machine $\mathcal{M}$, is branch-wqo, WSTS, or branch-WSTS, resp.

Remark 4. Branch-WSTS is a strict superclass of labeled WSTS. For example, machine $\mathcal{M}_1$ is branch-WSTS for the ordering $\leq_p$ but $\mathcal{M}_1$ is not WSTS for $\leq_p$ since $\leq_p$ is not a wqo on $\{q_0, q_1\} \times \{a, b\}^*$ or on the subset $\{(q_1, w) \mid w \in a^*b\}$.

Let us recall the *Reduced Reachability Tree (RRT)*, which was defined as Finite Reachability Tree in [10,13]. Suppose that $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ is an OLTS. Then, the *Reduced Reachability Tree* from x_0, denoted by $RRT(\mathcal{S}, x_0)$, is a tree where nodes are labeled by states of X, and $n(x)$ denotes that node n is labeled by state x. Nodes are either *dead* or *live*. The root node $n_0(x_0)$ is live. A dead node has no child node. A live node $n(y)$ has one child $n'(y')$ for each successor $y' \in Post_{\mathcal{S}}(y)$. If there is a path in the tree $n_0(x_0) \xrightarrow{*} n'(y') \xrightarrow{+} n(y)$ such that $n' \neq n$ and $y' \leq y$, we say that n' *subsumes* n, and then n is dead. Otherwise n is live. See Fig. 3 for the RRT of $\mathcal{M}_1$.

Proposition 5. *Let $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ be an OLTS that is finitely branching and branch-wqo. Then, $RRT(\mathcal{S}, x_0)$ is finite.*

Proposition 6. *Let $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ be a branch-WSTS, equipped with strict branch-monotony and such that $\leq$ is a partial ordering. The reachability set $Post^*_{\mathcal{S}}(x_0)$ is infinite iff there exists a branch $n_0(x_0) \xrightarrow{*} n_1(x_1) \xrightarrow{+} n_2(x_2)$ in $RRT(\mathcal{S}, x_0)$ such that $x_1 < x_2$.*

We now need a notion of effectivity adapted to branch-WSTS.

Definition 9. *A branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ is* branch-effective *if $\mathcal{S}$ is effective and $Post_{\mathcal{S}}(x)$ is a (finite) computable set, for all $x \in X$.*

Theorem 3. *Boundedness is decidable for branch-effective branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ with strict branch-monotony such that $\leq$ is a partial ordering.*

Proof. Suppose $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ satisfies the above conditions. From Proposition 5, we obtain that $RRT(\mathcal{S}, x_0)$ is finite. By hypothesis, $\mathcal{S}$ is finitely branching and branch-effective. In particular, for all x, $Post_{\mathcal{S}}(x)$ is a finite computable set. As $\leq$ is decidable, we deduce that $RRT(\mathcal{S}, x_0)$ is effectively computable. From Proposition 6, we know that $Post^*_{\mathcal{S}}(x_0)$ is infinite iff there exists a finite branch $n_0(x_0) \xrightarrow{*} n_1(x_1) \xrightarrow{+} n_2(x_2)$ such that $x_1 < x_2$. This last property can be decided on $RRT(\mathcal{S}, x_0)$, and so the boundedness property can be decided, too. □

We also generalize the decidability of termination for WSTS [13] to branch-WSTS.

Proposition 7. *A branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ does not terminate from state x_0 iff there exists a subsumed node in $RRT(\mathcal{S}, x_0)$.*

Theorem 4. *Termination is decidable for branch-effective branch-WSTS.*

Proof. Given a branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$, we apply Proposition 7 so that it is sufficient to build $RRT(\mathcal{S}, x_0)$ and check if there exists a subsumed node. Since $\mathcal{S}$ is branch-effective, we can effectively construct $RRT(\mathcal{S}, x_0)$ and verify the existence of a subsumed node. □

Note that we can thus solve the termination and boundedness problems for the example machine $\mathcal{M}_1$, and since there exists nodes $n_0(x_0)$ and $n_1(x_1)$ in the RRT such that $x_0 = (q_0, \varepsilon)$ and $x_1 = (q_0, a)$ such that $x_0 < x_1$ and $x_0 \xrightarrow{+} x_1$, the machine $\mathcal{M}_1$ is unbounded. Moreover, since $n_1(x_1)$ is also a subsumed node, it is non-terminating.

On the other hand, boundedness becomes undecidable if we relax the strict monotony condition to general monotony (even when we strengthen the order to be wqo). This is because boundedness is undecidable for Reset Petri nets [8]. Reset Petri nets are effective WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$, hence branch-effective WSTS, where $\leq$ is the wqo on vectors of integers. Hence, we deduce:

Proposition 8. *Boundedness is undecidable for branch-effective branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ where $\leq$ is a wqo.*

Counter Machines with Restricted Zero Tests. Now, we show an example of a class that is branch-WSTS. We study counter machines with restricted zero tests. In [4], it was shown that termination and boundedness (and moreover, reachability) are decidable for this class of systems. However, using the alternative approach of branch-WSTS, we can verify that termination and boundedness are decidable for this class without reducing these problems to reachability.

We recall that a *counter machine* (with zero tests) is a tuple $\mathcal{C} = (Q, V, T, q_0)$. Here, Q is the finite set of *control states* and $q_0 \in Q$ is the *initial control state*. Moreover, V is a finite set of *counters* and $T \subseteq Q \times A_{\mathcal{C}} \times Q$ is the transition relation where $A_{\mathcal{C}} = \{\mathsf{inc}(v), \mathsf{dec}(v) \mid v \in V\} \times 2^V$ (an element of 2^V will indicate the set of counters to be tested to 0).

The counter machine $\mathcal{C}$ induces an LTS $\mathcal{S}_{\mathcal{C}} = (X_{\mathcal{C}}, A_{\mathcal{C}}, \rightarrow_{\mathcal{C}}, x_0)$ with set of states $X_{\mathcal{C}} = Q \times \mathbb{N}^V$. In $(q, \ell) \in X_{\mathcal{C}}$, the first component q is the current control state and $\ell = (\ell_v)_{v \in V}$ represents the counter values. The initial state is then $x_0 = (q_0, \ell)$ with all ℓ_v equal to 0.

For $op \in \{\mathsf{inc}, \mathsf{dec}\}$, $v \in V$, and $Z \subseteq V$ (the counters tested for zero), there is a transition $(q, \ell) \xrightarrow{op(v),Z}_{\mathcal{C}} (q', m)$ if $(q, (op(v), Z), q') \in T$, $\ell_{v'} = 0$ for all $v' \in Z$ (applies the zero tests), $m_v = \ell_v + 1$ if $op = \mathsf{inc}$ and $m_v = \ell_v - 1$ if $op = \mathsf{dec}$, and $m_{v'} = \ell_{v'}$ for all $v' \in V \setminus \{v\}$.

We define *counter machines with restricted zero tests (CMRZ)* imposing the following requirement: Once a counter has been tested for zero, it cannot be incremented or decremented anymore. Formally, we require that, for all valid transition sequences $(q_1, \ell_1) \xrightarrow{op(v_1),Z_1}_{\mathcal{C}} (q_2, \ell_2) \xrightarrow{op(v_2),Z_2}_{\mathcal{C}} \dots \xrightarrow{op(v_n),Z_n}_{\mathcal{C}} (q_{n+1}, \ell_{n+1})$ and every two positions $1 \leq i \leq j \leq n$, we have $v_j \notin Z_i$.

Let us consider the wqo $\leq$ on $Q \times \mathbb{N}^V$ where $(q, \ell) \leq (q', m)$ if $q = q'$ and $\ell \leq m$. Note that this ordering is a partial ordering.

Proposition 9. *CMRZs are branch-monotone and strictly branch-monotone for the wqo $\leq$.*

Therefore, since $\leq$ is a wqo:

Theorem 5. *CMRZs are branch-WSTS.*

Furthermore, since $\leq$ and $\rightarrow_{\mathcal{C}}$ are decidable, and $Post_{\mathcal{S}_{\mathcal{C}}}(x)$ is a finite, computable set for all $x \in X_{\mathcal{C}}$, we have:

Proposition 10. *CMRZs are branch-effective.*

Hence, we deduce:

Theorem 6. *Termination and boundedness are decidable for counter machines with restricted zero tests.*

Restrictions on FIFO Machines. Next, we consider FIFO machines.

Definition 10. *A FIFO machine $\mathcal{M}$ with a unique channel over the finite message alphabet A is a tuple $\mathcal{M} = (Q, A, T, q_0)$ where Q is a finite set of control states and $q_0 \in Q$ is an initial control state. Moreover, $T \subseteq Q \times \{!, ?\} \times A \times Q$ is the transition relation, where $\{!\} \times A$ and $\{?\} \times A$ are the set of send and receive actions, respectively.*

The FIFO machine $\mathcal{M}$ induces the LTS $\mathcal{S}_{\mathcal{M}} = (X_{\mathcal{M}}, \Sigma_{\mathcal{M}}, \rightarrow_{\mathcal{M}}, x_0)$. Its set of states is $X_{\mathcal{M}} = Q \times A^*$. In $(q, w) \in X_{\mathcal{M}}$, the first component q denotes the current control state and $w \in A^*$ denotes the contents of the channel. The initial state is $x_0 = (q_0, \varepsilon)$, where ε denotes the empty channel. Moreover, $\Sigma_{\mathcal{M}} = \{!, ?\} \times A$. The transitions are given as follows:

- $(q, w) \xrightarrow{!a}_{\mathcal{M}} (q', w')$ if $(q, !a, q') \in T$ and $w' = w \cdot a$.

– $(q, w) \xrightarrow{?a}_{\mathcal{M}} (q', w')$ if $(q, ?a, q') \in T$ and $w = a \cdot w'$.

The index $\mathcal{M}$ may be omitted whenever $\mathcal{M}$ is clear from the context. When there is no ambiguity, we confuse machines and their associated LTS.

The FIFO machine $\mathcal{M}_1$ from Fig. 1 is an example of a system that is branch-WSTS but the underlying set of states is not well-quasi-ordered. We first try to generalize a class of systems which are branch-wqo, and which includes $\mathcal{M}_1$.

Branch-Wqo FIFO Machines. We consider a restriction that has been studied in [4], which we go on to prove is branch-wqo. These systems are known as input-bounded FIFO machines, which we formally define below. First, we recall the definition of a bounded language.

Let $w_1, \ldots, w_n \in A^+$ be non-empty words where $n \geq 1$. A *bounded language* over $(w_1, \ldots, w_n)$ is a language $L \subseteq w_1^* \ldots w_n^*$. We let $\mathsf{proj}_! : \Sigma_{\mathcal{M}}^* \to A^*$ be the homomorphism defined by $\mathsf{proj}_!(!a) = a$ for all $a \in A$ and $\mathsf{proj}_!(\beta) = \varepsilon$ if β is not of the form $!a$ for some $a \in A$. We define $\mathsf{proj}_?$ the same way. Using these notions, the input language of $\mathcal{M}$ is defined as $L_{\text{input}}(\mathcal{M}) = \{\mathsf{proj}_!(\sigma) \mid x_0 \xrightarrow{\sigma}_{\mathcal{M}} x \text{ for some } x \in X_{\mathcal{M}}\}$. Note that the input language is prefix-closed. Moreover, the prefix language of a bounded language is a bounded language.

Definition 11. *A FIFO machine $\mathcal{M} = (Q, A, T, q_0)$ is* input-bounded *if its input language $L_{\text{input}}(\mathcal{M})$ is bounded.*

Let us recall the extended prefix ordering on the states of a FIFO machine: we let $(q, w) \leq_p (q', w')$ if $q = q'$ and $w \preceq w'$.

Proposition 11. *Input-bounded FIFO machines are branch-wqo for the prefix-ordering $\leq_p$.*

It is clear that $\mathcal{M}_1$ belongs to this class of FIFO systems. But, we see that this subclass is not branch-WSTS (cf. Figure 4). We have $(q_1, \varepsilon) \xrightarrow{\sigma} (q_1, a)$, where $\sigma = !b?b!a$. Moreover, $(q_1, \varepsilon) \leq_p (q_1, a)$. However, there exists no (q_1, w) such that $(q_1, a) \xrightarrow{\sigma} (q_1, w)$, and $(q_1, a) \leq_p (q_1, w)$. In fact, σ cannot be executed from (q_1, a). Therefore, the machine is not branch-monotone under the prefix ordering.

Proposition 12. *Input-bounded FIFO machines are, in general, not branch-monotone for the prefix-ordering.*

However, when we consider *the normal form of input-bounded FIFO machines*, as defined in [4], we conjecture that they are branch-monotone for the prefix-ordering. Furthermore, since they are input-bounded, this would imply that they are branch-WSTS. Moreover, it was also shown in [4], that for every input-bounded FIFO machine, one can construct an equivalent normal form with an exponential blow-up. This would give us another method to verify if a given machine is bounded or has a terminating run, which bypasses checking the reachability of a state.

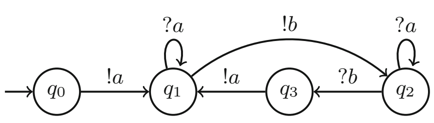

Fig. 4. The FIFO machine $\mathcal{M}_2$

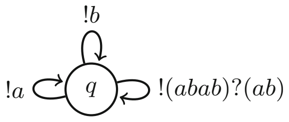

Fig. 5. The FIFO machine $\mathcal{M}_3$.

Branch-Monotone FIFO Machines. We now modify the prefix-ordering further, in order to construct another subclass of FIFO systems which are branch-WSTS. This relation has been previously studied, notably in [12].

Definition 12. *For two states* (q, w) *and* (q', w') *of a FIFO machine* $\mathcal{M}$*, we say that* (q, w) R (q', w') *if* $q = q'$ *and there exists a sequence* $\sigma \in \Sigma_{\mathcal{M}}^*$ *such that* $(q, w) \xrightarrow{\sigma} (q', w')$ *and*

- $\mathsf{proj}_?(\sigma) = \varepsilon$*, or*
- $w \preceq w'$ *and* $(\mathsf{proj}_?(\sigma))^\omega = w.(\mathsf{proj}_!(\sigma))^\omega$*.*

In fact, R is not a qo. It is reflexive, but not transitive:

Example 3. Consider the FIFO machine $\mathcal{M}_3$ in Fig. 5. Consider states $x_1 = (q, a)$, $x_2 = (q, ab)$, and $x_3 = (q, abab)$. We represent the sequence of actions $!a!b!a!b$ by a single transition $!abab$ in the figure, and omit the intermediate control-states for simplicity (and similarly, for $?ab$). It is easy to see that $x_1\ R\ x_2$ and $x_2\ R\ x_3$. When we consider $x_1 \leq_p x_3$, we have $x_1 \xrightarrow{\sigma} x_3$, where $\sigma = !b(!abab)(?ab)$. However, $(ab)^\omega \neq a.(babab)^\omega$, hence, $x_1 \not R\ x_3$, and thus, the relation is not transitive.

Earlier, we defined branch-monotony for transition systems equipped with a quasi-ordering. We now extend the notion for transition systems with a relation.

Proposition 13. *FIFO machines are branch-monotone for the relation* R*.*

Remark 5. This monotony relation is equivalent to the one described in [15], for FIFO systems. However, we have generalized the notion, and included it in the framework. Hence, we can extend this notion to prove the decidability of termination, something which was not shown earlier.

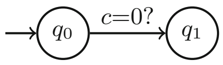

Fig. 6. System $\mathcal{M}_4$ is branch-WSTS.

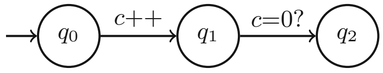

Fig. 7. System $\mathcal{M}_5$ is branch-WSTS.

4 Decidability of Coverability

Coverability Algorithms for Branch-WSTS. We show that the two existing coverability algorithms for WSTS do not allow one to decide coverability for branch-WSTS. Remark that, contrary to WSTS, $Pre^*(\uparrow x)$ is not necessarily upward-closed. In fact, even with a single zero-test, this property is not satisfied.

In Fig. 6, let us consider the counter machine $\mathcal{M}_4$ with a single counter c. Let $x = (q_1, 0)$. We see that $Pre^*(\uparrow x) = \{(q_1, n) \mid n \geq 0\} \cup \{(q_0, 0)\}$. However, $\uparrow Pre^*(\uparrow x) = Pre^*(\uparrow x) \cup \{(q_0, n) \mid n \geq 1\}$. Thus, we get:

Proposition 14. *Given a branch-effective branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ and a state $x \in X$, the set $Pre^*(\uparrow x)$ is not necessarily upward-closed. Hence, we cannot use the backward algorithm.*

Let us consider using the forward algorithm instead. The second procedure computes all sets X which satisfy the property $\downarrow Post^*(X) \subseteq X$. This is because for WSTS, the set $\downarrow Post^*(x)$ satisfies this property. However, we now show a counter-example of a branch-WSTS which does not satisfy this property.

Consider the counter machine $\mathcal{M}_5$ from Fig. 7, with $x_0 = (q_0, 0)$. We compute $\downarrow Post^*(x_0)$. We see that $Post^*(x_0) = \{(q_0, 0), (q_1, 1)\}$, hence, $Y = \downarrow Post^*(x_0) = \{(q_0, 0), (q_1, 1), (q_1, 0)\}$. However, $\downarrow Post^*(Y) \not\subseteq Y$, as $\downarrow Post^*(Y) = \{(q_0, 0), (q_1, 1), (q_1, 0), (q_2, 0)\}$, which is strictly larger than Y. Hence:

Proposition 15. *For branch-effective, branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ such that $\downarrow Post(\downarrow x)$ is computable for all $x \in X$, the set $Y = \downarrow Post^*(x_0)$ does not necessarily satisfy the property $\downarrow Post^*(Y) \subseteq Y$. Hence, the forward coverability algorithm may not terminate.*

We can deduce:

Proposition 16. *For branch-WSTS, both the backward coverability algorithm and the forward coverability algorithm do not terminate, in general.*

Not only the two coverability algorithms do not terminate but we may prove that coverability is undecidable.

Theorem 7. *The coverability problem is undecidable for branch-effective branch-WSTS $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ (even if $\mathcal{S}$ is strongly monotone and $\leq$ is wqo).*

Proof. We use the family of systems given in the proof of Theorem 4.3 [11]. Let us denote by TM_j the j^{th} Turing Machine in some enumeration. Consider the family of functions $f_j : \mathbb{N}^2 \rightarrow \mathbb{N}^2$ defined by $f_j(n, k) = (n, 0)$ if $k = 0$ and TM_j runs for more than n steps, else $f_j(n, k) = (n, n + k)$. Let $g : \mathbb{N}^2 \rightarrow \mathbb{N}^2$ be the function defined by $g(n, k) = (n + 1, k)$. The transition system $\mathcal{S}_j$ induced by the two functions f_j and g is strongly monotone hence it is also branch-monotone. Moreover, system $\mathcal{S}_j$ is branch-effective and we observe that $Post$ is computable and $\leq$ is wqo. Now, we have $(1, 1)$ is coverable from $(0, 0)$ in $\mathcal{S}_j$ iff TM_j halts. This proves that coverability is undecidable. □

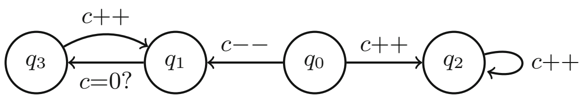

Fig. 8. Machine $\mathcal{M}_6$ is cover-monotone. However, if we modify the system such that the initial state $(q_0, 1)$, then it is not cover-monotone.

Decidability of Coverability. We show that coverability is decidable for a class of systems with a wqo but with a restricted notion of monotony. We define $Cover_{\mathcal{S}}(x) = \downarrow Post^*_{\mathcal{S}}(x)$. Let us consider the following monotony condition.

Definition 13 (cover-monotony). *Let $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ be a system. We say that $\mathcal{S}$ is* cover-monotone *(resp. strongly cover-monotone) if, for all $y_1 \in Cover_{\mathcal{S}}(x_0)$ and for all $x_1, x_2 \in X$ such that $x_1 \leq y_1$ and $x_1 \rightarrow x_2$, there exists a state $y_2 \in X$ such that $y_1 \xrightarrow{*} y_2$ (resp. $y_1 \rightarrow y_2$) and $x_2 \leq y_2$.*

Let us emphasize that cover-monotony of a system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ is a property that depends on the initial state x_0 while the usual monotony does not depend on any initial state (see Fig. 8).

Remark 6. The strong cover-monotony property is not trivially decidable for general models while (usual) strong-monotony is decidable for many powerful models like FIFO machines and counter machines. However, this notion is still of theoretical interest, as it shows that we can relax the general monotony condition.

However, there is a link between general monotony and cover-monotony.

Proposition 17. *A system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq)$ is monotone iff for all $x_0 \in X$, $(X, \Sigma, \rightarrow, \leq, x_0)$ is cover-monotone.*

We may now define cover-WSTS as follows.

Definition 14 (Cover-WSTS). *A* cover-WSTS *is a finitely branching cover-monotone system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ such that $(X, \leq)$ is wqo.*

For cover-WSTS, the backward algorithm fails. This is once again because the presence of a single zero test removes the property of the set being upward-closed. But we will now show that the forward coverability approach is possible.

Proposition 18. *Given a system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ and a downward-closed set $D \subseteq X$ such that $\downarrow Post(D) \subseteq D$, then we have the inclusion $\downarrow Post^*(D) \subseteq D$.*

Let us define a particular instance of the coverability problem in which we verify if a state is coverable from the initial state.

Definition 15. *Given a system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$.* The x_0-coverability problem *is: Given a state $y \in X$, do we have $y \in \downarrow Post^*_{\mathcal{S}}(x_0)$?*

We show that x_0-coverability is decidable for cover-WSTS:

Theorem 8. *Let $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ be an ideally effective cover-WSTS such that Post is computable. Then, the x_0-coverability problem is decidable.*

Proof. Consider a system $\mathcal{S} = (X, \Sigma, \rightarrow, \leq, x_0)$ that is cover-WSTS, and let us consider a state $y \in X$. To find a certificate of coverability (if it exists), we cannot use Procedure 1 since general monotony is not satisfied and then, in general, $\downarrow Post^*(x_0) \neq \downarrow Post^*(\downarrow x_0)$ but we can use a variation of Procedure 1, where we iteratively compute x_0, $Post(x_0)$, $Post(Post(x_0))$, and so on, and at each step check if $y \leq x$ for some x in the computed set. This can be done because $\mathcal{S}$ is finitely branching and the sets $Post^k(x_0)$ are computable for all $k \geq 0$. Hence, if there exists a state that can cover y reachable from x_0, it will eventually be found.

Now, let us prove that Procedure 2 terminates for input y iff y is not coverable from x_0. If Procedure 2 terminates, then at some point, the `while` condition is not satisfied and there exists a set D such that $y \notin D$ and $x_0 \in D$ and $\downarrow Post(D) \subseteq D$. Moreover, $\downarrow Post^*(I) \subseteq I$ for every inductive invariant I (see Proposition 18). Hence, $Cover_{\mathcal{S}}(x_0) \subseteq D$, therefore, since $y \notin D$, we deduce that $y \notin Cover_{\mathcal{S}}(x_0)$ and then y is not coverable from x_0.

Note that every downward-closed subset of X decomposes into finitely many ideals since $(X, \leq)$ is wqo. Moreover, since $\mathcal{S}$ is ideally effective, ideals of X may be effectively enumerated. By [2] and [3], for ideally effective systems, testing of inclusion of downward-closed sets, and checking the membership of a state in a downward-closed set, are both decidable.

To show the opposite direction, let us prove that if y is not coverable from x_0, the procedure terminates. It suffices to prove that $Cover_{\mathcal{S}}(x_0)$ is an inductive invariant. Indeed, this implies that $Cover_{\mathcal{S}}(x_0)$ is eventually computed by Procedure 2 when y is not coverable from x_0.

Let us show $\downarrow Post(Cover_{\mathcal{S}}(x_0)) \subseteq Cover_{\mathcal{S}}(x_0)$. Let $b \in \downarrow Post(Cover_{\mathcal{S}}(x_0))$. Then, there exists a', a', b' such that $x_0 \xrightarrow{*} a'$, $a' \geq a$, $a \rightarrow b'$ and $b' \geq b$. Furthermore, $a', a \in Cover(x_0)$. Hence, by cover-monotony, there exists $b'' \geq b'$ such that $a' \xrightarrow{*} b''$. Therefore, $x_0 \xrightarrow{*} b''$ and $b'' \geq b' \geq b$, hence, $b \in Cover_{\mathcal{S}}(x_0)$. Hence, the x_0-coverability problem is decidable. □

Theorem 9. *The coverability problem is undecidable for cover-WSTS.*

Proof. Given any counter machine $\mathcal{C} = (Q, V, T, q_0)$, let $\mathcal{S}_{\mathcal{C}} = (X, A_{\mathcal{C}}, \rightarrow, \leq, x_0)$ be its transition system equipped with the natural order on counters. We can construct a system $\mathcal{S}' = (X', A_{\mathcal{C}}, \rightarrow', \leq, x_0')$ such that $\mathcal{S}'$ is cover-monotone, and any state $x \in X$ is coverable iff it is also coverable in X'. The construction is as follows. We add a new control state q from the initial state in the counter machine (q_0) reachable via an empty transition, therefore, $X' = X \cup \{(q, 0)\}$. This new control state is a sink state, i.e. there are no transitions from q to any other control state (except itself). Moreover, we let $x_0' = (q, 0)$. Note that $\mathcal{S}'$ is cover-monotone, because there is no state reachable from x_0', hence, the property is vacuously satisfied. However, for all other states, as we leave the system unchanged, we see that a state x is coverable in $\mathcal{S}$ by a state y iff it is coverable in

$\mathcal{S}'$. Hence, coverability for counter machines reduces to the coverability problem for cover-WSTS, and coverability is therefore, undecidable for cover-WSTS. □

5 Conclusion

We have tried to relax the notions of monotony and of the wellness of the quasi-ordering which were traditionally used to define a WSTS. We observed that we do not need the wellness of the quasi-ordering or monotony between all states. By relaxing the conditions to only states reachable from one another, thus defining what we call *branch-WSTS*, we are still able to decide termination and boundedness. Furthermore, some systems that have been studied recently have been shown to belong to this class, which adds interest to this relaxation.

However, as coverability is undecidable for branch-WSTS, the notion of coverability seems to require a stricter condition than what we define for branch-WSTS. This leads us to introduce a different class of systems, incomparable to branch-WSTS, which we call *cover-WSTS*. These systems relax the condition of monotony to only states within the coverability set, while still retaining the decidability of a restricted form of coverability.

As future work, other systems that belong to these classes can be studied. It would also be interesting to see if the branch-WSTS relaxation translates to better hope for usability of WSTS and relaxations as a verification technique.

Acknowledgements. The work reported was carried out in the framework of ReLaX, UMI2000 (ENS Paris-Saclay, CNRS, Univ. Bordeaux, CMI, IMSc). It is partly supported by ANR FREDDA (ANR-17-CE40-0013) and ANR BRAVAS (ANR-17-CE40-0028).

References

1. Abdulla, P.A., Cerans, K., Jonsson, B., Tsay, Y.: Algorithmic analysis of programs with well quasi-ordered domains. Inf. Comput. **160**(1-2), 109–127 (2000). https://doi.org/10.1006/inco.1999.2843
2. Blondin, M., Finkel, A., McKenzie, P.: Handling infinitely branching WSTS. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) ICALP 2014. LNCS, vol. 8573, pp. 13–25. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43951-7_2
3. Blondin, M., Finkel, A., McKenzie, P.: Well behaved transition systems. Log. Methods Comput. Sci. **13**(3) (2017). https://doi.org/10.23638/LMCS-13(3:24)2017
4. Bollig, B., Finkel, A., Suresh, A.: Bounded reachability problems are decidable in FIFO machines. In: Konnov, I., Kovács, L. (eds.) 31st International Conference on Concurrency Theory, CONCUR 2020, September 1–4, 2020, Vienna, Austria (Virtual Conference). LIPIcs, vol. 171, pp. 49:1–49:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPIcs.CONCUR.2020.49
5. Chadha, R., Viswanathan, M.: Deciding branching time properties for asynchronous programs. Theor. Comput. Sci. **410**(42), 4169–4179 (2009). https://doi.org/10.1016/j.tcs.2009.01.021

6. Dickson, L.E.: Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. Am. J. Math. **35**(4), 413–422 (1913)
7. D'Osualdo, E., Stutz, F.: Decidable inductive invariants for verification of cryptographic protocols with unbounded sessions. In: Konnov, I., Kovács, L. (eds.) 31st International Conference on Concurrency Theory, CONCUR 2020, September 1–4, 2020, Vienna, Austria (Virtual Conference). LIPIcs, vol. 171, pp. 31:1–31:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPIcs.CONCUR.2020.31
8. Dufourd, C., Finkel, A., Schnoebelen, P.: Reset nets between decidability and undecidability. In: Larsen, K.G., Skyum, S., Winskel, G. (eds.) ICALP 1998. LNCS, vol. 1443, pp. 103–115. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055044
9. Erdős, P., Rado, R.: A partition calculus in set theory. Bull. Amer. Math. Soc. **62**(5), 427–489 (1956)
10. Finkel, A.: Reduction and covering of infinite reachability trees. Inf. Comput. **89**(2), 144–179 (1990). https://doi.org/10.1016/0890-5401(90)90009-7
11. Finkel, A., McKenzie, P., Picaronny, C.: A well-structured framework for analysing petri net extensions. Inf. Comput. **195**(1-2), 1–29 (2004). https://doi.org/10.1016/j.ic.2004.01.005, http://www.lsv.ens-cachan.fr/Publis/PAPERS/PS/FMP-wstsPN-icomp.ps
12. Finkel, A., Praveen, M.: Verification of flat fifo systems. In: Fokkink, W., van Glabbeek, R. (eds.) Proceedings of the 30th International Conference on Concurrency Theory (CONCUR 2019), Leibniz International Proceedings in Informatics, Leibniz-Zentrum für Informatik, Amsterdam, The Netherlands, August 2019
13. Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere! Theor. Comput. Sci. **256**(1–2), 63–92 (2001). https://doi.org/10.1016/S0304-3975(00)00102-X
14. Higman, G.: Ordering by divisibility in abstract algebras. In: Proceedings of The London Mathematical Society, pp. 326–336 (1952)
15. Jéron, T., Jard, C.: Testing for unboundedness of fifo channels. Theor. Comput. Sci. **113**, 93–117 (1993)