



Decidability of Resilience for Well-Structured Graph Transformation Systems

Okan Özkan (✉) 

University of Oldenburg, Oldenburg, Germany
o.oezkan@informatik.uni-oldenburg.de

Abstract. Resilience is a concept of rising interest in computer science and software engineering. For systems in which correctness w.r.t. a safety condition is unachievable, fast recovery is demanded. We ask whether we can reach a safe state in a bounded number of steps whenever we reach a bad state. In a well-structured framework, we investigate problems of this kind where the bad and safety conditions are given as upward/downward-closed sets. We obtain decidability results for graph transformation systems by applying our results for subclasses of well-structured transition systems. Moreover, we identify sufficient criteria of graph transformation systems for the applicability of our results.

Keywords: Resilience · Graph transformation systems · Decidability · Well-structured transition systems

1 Introduction

Resilience is a broadly used concept in computer science and software engineering (e.g., [11]). In general engineering systems, *fast recovery* from a degraded state is often termed as resilience, see, e.g., [17]. In view of the latter interpretation of resilience, we investigate on the question whether a SAFE state can be reached in a bounded number of steps from any BAD state (where BAD is not necessarily the complement of SAFE). This concept is meaningful for systems in which violation of SAFE cannot be avoided. Our notion of resilience generalizes correctness (e.g., [2, 10, 15]) w.r.t. a safety condition.

For modeling systems, we use *graph transformation systems (GTSs)* in the *single pushout approach (SPO)*, as considered, e.g., in [7], which provide visual interpretability but yet also a precise formalism. In this perception, system states are captured by graphs and state changes by graph transformations. Our goal is to obtain decidability results for GTSs by considering their induced *transition systems*. A transition system consists of a set of states of any kind (not necessarily graphs) and a transition relation on the state set.

Usually, the state set (set of graphs) is infinite. To handle infinite state sets, we employ the concept of well-structuredness studied, e.g., in [1, 9]. A *well-structured transition system (WSTS)* is, informally, a transition system equipped

with a *well-quasi-order* satisfying that larger states simulate smaller states (also called *compatibility* condition) and that certain predecessor sets can be effectively computed. In this well-structured setting, *ideal-based* sets (*upward-* or *downward-closed sets*) play an important role. They enjoy a number of suitable properties for verification such as finite representation (of upward-closed sets) and closure properties. For WSTSs, the *ideal reachability (coverability) problem* is decidable [1,9], which is an ingredient of our results.

Well-structuredness of GTSs is investigated, e.g., in [12] for several well-quasi-orders. The well-quasi-order we use is the subgraph order which permits strong compatibility but comes with the restriction of path-length-boundedness on the graph class.

We show decidability for subclasses of GTSs of bounded path length. Each subclass exhibits additional requirements, i.e., effectiveness or unreliability properties. Additionally, we identify sufficient criteria of GTSs for the applicability of the results.

More precisely, we consider the *explicit* resilience problem where the bound on the number of steps for recovery is given and the *bounded* resilience problem which asks whether there exists such a bound. These problems are formulated for marked GTSs each of which consists of a GTS together with a graph class closed under rule application and an INITIAL subset of graphs. We ask: Starting from any graph in INIT, whenever we reach a BAD graph, can we reach a SAFE graph in $\leq k$ (in a bounded number of) steps?

To illustrate the idea of our resilience concept, we give an example.

Example (circular process protocol). Consider a ring of three processes P_0, P_1, P_2 each of which has an unordered collection (multiset) containing commands. Each command belongs to a process and is labeled accordingly as c_0, c_1 , or c_2 . The protocol is described below. A formalization as GTS can be found in Sect. 4.

- The process P_0 *liberal*, i.e., it can *initiate* (generate) a command c_0 in the collection of the next process.
- Every process P_i can *forward* a command c_j , $i \neq j$, not belonging to itself.
- If a process P_i receives a command c_i , it is *enabled* and can
 1. *execute* its specific process action, or
 2. *clear* all commands in its collection and forward a command of the next process, or
 3. *leave* the process ring (if $i \neq 0$) and forward a command of the next process.

Afterwards, the command c_i is deleted.

- Any command may get *lost* in any state.

The process action of P_0 is to forward two commands, c_1 and c_2 . The process action of P_1 (P_2) is to forward a command c_2 (c_1). The topology of the process ring changes when a process leaves the ring. Processes P_1 and P_2 can leave the ring only if the other process has not left the ring before. In Fig. 1, the initial state where every process P_i has one command c_i in its collection and the three

possible topologies are shown. A process P_i is represented by an edge labeled with P_i . The collections are represented by white nodes which may have loops labeled with c_i corresponding to the contained commands.

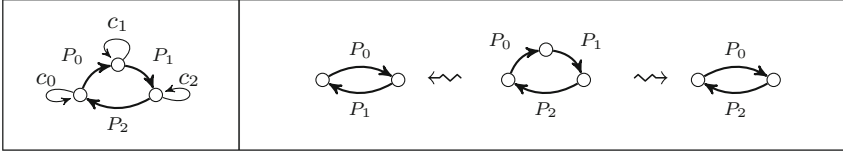


Fig. 1. Initial state and topologies of the circular process protocol.

Consider the following instances of the bounded resilience problem with the initial state as in Fig. 1:

BAD	\neg AllEnabled	Command(c_2)	AllEnabled	NoCommand
SAFE	AllEnabled	Collection(c_0, c_1)	\neg AllEnabled	No3Processes

For every instance of the bounded resilience problem, we are interested in a bound k for the number of steps needed for recovery. In the first instance, we ask whether we can reach a state where every process is enabled in $\leq k$ steps whenever we reach a state where this is not the case. In the second instance, we ask whether we can reach a state with a collection containing commands c_0, c_1 in $\leq k$ steps whenever we reach a state with a collection containing c_2 . The third instance is the “dual” problem to the first one where the constraints for BAD and SAFE are exchanged. In the fourth instance, we ask whether we can reach a state containing no three processes in $\leq k$ steps whenever we reach a state without commands. One may ask:

- Does such a k exist? If so, what is the minimal k ?
- Is there a generic method for problems of this kind?

We will answer these questions in Sect. 3 and 4.

This paper is organized as follows: In Sect. 2, we recall preliminary concepts of GTSs and (WS)TSs. We show decidability of the resilience problems for subclasses of marked WSTs in Sect. 3. In Sect. 4, we apply our results to marked GTSs. In Sect. 5, we give sufficient, rule-specific criteria for the applicability of our results. We present related concepts in Sect. 6 and close with a conclusion in Sect. 7. The proofs in full length and a further example can be found in a technical report [13].

2 Preliminaries

We recall the concepts used in this paper, namely graph transformation systems [6, 7] and (in particular well-structured) transition systems [9].

2.1 Graph Transformation Systems

In the following, we recall the definitions of graphs, graph constraints, rules, and graph transformation systems [6, 7].

A directed, labeled graph consists of a finite set of nodes and a finite set of edges where each edge is equipped with a source and a target node and where each node and edge is equipped with a label. Note that this kind of graphs are a special case of the hypergraphs considered in [12].

Definition 1 (graphs & morphisms). A *(directed, labeled) graph* (over a finite label alphabet $\Lambda = \Lambda_V \cup \Lambda_E$) is a tuple $G = \langle V_G, E_G, src_G, tgt_G, lab_G^V, lab_G^E \rangle$ with finite sets V_G and E_G of *nodes* (or *vertices*) and *edges*, functions $src_G, tgt_G : E_G \rightarrow V_G$ assigning *source* and *target* to each edge, and *labeling functions* $lab_G^V : V_G \rightarrow \Lambda_V$, $lab_G^E : E_G \rightarrow \Lambda_E$. A *(simple, undirected) path* in G of length ℓ is a sequence $\langle v_1, e_1, v_2, \dots, v_\ell, e_\ell, v_{\ell+1} \rangle$ of nodes and edges s.t. $src_G(e_i) = v_i$ and $tgt_G(e_i) = v_{i+1}$, or $tgt_G(e_i) = v_i$ and $src_G(e_i) = v_{i+1}$ for every $1 \leq i \leq \ell$, and all contained nodes and edges occur at most once. Given graphs G and H , a *(partial graph) morphism* $g : G \rightarrow H$ consists of partial functions $g_V : V_G \rightarrow V_H$ and $g_E : E_G \rightarrow E_H$ which preserve sources, targets, and labels, i.e., $g_V \circ src_G(e) = src_H \circ g_E(e)$, $g_V \circ tgt_G(e) = tgt_H \circ g_E(e)$, $lab_G^V(v) = lab_H^V \circ g_V(v)$, and $lab_G^E(e) = lab_H^E \circ g_E(e)$ on all nodes v and edges e , for which $g_V(v), g_E(e)$ is defined. Furthermore, if a morphism is defined on an edge, it must be defined on both incident nodes. The morphism g is *total (injective)* if both g_V and g_E are total (injective). If g is total and injective, we also write $g : G \hookrightarrow H$. The composition of morphisms is defined componentwise. A pair $\langle G \rightarrow C, G' \rightarrow C \rangle$ of morphisms is *jointly surjective* if every item of C has a preimage in G or G' .

Convention. We draw graphs as usual. Labels are indicated by a symbol or a color. In (partial) morphisms, we equip the image of a node with the same index. Nodes on which the morphism is undefined have no index.

We consider a special case of graph constraints [10, 16], which are non-nested and based on positive ($\exists G$)/negative ($\neg \exists G$) constraints. For simplicity, we call them also positive (negative) constraints.

Definition 2 (positive & negative constraints). The class of *positive (negative) (graph) constraints* is the smallest class of expressions which contains $\exists G$ (negative: $\neg \exists G$) for every graph G and is closed under \vee and \wedge . A graph G *satisfies* $\exists G'$ if there exists a total, injective morphism $G' \hookrightarrow G$. The semantics of the logical operators are as usual. We write $G \models c$ if G satisfies the positive/negative constraint c . For a positive/negative constraint c , we denote by $\llbracket c \rrbracket$ the set of all graphs G of the considered graph class with $G \models c$.

Using jointly surjective morphisms, every positive constraint can algorithmically be converted into an equivalent “ \vee -normal form”.

Fact 1 (from \wedge to \vee). For every positive constraint c , we can effectively construct a positive constraint c' of the form $\bigvee_{1 \leq i \leq n} \exists G_i$ s.t. $\llbracket c \rrbracket = \llbracket c' \rrbracket$ and there exists no total, injective morphism $G_i \hookrightarrow G_j$ for $i \neq j$.

We use the single pushout (SPO) approach [7] with injective matches for modeling graph transformations.

Definition 3 (graph transformation). A (graph transformation) rule $r = \langle L \rightarrow R \rangle$ is a partial morphism from a graph L to a graph R . A graph transformation system (GTS) is a finite set of rules. A transformation $G \Rightarrow H$ from a graph G to a graph H applying a rule r at a total, injective match morphism $g : L \hookrightarrow G$ is given by a pushout as shown in Fig. 2 (1) (for existence and construction of pushouts, see, e.g., [7]). We write $G \Rightarrow_r H$ to indicate the applied rule, and $G \Rightarrow_{\mathcal{R}} H$ if $G \Rightarrow_r H$ for a rule r in the rule set \mathcal{R} .

In Fig. 2 (2), an example for a transformation is shown.

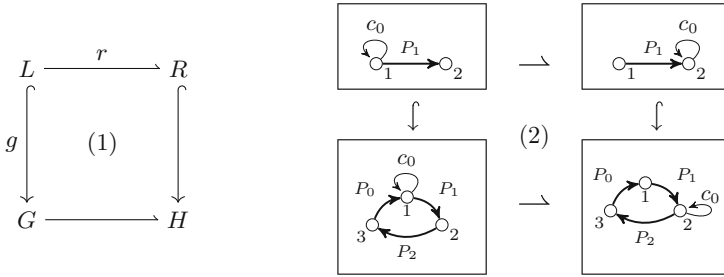


Fig. 2. Pushout scheme and example of a transformation.

2.2 Transition Systems and Well-structuredness

We recall the notion of transition systems.

Definition 4 (transition system). A transition system (TS) $\langle S, \rightarrow \rangle$ consists of a (possibly infinite) set S of states and a transition relation $\rightarrow \subseteq S \times S$. Let $\rightarrow^0 = \text{Id}_S$ (identity on S), $\rightarrow^1 = \rightarrow$, and $\rightarrow^k = \rightarrow^{k-1} \circ \rightarrow$ for every $k \geq 2$. Let $\rightarrow^{\leq k} = \bigcup_{0 \leq j \leq k} \rightarrow^j$ for every $k \geq 0$. The transitive closure is given by $\rightarrow^* = \bigcup_{k \geq 0} \rightarrow^k$.

Often we are interested in the predecessors or successors of state set.

Definition 5 (pre- & postsets). Let $\langle S, \rightarrow \rangle$ be a transition system. For $S' \subseteq S$ and $k \geq 0$, we define $\text{pre}^k(S') = \{s \in S \mid \exists s' \in S' : s \rightarrow^k s'\}$ and $\text{post}^k(S') = \{s \in S \mid \exists s' \in S' : s' \rightarrow^k s\}$. Let $\text{pre}^{\leq k}(S') = \bigcup_{j \leq k} \text{pre}^j(S')$, $\text{pre}^*(S') = \bigcup_{k \geq 0} \text{pre}^k(S')$, $\text{post}^{\leq k}(S') = \bigcup_{j \leq k} \text{post}^j(S')$, and $\text{post}^*(S') = \bigcup_{k \geq 0} \text{post}^k(S')$. We abbreviate $\text{post}^1(S')$ by $\text{post}(S')$ and $\text{pre}^1(S')$ by $\text{pre}(S)$. A TS $\langle S, \rightarrow \rangle$ is finite-branching if $\text{post}(s)$ is finite and computable for every given state s .

Several problems are undecidable for infinite-state TSs in general. However, interesting decidability results can be achieved if the system is well-structured [1, 9, 12]. A prerequisite for this concept is a well-quasi-order on the state set.

Definition 6 (well-quasi-order). A *quasi-order* is a reflexive, transitive relation. A *well-quasi-order (wqo)* over a set X is a quasi-order $\leq \subseteq X \times X$ s.t. every infinite sequence $\langle x_0, x_1, \dots \rangle$ in X contains an increasing pair $x_i \leq x_j$ with $i < j$. A (well-)quasi-order is *decidable* if it can be decided whether $x \leq x'$ for all $x, x' \in X$.

In our setting, the subgraph order is of crucial importance.

Example 1 (subgraph order). The subgraph order \leq is given by $G \leq H$ iff there is a total, injective morphism $G \hookrightarrow H$. Let S_ℓ be a graph class of bounded path length (with bound ℓ). The restriction of \leq to S_ℓ is a wqo [4, 12]. However, it is not a wqo on all graphs: The infinite sequence $\langle \text{C}_3, \text{C}_4, \text{C}_5, \dots \rangle$ of cyclic graphs of increasing length contains no increasing pair.

Assumption. From now on, we implicitly equip every set of graphs with the subgraph order. By “ \leq ” we mean either an abstract wqo or the subgraph order.

Upward- and downward-closed sets are of special interest.

Definition 7 (ideal & basis). Let X be a set and \leq a quasi-order on X . For every subset A of X , we denote by $\uparrow A = \{x \in X \mid \exists a \in A : a \leq x\}$ the *upward-closure* and $\downarrow A = \{x \in X \mid \exists a \in A : x \leq a\}$ the *downward-closure* of A . An *ideal* $I \subseteq X$ is an upward-closed set, i.e., $\uparrow I = I$. An *anti-ideal* $J \subseteq X$ is a downward-closed set, i.e., $\downarrow J = J$. An (anti-)ideal is *decidable* if membership for every $x \in X$ is decidable. A *basis* of an ideal I is a subset $B \subseteq I$ s.t. (i) $\uparrow B = I$ and (ii) $b \neq b' \Rightarrow b \not\leq b'$ for all $b, b' \in B$.

Fact 2 (ideals of graphs). For every positive (negative) constraint c , $\llbracket c \rrbracket$ is an (anti-)ideal.

Ideals are, in general, infinite but can be represented by finite bases (a minimal generating set), similar to algebraic structures.

Fact 3 (finite basis [1, Lemma 3.3]). Every ideal has a basis and every basis is finite, provided that the superset is equipped with a wqo. Given a finite set A , a basis of $\uparrow A$ is computable, provided that the quasi-order is decidable.

Anti-ideals are the complements of ideals. Since an anti-ideal does not have an “upward-basis” in general, we will later demand that membership is decidable.

For well-structuredness, we demand that the wqo yields a simulation of smaller states by larger states. This condition is called compatibility.

Definition 8 (well-structured transition systems). Let $\langle S, \rightarrow \rangle$ be transition system and \leq a decidable wqo on S . The tuple $\langle S, \leq, \rightarrow \rangle$ is a *well-structured transition system (WSTS)*, if:

- (i) The wqo is *compatible* with the transition relation, i.e., for all $s_1, s'_1, s_2 \in S$ with $s_1 \leq s'_1$ and $s_1 \rightarrow s_2$, there exists $s'_2 \in S$ with $s_2 \leq s'_2$ and $s'_1 \rightarrow^* s'_2$. If $s'_1 \rightarrow^1 s'_2$, we say that it is *strongly compatible*. Both is illustrated in Fig. 3.
- (ii) For every $s \in S$, a basis of $\uparrow \text{pre}(\uparrow \{s\})$ is computable.

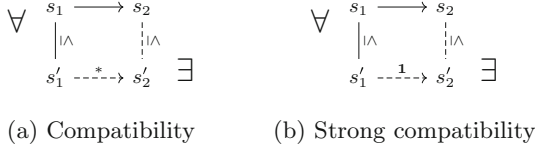


Fig. 3. Visualization of (strong) compatibility.

A *strongly WSTS* (*SWSTS*) is a WSTS with strong compatibility.

Remark. For GTSSs, strong compatibility is achieved by applying the same rule to the bigger graph. In contrast to the double pushout (DPO) approach [6], SPO has the suitable property that every rule is applicable to the bigger graph.

Assumption. Let $\langle S, \leq, \rightarrow \rangle$ be a well-structured transition system.

The set of ideals of S is closed under preset, union, and intersection.

Fact 4 (stability of ideals [1, Lemma 3.2]). For ideals $I, I' \subseteq S$, the sets $\text{pre}^*(I)$, $I \cup I'$, and $I \cap I'$ are ideals. For SWSTSs, the sets $\text{pre}(I)$, $\text{pre}^{\leq k}(I)$ for every $k \geq 0$ are ideals.

An important point in our argumentation is the observation that every infinite, ascending sequence of ideals w.r.t. a wqo eventually becomes stationary.

Lemma 1 (Noetherian state set [1, Lemma 3.4]). For every infinite, ascending sequence $\langle I_0 \subseteq I_1 \subseteq \dots \rangle$ of ideals, $\exists k_0 \geq 0$ s.t. $I_k = I_{k_0}$ for all $k \geq k_0$.

Abdulla et al. [1] exploit Lemma 1 to show the decidability of ideal reachability (coverability) for SWSTSs. The idea is to iteratively construct the sequence of the ideals $I^k = \text{pre}^{\leq k}(I)$ until it becomes stable. This construction is carried out by representing ideals by bases. This argumentation is similarly feasible for WSTSs, see, e.g., [9, proof of Thm. 3.6].

Lemma 2 (ideal reachability [1, Thm. 4.1]). Given a basis of an ideal $I \subseteq S$ and a state s of a SWSTS, we can decide whether we can reach a state $s_I \in I$ from s . In particular, $\text{pre}^{\leq k}(I) = \text{pre}^*(I) \iff \text{pre}^{\leq k+1}(I) = \text{pre}^{\leq k}(I)$, and a basis of $\text{pre}^*(I)$ is computable.

3 Decidability

We show the decidability of resilience problems for subclasses of SWSTSs by extending the idea in [14] to a systematic investigation.

In our setting, ideal-based sets of states play an important role.

Definition 9 (ideal-based). A set is *ideal-based* if it is (i) an ideal with a given basis, or (ii) a decidable anti-ideal. We denote by

- (i) \mathcal{I} the set of ideals with given bases, (ii) \mathcal{J} the set of decidable anti-ideals.

We formulate resilience problems for marked WSTSs, i.e., WSTSs with a specified set INIT of initial states starting from which we investigate resilience.

Definition 10 (marked WSTS). A *marked WSTS* is a tuple $\langle S, \leq, \rightarrow, \text{INIT} \rangle$ where $\langle S, \leq, \rightarrow \rangle$ is a WSTS and $\text{INIT} \subseteq S$. If INIT is finite, we call it *fin-marked*.

EXPLICIT RESILIENCE PROBLEM FOR WSTSs

Given: A marked WSTS $\langle S, \leq, \rightarrow, \text{INIT} \rangle$, ideal-based sets $\text{SAFE}, \text{BAD} \subseteq S$, a natural number $k \geq 0$.

Question: $\forall s \in \text{INIT} : \forall (s \rightarrow^* s' \in \text{BAD}) : \exists (s' \rightarrow^{\leq k} s'' \in \text{SAFE}) ?$

BOUNDED RESILIENCE PROBLEM FOR WSTSs

Given: A marked WSTS $\langle S, \leq, \rightarrow, \text{INIT} \rangle$, ideal-based sets $\text{SAFE}, \text{BAD} \subseteq S$.

Question: $\exists k \geq 0 : \forall s \in \text{INIT} : \forall (s \rightarrow^* s' \in \text{BAD}) : \exists (s' \rightarrow^{\leq k} s'' \in \text{SAFE}) ?$

For our further considerations, we regard requirements in order to obtain decidability, i.e., we consider the following subclasses of marked WSTSs.

Definition 11 (requirements). A marked WSTS $\langle S, \leq, \rightarrow, \text{INIT} \rangle$ is

- (1) *post**-effective if INIT is finite and a basis of $\uparrow \text{post}^*(\text{INIT})$ is computable,
- (2) *lossy* if $\downarrow \text{post}^*(\text{INIT}) = \text{post}^*(\text{INIT})$,
- (3) \perp -*bounded* (*bottom-bounded*) if there exists $\ell \geq 0$ s.t. $s_B \in \text{post}^{\leq \ell}(s)$ for every $s \in S$ and every element s_B of a basis of S with $s \geq s_B$.

The requirement of *post**-effectiveness describes the computability of the smallest reachable states from the initial states. The notion of lossiness means that the set of reachable states from the initial states is downward-closed. This is an abstraction from the lossiness concept in [9, p. 83]. Usually, the term “lossy” describes the circumstance that (almost) any piece of information of a state may get lost. Another kind of unreliability is \perp -boundedness which means that from every state, every smaller basis element (the bottom underneath) is reachable in a bounded number of steps. Thereby (almost) all information of a state may get lost in a bounded number of steps.

The following lemma is crucial for many following proofs.

Lemma 3 (ideal-inclusion [14, Lemma 4]). Let A be a set, I an ideal, and J an anti-ideal. Then, $A \cap J \subseteq I \iff \uparrow A \cap J \subseteq I$.

Applying this lemma to a basis B_I of an ideal I , we obtain that the inclusion $I \cap J \subseteq I'$ in an ideal I' can be checked by computing $B_I \cap J$ and then checking whether $B_I \cap J \subseteq I'$.

We give a characterization of *post**-effectiveness via “anti-ideal reachability”.

Proposition 1 (characterization of post^* -effectiveness). For a class of finite-branching WSTSs, a basis of $\uparrow \text{post}^*(s)$ is computable for every given state s iff the anti-ideal reachability problem is decidable, i.e., given a state s of a WSTS in the regarded class and a decidable anti-ideal J , it can be decided whether $\exists s' \in J : s \rightarrow^* s'$.

Proof (sketch) On one hand, we can decide the anti-ideal reachability problem by computing a basis of $\uparrow \text{post}^*(s)$ and checking whether the intersection with the anti-ideal is empty (Lemma 3). On the other hand, we can compute a basis of $\uparrow \text{post}^*(s)$ by computing the sequence of ideals $P_k = \uparrow \text{post}^{\leq k}(s)$ until it becomes stationary (Lemma 1). The stop condition, i.e., the condition which guarantees that we can terminate the algorithm, is formalized as anti-ideal reachability. \square

The characterization in Proposition 1 is used to show that Petri nets are post^* -effective. It is well-known that Petri nets constitute SWSTSs [9, Thm. 6.1].

Example 2 (variations of Petri nets). (1) Petri nets (equipped with any finite set of initial states) are post^* -effective by Proposition 1: Reachability for Petri nets is decidable and recursively equivalent to submarking reachability [8, p.6]. This corresponds to the anti-ideal reachability problem for Petri nets. (2) *Lossy Petri nets* are Petri nets where in any state, one token may get lost at any place. Lossy Petri nets are lossy for every set of initial states. (3) *Reset-lossy (mixed-lossy) Petri nets* are reset Petri nets [5] where in any state, all tokens (or one token) may get lost at any place. Reset-lossy (mixed-lossy) Petri nets are \perp -bounded (and lossy for every set of initial states).

For some results, we assume that a basis of the set of all states is given. This is only relevant if we use these basis elements for computations.

Notation. For a WSTS $\langle S, \leq, \rightarrow \rangle$ with a given basis of S , we write WSTS^B .

The next proposition shows how the requirements are related provided that a basis of the set of all states is given. The Venn diagram in Fig. 4 illustrates the relations of the subclasses corresponding to the requirements.

Proposition 2. Lossy (\perp -bounded) fin-marked WSTS^B s are post^* -effective.

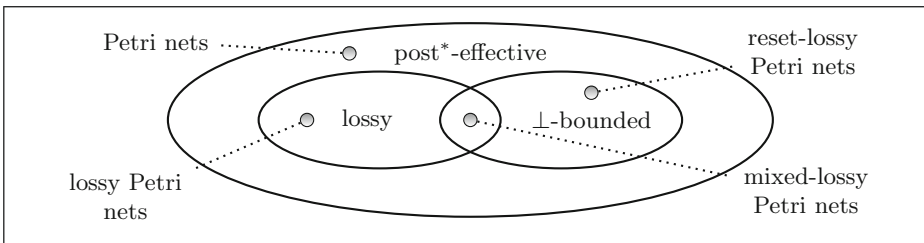


Fig. 4. Subclasses of fin-marked WSTS^B s.

Proof (sketch). Let $\langle S, \leq, \rightarrow, \text{INIT} \rangle$ be a lossy (\perp -bounded) fin-marked WSTS^B . To compute a basis of $\uparrow \text{post}^*(\text{INIT})$ for a finite set INIT , we look at the reachable elements of a basis of the set S of all states. Such a basis element is reachable iff its upward-closure is reachable. By Lemma 2, the latter is decidable. \square

Our main result for fin-marked SWSTSs terms sufficient criteria under which the resilience problems are decidable.

Theorem 1 (decidability for fin-marked SWSTSs). Both resilience problems are decidable for fin-marked SWSTSs which are

- (1) post^* -effective if $\text{BAD} \in \mathcal{J}$, $\text{SAFE} \in \mathcal{I}$ (corresp. [14, Thm. 1]),
- (2) lossy if $\text{BAD}, \text{SAFE} \in \mathcal{I}$.

The bounded resilience problem is decidable for fin-marked SWSTS^B s which are

- (3) lossy and \perp -bounded if $\text{BAD} \in \mathcal{I}$, $\text{SAFE} \in \mathcal{J}$,
- (4) \perp -bounded if $\text{BAD}, \text{SAFE} \in \mathcal{J}$.

Key Idea of the Proof. We compute a finite representation of $\text{post}^*(\text{INIT}) \cap \text{BAD}$ for checking inclusion in a decidable ideal I which is a predecessor set of SAFE .

The proof structure is shown in Fig. 5: Lemma 4 states that for post^* -effective (lossy) fin-marked SWSTSs, a finite representation of $\text{post}^*(\text{INIT}) \cap \text{BAD}$ is computable, i.e., inclusion in a decidable ideal is decidable. In the case $\text{SAFE} \in \mathcal{I}$, the set $\text{pre}^{\leq k}(\text{SAFE})$ is a decidable ideal for every $k \geq 0$. (Lemma 5 shows the existence of bounds for the set of all predecessors of $\text{SAFE} \in \mathcal{J}$ provided that the SWSTS is \perp -bounded.) Proposition 3 shows that $\text{pre}^*(\text{SAFE})$ constitutes a decidable ideal in the case $\text{SAFE} \in \mathcal{J}$ if the SWSTS^B is \perp -bounded.

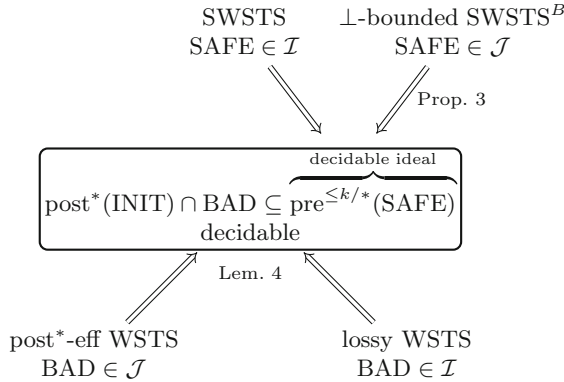


Fig. 5. Structure of the decidability proof for fin-marked SWSTSs.

The following lemma states that the inclusion of $\text{post}^*(\text{INIT}) \cap \text{BAD}$ in an decidable ideal is decidable if we consider post^* -effective in the case $\text{BAD} \in \mathcal{J}$ or lossy fin-marked WSTSs in the case $\text{BAD} \in \mathcal{I}$.

Lemma 4 (checking inclusion). Let $\langle S, \leq, \rightarrow, \text{INIT} \rangle$ be a fin-marked WSTS, $\text{BAD} \subseteq S$, and $I \subseteq S$ be a decidable ideal. Then, it is decidable whether $\text{post}^*(\text{INIT}) \cap \text{BAD} \subseteq I$ provided that the fin-marked WSTS is (a) post^* -effective and $\text{BAD} \in \mathcal{J}$, (b) lossy and $\text{BAD} \in \mathcal{I}$.

Proof (sketch). We compute a finite representation of $\text{post}^*(\text{INIT}) \cap \text{BAD}$ for checking inclusion in the decidable ideal I . To this aim, we use Lemma 3. In case (a), the finite representation is $B_{\text{post}} \cap \text{BAD}$ where B_{post} is a basis of $\uparrow \text{post}^*(\text{INIT})$. In case (b), the finite representation is $\downarrow \text{post}^*(\text{INIT}) \cap B_{\text{BAD}}$ where B_{BAD} is a basis of BAD . \square

By the next lemma, \perp -boundedness implies that for any anti-ideal J , $\text{pre}^*(J)$ is an ideal and $\text{pre}^*(J) = \text{pre}^{\leq k}(J)$ for a $k \geq 0$.

Lemma 5 (existence of bounds). For every \perp -bounded SWSTS and every anti-ideal J , there exists a $k \geq 0$ s.t. $\text{pre}^*(J) = \uparrow \text{pre}^*(J) = \text{pre}^{\leq k}(J)$.

Proof (sketch). By Lemma 1, for every set A of states, there exists a $k_0 \geq 0$ s.t. $\uparrow \text{pre}^*(A) = \uparrow \text{pre}^{\leq k_0}(A)$. By strong compatibility and \perp -boundedness, there exists a constant $\ell \geq 0$ s.t. $\uparrow \text{pre}^{\leq k}(J) \subseteq \text{pre}^{\leq k+\ell}(J)$ for every anti-ideal J . Hence, $\text{pre}^*(J) \subseteq \uparrow \text{pre}^*(J) = \uparrow \text{pre}^{\leq k_0}(J) \subseteq \text{pre}^{\leq k_0+\ell}(J) \subseteq \text{pre}^*(J)$. \square

The following proposition identifies sufficient prerequisites s.t. $\text{pre}^*(\text{SAFE})$ constitutes a decidable ideal in the case $\text{SAFE} \in \mathcal{J}$.

Proposition 3 (decidable ideals). For every \perp -bounded SWSTS^B and every decidable anti-ideal J , the set $\text{pre}^*(J)$ is a decidable ideal.

Proof (sketch). By Lemma 5, $\uparrow \text{pre}^*(J) = \text{pre}^*(J)$. Thus, it is an ideal. By Lemma 3, $s \notin \text{pre}^*(J) \iff B_{\text{post}}(s) \cap J = \emptyset$ where $B_{\text{post}}(s)$ is a basis of $\uparrow \text{post}^*(s)$. By Proposition 2, \perp -boundedness implies post^* -effectiveness w.r.t. any finite set of initial states, provided that a basis of S is given. Hence, membership is decidable. \square

We compile our preparatory results to prove Theorem 1.

Proof (of Theorem 1). Cases (1) & (2). By Fact 4, $\text{pre}^{\leq k}(\text{SAFE})$ is an ideal for every $k \geq 0$ since $\text{SAFE} \in \mathcal{I}$. For every $k \geq 0$, $\text{pre}^{\leq k+1}(\text{SAFE}) = \text{pre}(\text{pre}^{\leq k}(\text{SAFE})) \cup \text{SAFE}$. By Definition 8 and Fact 3, a basis of $\text{pre}^{\leq k}(\text{SAFE})$ is iteratively computable. By Lemma 4, we can decide whether $\text{post}^*(\text{INIT}) \cap \text{BAD} \subseteq \text{pre}^{\leq k}(\text{SAFE})$ for (1) post^* -effective fin-marked SWSTSs and (2) lossy fin-marked SWSTSs, respectively. By Lemma 1, the infinite ascending sequence $\text{SAFE} \subseteq \text{pre}^{\leq 1}(\text{SAFE}) \subseteq \text{pre}^{\leq 2}(\text{SAFE}) \subseteq \dots$ becomes stationary, i.e., there is a minimal $k_0 \geq 0$ s.t. $\text{pre}^{\leq k_0}(\text{SAFE}) = \text{pre}^*(\text{SAFE})$. By Lemma 2, we can also determine this k_0 . Thus, we can determine the minimal number $k = k_{\min}$ s.t. $\text{post}^*(\text{INIT}) \cap \text{BAD} \subseteq \text{pre}^{\leq k}(\text{SAFE})$ (if it exists) and also whether it exists. Hence, we can decide the bounded resilience problem and given any k , we can check whether $k_{\min} \leq k$ to decide the explicit resilience problem.

Cases (3) & (4). By Lemma 5, for \perp -bounded SWSTSs, there exists a $k \geq 0$ s.t. $\text{pre}^*(\text{SAFE}) = \text{pre}^{\leq k}(\text{SAFE})$. Hence, checking bounded resilience is equivalent to testing inclusion in $\text{pre}^*(\text{SAFE})$. By Proposition 3, for \perp -bounded SWSTS^B s, $\text{pre}^*(\text{SAFE})$ is a decidable ideal since $\text{SAFE} \in \mathcal{J}$. By Lemma 4, we obtain that checking $\text{post}^*(\text{INIT}) \cap \text{BAD} \subseteq \text{pre}^*(\text{SAFE})$ is decidable for (3) lossy, \perp -bounded fin-marked SWSTS^B s and (4) post^* -effective, \perp -bounded fin-marked SWSTS^B s, respectively. By Proposition 2, \perp -boundedness implies post^* -effectiveness provided that a basis of the set of all states is given. \square

4 Application to Graph Transformation Systems

We translate the results for WSTSs into the GTS setting.

The sets of positive and negative constraints are subsumed as ideal-based constraints.

Definition 12 (ideal-based constraints). We denote the set of positive (negative) constraints by \mathcal{I}_c (\mathcal{J}_c). An *ideal-based constraint* is an element of $\mathcal{I}_c \cup \mathcal{J}_c$.

Recall that we consider the subgraph order as wqo. Path-length-boundedness on the graph class guarantees that the subgraph order yields a wqo.

Similarly to marked WSTSs, a marked GTS is a GTS together with a graph class closed under rule application and a subset INIT of graphs.

Definition 13 (marked GTS). A *marked GTS* is a tuple $\langle S, \mathcal{R}, \text{INIT} \rangle$ where S is a (possibly infinite) set of graphs, \mathcal{R} is a GTS with $\Rightarrow_{\mathcal{R}} \subseteq S \times S$, and $\text{INIT} \subseteq S$. We speak of a marked GTS *of bounded path length*, shortly GTS^{bp} , if S is of bounded path length and there exist $I \in \mathcal{I}$, $J \in \mathcal{J}$ (in the class of all graphs) s.t. $S = I \cap J$.

Remark. Usually, one considers S as a decidable anti-ideal, as, e.g., in [12]. Then, the basis of S is given by the empty graph. By allowing $S = I \cap J$, we can consider more arbitrary bases of graphs. This is relevant for lossiness and \perp -boundedness. A basis of S is given by $B_I \cap J$ where B_I is basis of I .

Example 3 (starry sky). The rules $\langle \emptyset \rightarrow \textcircled{A} \rangle$ and $\langle 1\textcircled{A} \rightarrow 1\textcircled{A} \rightarrow \bullet \rangle$ together with the set of disjoint unions of unboundedly many star-shaped graphs (including isolated nodes) and any subset form a marked GTS^{bp} .

We formulate the resilience problems for marked GTSS.

EXPLICIT RESILIENCE PROBLEM FOR GTSS

Given: A marked GTS $\langle S, \mathcal{R}, \text{INIT} \rangle$, ideal-based constraints Safe, Bad , a natural number $k \geq 0$.

Question: $\forall G \in \text{INIT} : \forall (G \Rightarrow^* G' \models \text{Bad}) : \exists (G' \Rightarrow^{\leq k} G'' \models \text{Safe})$?

BOUNDED RESILIENCE PROBLEM FOR GTSS

Given: A marked GTS $\langle S, \mathcal{R}, \text{INIT} \rangle$, ideal-based constraints Safe, Bad.

Question: $\exists k \geq 0 : \forall G \in \text{INIT} : \forall (G \Rightarrow^* G' \models \text{Bad}) : \exists (G' \Rightarrow^{\leq k} G'' \models \text{Safe}) ?$

In the resilience problems for WSTSs, we considered ideal-based sets. We show that one can input ideal-based constraints instead.

Lemma 6 (ideal-based graph sets). Let $S = I \cap J$ be a graph class where $I \in \mathcal{I}$ and $J \in \mathcal{J}$. For every positive (negative) constraint c , $\llbracket c \rrbracket \in \mathcal{I}$ (\mathcal{J}).

Proof. By Fact 2, for every positive (negative) constraint c , the set $\llbracket c \rrbracket$ is an (anti-)ideal. Satisfaction (\models) of negative constraints is decidable. Let c be a positive constraint and $b = \bigvee_{G \in B} \exists G$ where B is a given basis of I . Then, $b \wedge c$ is a positive constraint. By Fact 1, we can compute a positive constraint c' s.t. $\llbracket c' \rrbracket = \llbracket b \wedge c \rrbracket$ (in the class of all graphs) and c' is of the form $\bigvee_{1 \leq i \leq n} \exists G_i$ where $G_i \not\leq G_j$ for $i \neq j$. Since $J \in \mathcal{J}$, we can compute the set $\{G_i \in J : 1 \leq i \leq n\}$ which is a basis of $\llbracket c \rrbracket_S = \{G \in S : G \models c\}$. Hence, we can assume that a basis of $\llbracket c \rrbracket_S$ is given. \square

Remark. More general constraints [10,16] do not constitute (anti-)ideals w.r.t. the subgraph order, in general. Consider, e.g., the “nested” constraint $\text{AllLoop} = \forall (\text{ } \circ, \exists (\text{ } \circ \curvearrowright))$ expressing that every node has a loop. The graph consisting of one node and one loop satisfies the latter constraint. However, the bigger graph consisting of two nodes and one loop does not satisfy it. (The smaller graph consisting of a single node does not satisfy it either.) Thus, $\llbracket \text{AllLoop} \rrbracket$ is not an (anti-)ideal. Regarding the induced subgraph order [12], some “nested” constraints constitute ideals: The constraint $\exists(G, \bigwedge_{G^+ \in \text{Ext}(G)} \neg \exists(G \hookrightarrow G^+))$ expresses that the graph G is an induced subgraph of the considered graph. Here $\text{Ext}(G)$ is the set of all graphs G^+ obtained from G by adding one edge.

The following result of König & Stückrath terms a sufficient criterion for GTSSs to be well-structured.

Lemma 7 (well-structured GTS [12, Prop. 7]). Every marked GTS^{bp} induces a marked SWSTS^B (equipped with the subgraph order).

In particular, they give an effective procedure for obtaining a basis of $\text{pre}(\uparrow \{G\})$ for every given graph G . Note that in [12], König & Stückrath consider labeled hypergraphs. However, the proof in our case is the same.

Convention. When speaking of a (fin-)marked GTS^{bp}, we consider the induced (fin-)marked SWSTS^B. We also adopt the terminology for “post*-effective”, “lossy”, and “ \perp -bounded”.

We apply our results from Sect. 3 to fin-marked GTS^{bp}.

Theorem 2 (decidability for fin-marked GTSS). Both resilience problems are decidable for fin-marked GTS^{bP} s which are

- (1) post*-effective if $\text{Bad} \in \mathcal{J}_c$, $\text{Safe} \in \mathcal{I}_c$,
- (2) lossy if $\text{Bad}, \text{Safe} \in \mathcal{I}_c$.

The bounded resilience problem is decidable for fin-marked GTS^{bP} s which are

- (3) lossy and \perp -bounded if $\text{Bad} \in \mathcal{I}_c$, $\text{Safe} \in \mathcal{J}_c$,
- (4) \perp -bounded if $\text{Bad}, \text{Safe} \in \mathcal{J}_c$.

Proof. By Lemma 7 [12], every fin-marked GTS^{bP} induces a fin-marked SWSTS^B . Thus, the statements of Theorem 1 apply to GTS^{bP} s with the respective requirements. By Lemma 6, one can input ideal-based constraints instead of ideal-based sets. \square

We illustrate our decidability results by an example.

Example: Circular Process Protocol

In Fig. 6, the formalization as GTS of the circular process protocol in Sect. 1 is shown. Note that each **Clear**-rule is undefined on the node which has a c_i -labeled loop. In a rule application, this node will be deleted and recreated. Note also that each **Leave**-rule identifies two nodes.

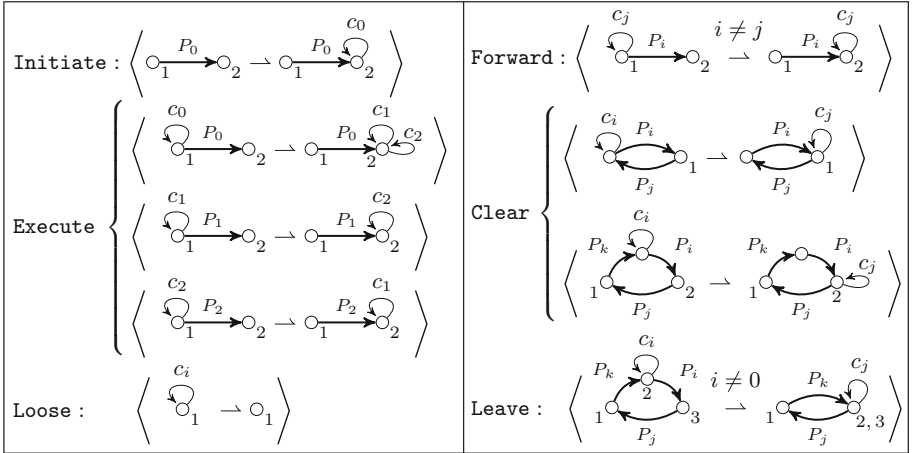


Fig. 6. Rules of the circular process protocol.

We consider all graphs with arbitrarily many commands of any kind (labeled with c_0 , c_1 , or c_2) in any collection, fitting to one of the topologies shown in Fig. 1. This graph class is of bounded path length. A basis of this graph class is given by the

topologies without any commands as in Fig. 1. The marked GTS^{bp} is lossy since the rules for loosing a command c_i may be applied to any graph containing a command c_i . It is \perp -bounded since we can reach a graph with the same topology but containing no commands by (i) initiating a command c_0 , (ii) forwarding it to the collection of P_0 , (iii) clearing all collections one after another, and (iv) loosing the only remaining command. By Proposition 2, it is post^* -effective.

The example constraints for BAD and SAFE in Sect. 1 can be expressed as positive/negative constraints:

$$\begin{aligned} \text{AllEnabled} &= \exists \left(\begin{array}{c} c_1 \\ \begin{array}{ccc} P_0 & & P_1 \\ \circ & \xrightarrow{c_1} & \circ \\ \circ & \xrightarrow{c_2} & \circ \\ P_2 & & \end{array} \\ c_0 \end{array} \right) \vee \bigvee_{i=1,2} \exists \left(\begin{array}{c} c_i \\ \begin{array}{ccc} P_0 & & \\ \circ & \xrightarrow{c_i} & \circ \\ \circ & \xrightarrow{c_i} & \circ \\ P_i & & \end{array} \\ c_0 \end{array} \right), \\ \text{Collection}(c_0, c_1) &= \exists \left(\begin{array}{c} c_0 \\ \begin{array}{ccc} & & c_1 \\ \circ & \xrightarrow{c_0} & \circ \\ \circ & \xrightarrow{c_0} & \circ \\ & & \end{array} \end{array} \right), \quad \text{Command}(c_2) = \exists \left(\begin{array}{c} c_2 \\ \begin{array}{ccc} & & \\ \circ & \xrightarrow{c_2} & \circ \\ \circ & \xrightarrow{c_2} & \circ \\ & & \end{array} \end{array} \right), \\ \text{No3Processes} &= \neg \exists \left(\begin{array}{c} P_0 & & P_1 \\ \circ & \xrightarrow{c_0} & \circ \\ \circ & \xrightarrow{c_1} & \circ \\ P_2 & & \end{array} \right), \quad \text{NoCommand} = \bigwedge_{i=0,1,2} \neg \exists \left(\begin{array}{c} c_i \\ \begin{array}{ccc} & & \\ \circ & \xrightarrow{c_i} & \circ \\ \circ & \xrightarrow{c_i} & \circ \\ & & \end{array} \end{array} \right). \end{aligned}$$

It can be verified that the given k 's in the following table are minimal.

BAD	$\neg \text{AllEnabled}$	$\text{Command}(c_2)$	AllEnabled	NoCommand
SAFE	AllEnabled	$\text{Collection}(c_0, c_1)$	$\neg \text{AllEnabled}$	No3Processes
k	6	4	1	5

By clearing the collection of P_i , it is not enabled. Thus, for the third instance, $k_{\min} = 1$ since $k_{\min} \neq 0$. Using the algorithms presented in Sect. 3, we can compute k_{\min} for the remaining cases.¹

5 Rule-Specific Criteria

We identify sufficient and handy GTS criteria for the requirements in Theorem 2. These criteria comprise properties of the rules.

Definition 14 (rule properties). A rule $\langle L \xrightarrow{R} R \rangle$ is *node-bijecitve* if p is bijective on the nodes. It is *preserving* if p is total and injective. A GTS is *node-bijecitve (preserving)* if all its rules are *node-bijecitve (preserving)*.

For lossiness and \perp -boundedness, we consider sets of rules contained in a GTS in order to reach smaller graphs (but not smaller than basis elements).

Assumption. Let S be a graph class over A and B a basis of S .

¹ If $\text{SAFE} \in \mathcal{J} \setminus \mathcal{I}$, our method provides only the answer whether there is a bound k .

Each lossy rule deletes one item (node or edge/loop) outside of a basis element.² Therefore they are constructed s.t. in each rule, a basis element is present.

Construction 1 (lossy rules). The set $\mathcal{R}_{\text{loss}}(S)$ of *lossy rules* w.r.t. S are constructed as follows.

- (1) For graphs $G \in B$, $H \in \mathcal{H}_A$, the set $\mathcal{C}(G, H)$ is defined as all graphs C s.t. $\exists(G \hookrightarrow C, H \hookrightarrow C)$ jointly surjective, and

$$\mathcal{H}_A = \left\{ \begin{array}{c} \textcircled{x}, \quad \textcircled{x} \xrightarrow{a} \textcircled{y}, \quad \textcircled{x} \xrightarrow{a} \textcircled{x} \\ \left| \quad x, y \in \Lambda_V, a \in \Lambda_E \right. \end{array} \right\}.$$

- (2) For every graph $C \in \mathcal{C}(G, H) \cap S$, every rule $\langle C \xrightarrow{p} p(C) \rangle$ where p is undefined on exactly one item (node or edge) which is not in (the image of) G and the identity otherwise, is a lossy rule.

A similar idea works for \perp -boundedness. Each bottom rule either deletes a node outside of a basis element, or deletes and recreates a node (with its incident edges) of a basis element.

Construction 2 (bottom rules). The set $\mathcal{R}_{\perp}(S)$ of *bottom rules* w.r.t. S are constructed as follows. For every basis element $G \in B$ and

- (1) for every label $x \in \Lambda_V$ s.t. $G + \textcircled{x} \in S$, the rule $\langle G + \textcircled{x} \xrightarrow{p} G \rangle$ where p is undefined on the node \textcircled{x} and the identity otherwise, is a bottom rule,³
- (2) for every node $v \in V_G$, the rule $\langle G \xrightarrow{p} G \rangle$ where p is undefined on v and its incident edges, and the identity otherwise, is a bottom rule.

For \perp -boundedness, we additionally restrict the graph class. A graph class is *node-bounded* if the number of nodes in any graph of the class is bounded.

The following result shows that the rule-specific criteria are sufficient.

Theorem 3 (criteria). A marked $\text{GTS}^{\text{bp}} \langle S, \mathcal{R}, \text{INIT} \rangle$ is

- (1) post*-effective if (INIT is finite and) \mathcal{R} is node-bijective or preserving,
- (2) lossy if $\mathcal{R}_{\text{loss}}(S) \subseteq \mathcal{R}$,
- (3) \perp -bounded if S is node-bounded and $\mathcal{R}_{\perp}(S) \subseteq \mathcal{R}$.

Proof (sketch). (1) If \mathcal{R} is preserving, the statement follows by Fact 3 since $\uparrow \text{post}^*(\text{INIT}) = \uparrow \text{INIT}$. If \mathcal{R} is node-bijective, the statement follows by the reduction in the proof of [3, Prop. 10]. For any graph G , a Petri net with initial marking is constructed s.t. reachability and the wqo correspond to $G \Rightarrow_{\mathcal{R}}^*$ and the subgraph order, respectively. Petri nets are post*-effective, see Example 2.

² In [12], “lossy rules” w.r.t. the minor order, i.e., edge contraction rules, are considered in order to obtain well-structuredness for GTSS.

³ The symbol “+” denotes the disjoint union of graphs.

(2) Using the lossy rules, we can delete any item in $G \setminus i(G_B)$ for every $G \in S$ and $G \geq G_B \in B$ where B is a basis of S and $i : G_B \hookrightarrow G$. Hence, a sequence of node and edge deletions from G to any smaller graph $G' \in S$ is also feasible via the lossy rules.

(3) Since S is node-bounded, we can delete all nodes in $G \setminus i(G_B)$ in a bounded number of steps for every $G \in S$ and $G \geq G_B \in B$ where B is a basis of S and $i : G_B \hookrightarrow G$. Then, by applying the rules for deleting and recreating, we can reach any smaller basis element in $\leq \max_{G_B \in B} |V_{G_B}|$ steps. \square

Remark. A lossy/bottom rule intended for node deletion will delete dangling edges outside of (the image of) a basis element. A bottom rule of the “second type” is intended to delete dangling edges and restore items of the basis element.

Example 4 (criteria). (1) The GTS in Fig. 6 (circular process protocol) without the **Clear**- and **Leave**-rules is node-bijective. (2) The **Loose**-rules in Fig. 6 can be adapted (see Fig. 7) s.t. they fit in our definition of lossy rules, taking into account the three basis elements in Fig. 1.

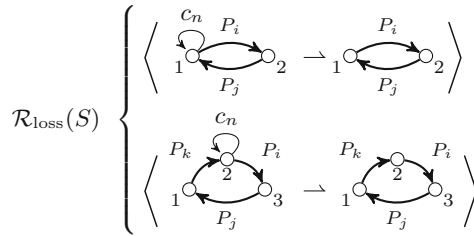


Fig. 7. The lossy rules of the circular process protocol.

(3) We adapt Example 3 (starry sky) s.t. the criterion for \perp -boundedness is fulfilled. Let D_n be the disjoint union of n A -labeled nodes and D_1^{loop} an A -labeled node with a single loop. We restrict the graph class to all graphs with exactly n A -labeled nodes and unboundedly many loops. The single basis element is the graph D_n . Consider the rule $\langle D_1 \hookrightarrow D_1^{\text{loop}} \rangle$ and the bottom rule $\langle D_n \xrightarrow{p} D_n \rangle$, i.e., deleting and recreating one node in D_n .

6 Related Concepts

The concept of resilience [11, 17] is broadly used with varying definitions.

For modeling systems, we use SPO graph transformation as in [7].

Abdulla et al. [1] show the decidability of ideal reachability (coverability), eventuality properties and simulation in (labeled) SWSTSs. We use the presented algorithm as an essential integrant of our decidability proof.

Finkel and Schnoebelen [9] show that the concept of well-structuredness is ubiquitous in computer science by providing a large class of example models. They give several decidability results for well-structured systems with varying notions of compatibility, also generalizing the algorithm of [1] to WSTSs.

König and Stückrath [12] extensively study the well-structuredness of GTSSs regarding three types of wqos (minor, subgraph, induced subgraph). All GTSSs are strongly well-structured on graphs of bounded path length w.r.t. the subgraph order. This result enables us to apply our abstract results to GTSSs. They regard Q -restricted WSTSs whose state sets have not to be a wqo but rather a subset Q of the states is a wqo. König & Stückrath develop a backwards algorithm based on [9] for Q -restricted WSTSs (GTSSs).

Bertrand et al. [3] study the decidability of reachability and coverability for GTSSs using, in parts, well-structuredness. A variety of rule-specific restrictions is investigated, e.g., containedness of node/edge-deletion rules. We use one of their results to obtain a sufficient criterion for post*-effectiveness. In contrast to [3], we stay in the framework of well-structured GTSSs.

In Fig. 8, the main results of this paper (bold boxes) are placed in the context of known results. The arrows (\rightarrow) mean “used for”, the hooked arrows (\hookrightarrow) mean “instance of” or “generalized to”. Our result for SWSTSs uses the well-known coverability algorithm [1,9] for (S)WSTSs which exploits the Noetherian property (a general concept for algebraic structures). For \perp -bounded SWSTSs, we also employ the Noetherian property. On the level of GTSSs, we use the predecessor-basis procedure of [12]. To the best of the author’s knowledge, the considered notion of resilience was first studied in [14]. We extended the latter to a systematic investigation. The result for SWSTSs in [14] (Thm. 1) corresponds to case (1) of our Theorem 1. The result for GTSSs in [14] (Thm. 2) is slightly less general than case (1) of our Theorem 2. For case (1) of Theorem 3, we use a result in [3] and well-known results for Petri nets [8].

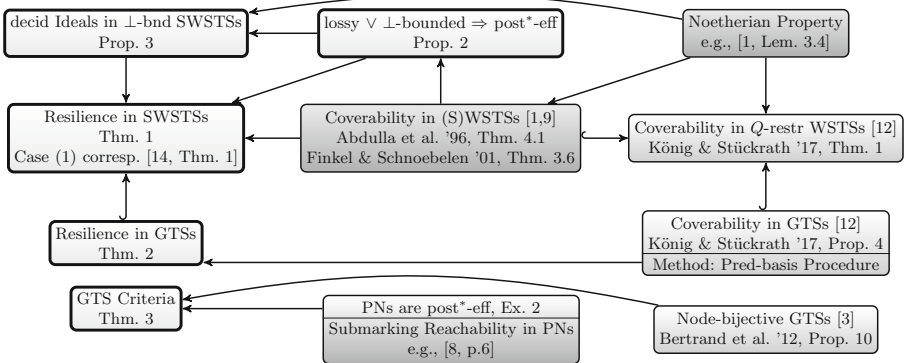


Fig. 8. Our results in the context of the theory of WSTSs and (ideal) reachability.

7 Conclusion

We provided a systematic investigation on resilience problems obtaining decidability results for subclasses of marked GTSs by using the concept of well-structuredness. The used well-quasi-order on graphs is the subgraph order, i.e., a prerequisite is the path-length-boundedness on the graph class. The requirements for decidability are post*-effectiveness or a kind of unreliability (lossy, \perp -bounded). We identified sufficient rule-specific criteria for these requirements.

For future work, we will consider (1) possibilities of a modified approach for typed graphs [6], (2) other proof methods to handle nested constraints [10], and (3) other well-quasi-orders on graphs, e.g., the induced subgraph order [12].

Acknowledgment. I am grateful to Annegret Habel, Nick Würdemann, and the anonymous reviewers for their helpful comments.

References

1. Abdulla, P.A., Cerans, K., Jonsson, B., Tsay, Y.K.: General decidability theorems for infinite-state systems. In: Proceedings of the LICS 1996, pp. 313–321. IEEE (1996). <https://doi.org/10.1109/LICS.1996.561359>
2. Apt, K.R., Olderog, E.: Verification of Sequential and Concurrent Programs. Texts and Monographs in Computer Science. Springer, Heidelberg (1991). <https://doi.org/10.1007/978-1-4757-4376-0>
3. Bertrand, N., Delzanno, G., König, B., Sangnier, A., Stückrath, J.: On the decidability status of reachability and coverability in graph transformation systems. In: 23rd International Conference on Rewriting Techniques and Applications (RTA 2012). LIPIcs, vol. 15, pp. 101–116 (2012). <https://doi.org/10.4230/LIPIcs.RTA.2012.101>
4. Ding, G.: Subgraphs and well-quasi-ordering. J. Graph Theory **16**(5), 489–502 (1992). <https://doi.org/10.1002/jgt.3190160509>
5. 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>
6. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. Monographs in Theoretical Computer Science. An EATCS Series, Springer, Heidelberg (2006). <https://doi.org/10.1007/3-540-31188-2>
7. Ehrig, H., et al.: Algebraic approaches to graph transformation - part II: single pushout approach and comparison with double pushout approach. In: Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations, pp. 247–312. World Scientific (1997). https://doi.org/10.1142/9789812384720_0004
8. Esparza, J., Nielsen, M.: Decidability issues for petri nets. BRICS Rep. Ser. **1**(8) (1994). <https://doi.org/10.7146/brics.v1i8.21662>
9. 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](https://doi.org/10.1016/S0304-3975(00)00102-X)

10. Habel, A., Pennemann, K.: Correctness of high-level transformation systems relative to nested conditions. *Math. Struct. Comput. Sci.* **19**(2), 245–296 (2009). <https://doi.org/10.1017/S0960129508007202>
11. Jackson, S., Ferris, T.L.J.: Resilience principles for engineered systems. *Syst. Eng.* **16**, 152–164 (2013). <https://doi.org/10.1002/sys.21228>
12. König, B., Stückrath, J.: Well-structured graph transformation systems. *Inf. Comput.* **252**, 71–94 (2017). <https://doi.org/10.1016/j.ic.2016.03.005>
13. Özkan, O.: Decidability of resilience for well-structured graph transformation systems. Technical report, Department of Computing Science, University of Oldenburg (2022). <https://uol.de/fs/publikationen/#c352844>
14. Özkan, O., Würdemann, N.: Resilience of well-structured graph transformation systems. In: Proceedings of 12th International Workshop on Graph Computational Models. EPTCS, vol. 350, pp. 69–88 (2021). <https://doi.org/10.4204/EPTCS.350.5>
15. Poskitt, C.M., Plump, D.: Hoare-style verification of graph programs. *Fundam. Informaticae* **118**(1–2), 135–175 (2012). <https://doi.org/10.3233/FI-2012-708>
16. Rensink, A.: Representing first-order logic using graphs. In: Ehrig, H., Engels, G., Parisi-Presicce, F., Rozenberg, G. (eds.) ICGT 2004. LNCS, vol. 3256, pp. 319–335. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30203-2_23
17. Trivedi, K.S., Kim, D.S., Ghosh, R.: Resilience in computer systems and networks. In: Proceedings of the ICCAD 2009, pp. 74–77. IEEE/ACM (2009). <https://doi.org/10.1145/1687399.1687415>