# Efficient SMT-Based Analysis of Failure Propagation

Marco Bozzano[1], Alessandro Cimatti[1], Anthony Fernandes Pires[1], Alberto Griggio[1], Martin Jonáš[1], and Greg Kimberly[2](✉)

[1] Fondazione Bruno Kessler, Trento, Italy
{bozzano,cimatti,griggio,mjonas}@fbk.eu
[2] The Boeing Company, Seattle, USA
greg.kimberly@boeing.com

**Abstract.** The process of developing civil aircraft and their related systems includes multiple phases of Preliminary Safety Assessment (PSA). An objective of PSA is to link the classification of failure conditions and effects (produced in the functional hazard analysis phases) to appropriate safety requirements for elements in the aircraft architecture. A complete and correct preliminary safety assessment phase avoids potentially costly revisions to the design late in the design process. Hence, automated ways to support PSA are an important challenge in modern aircraft design. A modern approach to conducting PSAs is via the use of abstract propagation models, that are basically hyper-graphs where arcs model the dependency among components, e.g. how the degradation of one component may lead to the degraded or failed operation of another. Such models are used for computing *failure propagations*: the fault of a component may have multiple ramifications within the system, causing the malfunction of several interconnected components. A central aspect of this problem is that of identifying the minimal fault combinations, also referred to as *minimal cut sets*, that cause overall failures.

In this paper we propose an expressive framework to model failure propagation, catering for multiple levels of degradation as well as cyclic and nondeterministic dependencies. We define a formal sequential semantics, and present an efficient SMT-based method for the analysis of failure propagation, able to enumerate cut sets that are minimal with respect to the order between levels of degradation. In contrast with the state of the art, the proposed approach is provably more expressive, and dramatically outperforms other systems when a comparison is possible.

## 1 Introduction

The process of developing civil aircraft and their related systems is guided by documents ARP4754A [17] and ARP4761 [16] produced by the engineering and standards organization SAE International. These documents describe a structured process for the safety assessment of these classes of platforms. An important stage is that of the Preliminary Aircraft Safety Assessment (PASA) and

A. Silva and K. R. M. Leino (Eds.): CAV 2021, LNCS 12760, pp. 209–230, 2021.
https://doi.org/10.1007/978-3-030-81688-9_10

Preliminary System Safety Assessment (PSSA). The PASA is followed by multiple PSSA, carried out at the level of the systems composing the aircraft. One important goal of these process stages is to link the classification of failure conditions and effects (produced in the aircraft functional hazard analysis phase) to appropriate safety requirements for elements in the aircraft architecture. These safety requirements drive, among other things, assignment of target Development Assurance Levels (DAL) for items within the architecture. A complete and correct preliminary safety assessment phase avoids potentially costly revisions to the design late in the design process. Hence, automated ways to support PSA are an important challenge in modern aircraft design [18].

An important goal of PSAs is to fully understand how faults of simple functions (e.g. providing electrical power, on-ground braking) interact and propagate to affect the overall behaviours (e.g. landing, take-off, taxiing). A modern approach to conducting such safety assessments is via propagation models [1,14,19], that model the dependency among components, e.g. how the degradation of one component may lead to the degraded or failed operation of another. Such models are used for computing *failure propagations*: the fault of a component may have multiple ramifications within the system, causing the malfunction of several interconnected components. A central problem is identifying the minimal fault combinations, also referred to as *minimal cut sets*, that cause overall failures [12].

Given that PSAs occur in the early stages of the development process when limited information regarding the design is available, reasoning is carried out at a very high level of abstraction. Therefore, instead of using behavioural models (e.g., infinite-state transition systems) adopted in formal verification, the system is more naturally modeled by a simpler formalism of propagation graphs. This does not make PSA any easier. There are in fact several aspects that must be taken into account. The first problem is the sheer size of propagation graphs, both in terms of nodes and hyper-paths to be explored, which make enumerative techniques completely inadequate.

Second, the propagation is non-Boolean [19]. That is, the degradation levels of the system functions are not binary (working vs not working) but the functions may be subject to different levels of degradation (e.g. fully operational, partly failed, completely failed), and fail in different ways (e.g. detected vs undetected, stuck open vs stuck closed), and different failures may be associated to different probabilities [19]. For example, the state of a component can be abstractly modeled into *working (w)*, *failed safe (fs)*, *failed detected (fd)*, or *failed undetected (fu)*, with degrees of degradation partially ordered as shown in Fig. 1.

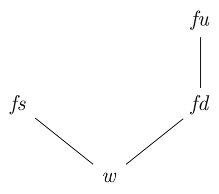


**Fig. 1.** Hasse diagram of the FDS W3F [14].

In this setting, the notion of minimality needs to take into account the order among the levels of degradation, and can not be simply considered in terms of minimality with respect to set-inclusion. Third, various forms of failure propagation may be possible, e.g., nondeterministic, temporally-constrained, cyclic. For example, the failure of a power generator may lead, within a certain amount of time, to a depleted battery and then to the loss of an engine. In turn, the loss of an engine may compromise the ability to generate power, which clearly requires the ability to deal with cyclic propagation graphs. Additionally, a failure of the control system might cause a pressure valve to become either stuck open or stuck closed; this requires the ability to deal with nondeterministic propagations.

In this paper we tackle the problem of analyzing failure propagation in the full generality required by real-world applications. We start from Finite Degradation Structures (FDS) [14], a recently-proposed modeling framework, which unifies various combinational models traditionally used in safety analysis (such as fault trees and minimal cut sets) and generalizes them to deal with different levels of degradation. We propose a framework, referred to as PGFDS (Propagation Graphs over FDS), that allows to model non-deterministic and cyclic propagation graphs. The framework is general and can be used in other safety-critical domains.

In order to deal with cyclic behaviours, PGFDS require a sequential semantics, expressed via symbolic transition systems. The computation of minimal cut sets over PGFDS can be carried out by means of techniques based on model checking, developed for the general case of behavioural models [6].

Then, we prove that it is possible to carry out the same analysis within a combinational setting, leveraging two widely adopted assumptions: that faults are persistent and that the fault propagation is monotone. These assumptions allow us to devise an efficient algorithm that can analyze fault propagations of realistic industrial benchmarks that are currently out of reach of state-of-the-art methods. The analysis of PGFDS is reduced to model enumeration for an SMT formula that does not require the explicit unrolling of the transition system. We tackle two key difficulties. The first one is to ensure causality and rule out self-supporting fault configurations in the combinational encoding. This is done by imposing cycle-breaking constraints requiring the existence of a partial order that is then constructed by the SMT solver during the analysis. The second one is to devise efficient enumeration techniques of models that are FDS-minimal, i.e., minimal with respect to *the severity of the degradation* given by the FDS. To this end, we propose an SMT-based enumerator of FDS-minimal models.

We have experimentally evaluated our approach on a comprehensive set of realistic benchmarks, also generating random systems that have a similar structure as our proprietary systems[1]. The results demonstrate substantial advances with respect to the state of the art. Our approach is clearly superior to the approach proposed in [14], that is limited to the case of acyclic deterministic PGFDS. For the cyclic PGFDSs, we contrast our approach against the sequential approach based on model-checking and show that our approach is able to scale to large PGFDS, dramatically outperforming the sequential approach.

This paper is structured as follows. In Sect. 2 we present the mathematical notation and background on FDS. In Sect. 3 we describe Propagation Graphs over FDS (PGFDS). In Sect. 4 we present the combinational encoding of PGFDS into SMT. In Sect. 5 we describe how to use the SMT encoding for the enumeration of FDS-minimal cut sets. In Sect. 6 we discuss some related work, and in Sect. 7

[1] Unfortunately the proprietary systems cannot be disclosed.

we present the experimental evaluation. In Sect. 8 we draw some conclusions and outline directions for future work.

## 2 Preliminaries

In the section, we explain the basic mathematical conventions that are used in the paper. We assume that the reader is familiar with the basic ideas of Satisfiability Modulo Theories (SMT) and in particular with the theory of linear integer arithmetic and the DPLL(T) procedure, as presented, e.g., in [2].

If convenient, we define unary functions with small domains in-place extensionally, e.g., $\{1 \mapsto 2, 2 \mapsto 3\}$ is a function with domain $\{1, 2\}$ that maps 1 to 2 and 2 to 3. We say that the $n$-ary function $f(x_1, x_2, \ldots, x_n)$ *depends* on its formal argument $x_i$ if there are some values $v_1, v_2, \ldots, v_n, v'_i$ in the corresponding domains such that $f(v_1, v_2, \ldots, v_i, \ldots v_n) \neq f(v_1, v_2, \ldots, v'_i, \ldots v_n)$. Given sets $A$ and $B$, we denote as $B^A$ the set of all functions from $A$ to $B$. Given a partially ordered set $(A, \leq)$, its subset $B \subseteq A$ is called an *upper* (resp. *lower*) set if for all $b \in B$, $a \in A$, the condition $a \geq b$ (resp. $a \leq b$) implies $a \in B$.

A Finite Degradation Structure (FDS) [14] is a triple $(FM, \leq, \bot)$, where $FM$ is a finite set of failure modes and $\leq$ is a partial order on $FM$ with the least element $\bot$. For any set $A$ and an FDS $B = (FM_B, \leq_B, \bot_B)$, the FDS $B^A$ for the set of functions from $A$ to $FM_B$ is defined as $((FM_B)^A, \leq_{B^A}, \bot_{B^A})$, where $\bot_{B^A}(a) = \bot_B$ for all $a \in A$, and $f \leq_{B^A} f'$ if and only if $f(a) \leq_B f'(a)$ for all $a \in A$. We assume that each FDS contains at least two elements. We say that an FDS is *Boolean* if it is isomorphic to the structure $(\{\bot, \top\}, \bot \leq \top, \bot)$. In the following, for an FDS $D = (FM, \leq, \bot)$, we denote elements of the set $FM$ with $f, f'$ and call them *failure modes*.

Given a first-order formula $\varphi$ over the language of the theory of linear integer arithmetic, an assignment $\mu$ that assigns a value $\mu(b) \in \{\mathbf{false}, \mathbf{true}\}$ to each free Boolean variable $b$ of $\varphi$ and a value $\mu(n) \in \mathbb{Z}$ to each free integer variable $n$ of $\varphi$ is called a model of $\varphi$ (denoted $\mu \models \varphi$) if $\mu$ makes $\varphi$ true. If $B$ is a subset of free Boolean variables of $\varphi$, the model $\mu \models \varphi$ is called *subset-minimal with respect to* $B$ if there is no model $\mu' \models \varphi$ such that $\{b \in B \mid \mu'(b) = \mathbf{true}\} \subsetneq \{b \in B \mid \mu(b) = \mathbf{true}\}$.

A *transition system* $TS$ is a tuple $(X, I, T)$ where $X$ is a set of (state) variables, $I(X)$ is a formula representing the initial states, and $T(X, X')$ is a formula representing the transitions. A *state* of $TS$ is an assignment to the variables $X$. A *trace* of $M$ is a (possibly infinite) sequence $s_0, s_1, \ldots$ of states such that $s_0 \models I$ and, for all $i \geq 0$, $s_i, s'_{i+1} \models T$.

## 3 Propagation Graphs over FDSs

In this section, we introduce our model for fault propagation, which we call Propagation Graphs over FDSs (PGFDS), and provide a sequential semantics for it which can be used to encode PGFDSs into transition systems.

Intuitively, a Propagation Graph over FDS (PGFDS) consists of a set of components of the system and of the *next* function. In each step of the failure propagation, each component is in some failure mode from the underlying FDS. In the next step of the failure propagation, each component can either 1) stay in its previous failure mode or 2) switch to an arbitrary failure mode from the set of possible next failure modes. The set of possible next failure modes for each component is given by the function *next*, based on the current failure modes of all components in the system.

**Definition 1 (Propagation Graph over FDS (PGFDS)).** *Given a finite degradation structure* $D = (FM, \leq, \bot)$*, a* propagation graph over $D$ *is a pair* $S = (C, next)$*, where*

- $C$ *is a finite set of* system components*, and*
- $next\colon C \to (FM^C \to 2^{FM})$ *is a mapping that assigns to each component* $c \in C$ *a* next failure mode function $next(c)$*, which maps failure modes of all components in* $C$ *to a set of possible next failure modes of* $c$*.*

*A* state *of* $S$ *is a mapping* $s\colon C \to FM$ *that assigns a failure mode* $f \in FM$ *to each system component* $c \in C$*.*

*Example 1.* Consider a system with three components, H (hydraulic), E (electric), and G (control on ground), over the Boolean FDS $(\{\bot, \top\}, \bot \leq \top, \bot)$. Each of the components is either working correctly (represented by the failure mode $\bot$) or incorrectly ($\top$). Component G depends on the correct functionality of either E or H. Component E depends on H to function correctly and, symmetrically, H depends on E. The failure propagation of this system can be described by a PGFDS $S = (\{\text{G}, \text{E}, \text{H}\}, next)$, where

- $next(\text{G})(s) = \{\top\}$ if $s(\text{E}) = s(\text{H}) = \top$ and $next(\text{G})(s) = \emptyset$ otherwise;
- $next(\text{E})(s) = \{\top\}$ if $s(\text{H}) = \top$ and $next(\text{E})(s) = \emptyset$ otherwise;
- $next(\text{H})(s) = \{\top\}$ if $s(\text{E}) = \top$ and $next(\text{H})(s) = \emptyset$ otherwise.

Note that $next(c)(s) = \emptyset$ means that if the system is in the state $s$, the component $c$ cannot change its current failure mode.

The structure is intuitively associated with the hypergraph depicted in Fig. 2. The dashed rectangles represent the fact that each component can fail on its own (*locally*); the hyper-arc from E and H to G is conjunctive, while the arcs incoming into a node are disjunctive. □

The important assumption of our approach is that we consider only fault-persistent propagations, i.e., fault propagations where each component can fail only once and after it does, it stays in the same failure mode forever. Note that this is a realistic assumption that is also used in other techniques for reliability analysis [5]. It is also implicitly used in other modeling techniques that are purely combinational (e.g., [19]) because they model the system only in a single time step, without considering any change in time whatsoever. Single propagation step of such computations can be described by a *fault-persistent transition relation*; the whole such computation as *fault-persistent failure propagation*.

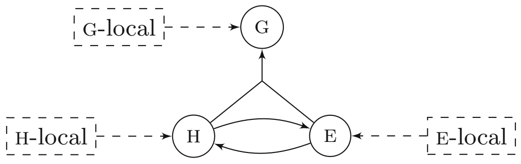


**Fig. 2.** The hypergraph view of a simple PGFDS.

**Definition 2 (Fault-persistent transition relation).** *Let $S = (C, next)$ be a* PGFDS *over an* FDS *with the least element $\perp$. The* fault-persistent transition relation *of $S$, denoted as $R_s$, is the binary relation between states of $S$ such that for all states $s, s'$, the relation $R_s(s, s')$ holds if and only if for each $c \in C$*

- *$s'(c) = s(c)$ or*
- *$s(c) = \perp$ and $s'(c) \in next(c)(s)$.*

**Definition 3 (Fault-persistent failure propagation).** *Given a* PGFDS *$S = (C, next)$, its fault-persistent transition relation $R_s$, and $k \in \mathbb{N}$, the sequence $(s_i)_{0 \leq i \leq k}$ of states of $S$ is called a* fault-persistent failure propagation *if the relation $R_s(s_i, s_{i+1})$ holds for all $0 \leq i < k$.*

Because we deal only with fault-persistent failure propagations in this paper, we from now on refer to the fault-persistent transition relation and the fault-persistent failure propagation only as *transition relation* and *failure propagation*, respectively.

**Definition 4 (Cyclic PGFDS).** *Let $S = (C, next)$ be a* PGFDS. *A component $c \in C$ depends on a component $d \in C$ iff $next(c)(s) \neq next(c)(s')$ for some $s, s' \colon C \to FM$ such that $s(d) \neq s'(d)$ and $s(c') = s'(c')$ for all $c' \neq d$. Let $deps(c) := \{d \in C \mid c \text{ depends on } d\}$, $D \subseteq C \times C$ be such that $D(c, c')$ if and only if $c' \in deps(c)$, and let $D^+$ be the transitive closure of $D$. Then we say that $S$ is* cyclic *if and only if there exists $c \in C$ such that $D^+(c, c)$ holds.*

*Example 2.* In the PGFDS $S$ from Example 1, the component G depends on components E and H, the component E depends on H, and the component H depends on E. The PGFDS $S$ is therefore cyclic because E (and also H) transitively depends on itself. □

To analyze reliability of the modeled system, it is important to identify the failures of its components (i.e., assignment of failure modes to the components) which cause the system to reach a given set of dangerous states, usually called *top level event (TLE)*. Such assignments are called *cut sets*. Since the number of all cut sets can be prohibitively large, it is often enough to identify the least severe failures in terms of the underlying FDS that are sufficient to cause the TLE. Such cut sets are called FDS-*minimal*, or *minimal* for short. These concepts are formalized in the following definitions.

**Definition 5 (Top Level Event).** *Given a* PGFDS *$S$, a* Top Level Event *(TLE) is an arbitrary set of states of $S$.*

**Definition 6 ((FDS-Minimal) Cut Set).** *Given a* PGFDS *$S = (C, next)$, and a top level event $TLE$, a* cut set *is any state $s$ for which there is a fault-persistent failure propagation that starts in $s$ and ends in some $s_k \in TLE$. A cut set is called* FDS-minimal *(or* minimal *for short) if it is minimal with respect to the pointwise ordering $\leq$ of the underlying* FDS.

Given a system $S$ and a top level event $TLE$, we denote the set of all corresponding cut sets as $CS(S, TLE)$ and the set of all minimal cut sets as $MCS(S, TLE)$. As a convention, when talking about cut sets, we will explicitly mention only the components to which the cut set assigns a failure mode different from $\bot$.

*Example 3.* Consider again the PGFDS $S$ from Example 1 and the top level event TLE $= \{s\colon \{\text{G}, \text{E}, \text{H}\} \to \{\top, \bot\} \mid s(\text{G}) = \top\}$, which corresponds to the component G not working correctly. The minimal cut sets for the PGFDS $S$ and the given top level event are

1. $\{\text{G} \mapsto \top\}$, witnessed by a failure propagation $(\{\text{G} \mapsto \top, \text{E} \mapsto \bot, \text{H} \mapsto \bot\})$ of length 1.
2. $\{\text{E} \mapsto \top\}$, witnessed by a failure propagation $(\{\text{G} \mapsto \bot, \text{E} \mapsto \top, \text{H} \mapsto \bot\}, \{\text{G} \mapsto \bot, \text{E} \mapsto \top, \text{H} \mapsto \top\}, \{\text{G} \mapsto \top, \text{E} \mapsto \top, \text{H} \mapsto \top\})$ of length 3.
3. $\{\text{H} \mapsto \top\}$, witnessed by a failure propagation $(\{\text{G} \mapsto \bot, \text{E} \mapsto \bot, \text{H} \mapsto \top\}, \{\text{G} \mapsto \bot, \text{E} \mapsto \top, \text{H} \mapsto \top\}, \{\text{G} \mapsto \top, \text{E} \mapsto \top, \text{H} \mapsto \top\})$ of length 3.

Note that besides these three minimal cut sets, there are other cut sets that are not minimal, such as $\{\text{E} \mapsto \top, \text{H} \mapsto \top\}$. □

Fault-persistent computations of a PGFDS can be easily represented as traces of a (symbolic) transition system.

**Definition 7 (Fault-persistent transition system).** *Given a* PGFDS *$S = (C, next)$ and an* FDS *$D = (FM, \leq, \bot)$, the corresponding* fault-persistent (symbolic) transition system *is given by $TS_S = (X, \mathbf{true}, T)$, where:*

- *$X = \{x_c \mid c \in C\}$ is the set of state variables, with domain $FM$;*
- *$T(X, X')$ is a symbolic encoding of the fault-persistent transition relation of $S$ as given in Definition 2. That is, for each assignment $\mu\colon X \cup X' \to FM$, $\mu \models T$ if and only if $R_s(s, s')$ holds, where $s\colon C \to FM$ is defined as $s(c) = \mu(x_c)$ (and similarly for $s'$).*

By definition, every fault-persistent computation of $S$ has a corresponding trace (of the same length) in $TS_S$. Therefore, encoding PGFDSs as transition systems allows leveraging off-the-shelf algorithms for subset-minimal cut set enumeration, such as those given in [6]. However, this might be inefficient, particularly for TLEs that are triggered by long failure propagations (corresponding to equally-long traces of the induced transition system). Moreover, as we show later, enumerating FDS-minimal cut sets is more involved.

Fault propagation systems used in practice often have the property that no transition can be disabled by additional faults, i.e., by switching a failure mode of a component from $\perp$ to $f \neq \perp$. This is also the case for the PGFDS from Example 1. Such systems are called *subset-monotone* or *monotone* for short. This is formalized by the following definition.

**Definition 8 (Subset-monotone PGFDS).** *A* PGFDS $S = (C, next)$ *is called* subset-monotone *if for all* $s, s' \colon C \to FM$*, the condition* $\forall c \in C.\ s(c) \neq \perp \to s(c) = s'(c)$ *implies* $\forall c \in C.\ next(c)(s) \subseteq next(c)(s')$.

## 4 From Sequential to Combinational

In this section, we describe a combinational encoding of fault-persistent computations of a PGFDS, which is guaranteed to be exact for subset-monotone PGFDSs and provides a useful overapproximation for general PGFDSs. In the rest of the section, let $S = (C, next)$ be a PGFDS over the FDS $D = (FM, \leq, \perp)$, and *TLE* be a top level event. We show how to construct a first-order formula $\varphi_{cs}$ over the theory of linear integer arithmetic whose models correspond to cut sets of $S$ with respect to *TLE*. In the next section, we then use this formula to enumerate all FDS-minimal cut sets of $S$.

To encode the propagations of $S$, for each component $c \in C$ and each failure mode $f \in FM$ we introduce two Boolean variables: $I_{c,f}$ and $F_{c,f}$. The variable $I_{c,f}$ encodes whether $c$ was in the failure mode $f$ in the initial state of the propagation. The variable $F_{c,f}$ encodes whether $c$ has been in the failure mode $f$ at any time during the propagation. We can then encode *TLE* as a formula $\varphi_{TLE}$ over variables $F_{c,f}$.[2]

Considering now a possible propagation, a component $c$ can be in failure mode $f \neq \perp$ at some time during the propagation for two reasons: either it was already in $f$ in the initial state of the propagation, or it transitions to $f$ because of its *next* function. The first case is represented by $I_{c,f}$ being true. The second case can be encoded as follows (for each $c \in C$ and $f \in FM \setminus \{\perp\}$):

$$\bigvee_{\substack{s\colon C \to FM \\ f \in next(c)(s)}} \ \bigwedge_{\substack{d \in deps(c) \\ s(d) \neq \perp}} F_{d,s(d)}, \tag{1}$$

stating that there must exist a row in the truth table of $next(c)$, whose result includes $f$ and which agrees with the current state on the failure modes of failed dependencies.[3] The above, however, would *not* work in the presence of cycles. This can already be seen on the simple cyclic PGFDS from Example 1.

[2] A naive encoding would be using the formula $\bigvee_{s \in TLE}(\bigwedge_{c \in C, s(c) \neq \perp} F_{c,s(c)} \wedge \bigwedge_{c \in C, s(c) = \perp} \bigwedge_{f \in FM \setminus \{\perp\}} \neg F_{c,f})$, but more compact representations are of course possible (particularly if *TLE* is given symbolically).

[3] This formula can again be encoded more compactly; particularly if the *next* function is given symbolically, which is usually the case in practice.

*Example 4.* Consider again the PGFDS $S$ from Example 1. The above-described encoding of the propagations of $S$ is

$$\begin{array}{lll}(F_{\mathrm{G},\top} & \rightarrow & (I_{\mathrm{G},\top} \vee (F_{\mathrm{E},\top} \wedge F_{\mathrm{H},\top}))) \quad \wedge \\ (F_{\mathrm{E},\top} & \rightarrow & (I_{\mathrm{E},\top} \vee F_{\mathrm{H},\top})) \quad \wedge \\ (F_{\mathrm{H},\top} & \rightarrow & (I_{\mathrm{H},\top} \vee F_{\mathrm{E},\top})).\end{array}$$

Although this encoding has a model $\mu$ such that $\mu \models \neg I_{\mathrm{G},\top} \wedge \neg I_{\mathrm{E},\top} \wedge \neg I_{\mathrm{H},\top} \wedge F_{\mathrm{G},\top} \wedge F_{\mathrm{E},\top} \wedge F_{\mathrm{H},\top}$, there is no propagation path of $S$ in which both components E and H are initially in the state $\bot$ and switch to state $\top$ during the propagation. The problem is that the encoding allows models where a failure of E was caused by a failure of H, which was in turn caused by the same failure of E. □

In order to solve the problem, we introduce constraints imposing a *causal ordering* among the components, stating that the failure of a component can be caused only by other components that precede it in the causal order. We encode this by introducing one additional integer variable $o_c$ for each component $c$, which intuitively corresponds to the time when the component $c$ switched to a failure mode different from $\bot$, and modifying the formula (1) to take the causal ordering into account:[4]

$$\bigvee_{\substack{s\colon C \rightarrow FM \\ f \in next(c)(s)}} \bigwedge_{\substack{d \in deps(c) \\ s(d) \neq \bot}} \left(F_{d,s(d)} \wedge o_d < o_c\right). \tag{2}$$

Putting it all together, the encoding for the failure mode changes is given by the formula $\varphi_{next}$ below:

$$\varphi_{next} = \bigwedge_{\substack{c \in C \\ f \in FM \setminus \{\bot\}}} (F_{c,f} \rightarrow (I_{c,f} \vee (2))) \wedge (I_{c,f} \rightarrow F_{c,f}).$$

*Example 5.* For the PGFDS $S$ from Example 1, the correct encoding of the propagations of $S$ is thus the following formula $\varphi_{next}$:

$$\begin{array}{lll}(F_{\mathrm{G},\top} & \rightarrow & (I_{\mathrm{G},\top} \vee ((F_{\mathrm{E},\top} \wedge o_{\mathrm{E}} < o_{\mathrm{G}}) \wedge (F_{\mathrm{H},\top} \wedge o_{\mathrm{H}} < o_{\mathrm{G}}))) \quad \wedge \\ (I_{\mathrm{G},\top} & \rightarrow & F_{\mathrm{G},\top}) \quad \wedge \\ (F_{\mathrm{E},\top} & \rightarrow & (I_{\mathrm{E},\top} \vee (F_{\mathrm{H},\top} \wedge o_{\mathrm{H}} < o_{\mathrm{E}}))) \quad \wedge \\ (I_{\mathrm{E},\top} & \rightarrow & F_{\mathrm{E},\top}) \quad \wedge \\ (F_{\mathrm{H},\top} & \rightarrow & (I_{\mathrm{H},\top} \vee (F_{\mathrm{E},\top} \wedge o_{\mathrm{E}} < o_{\mathrm{H}}))) \quad \wedge \\ (I_{\mathrm{H},\top} & \rightarrow & F_{\mathrm{H},\top}).\end{array}$$

Note that the constraints for causal ordering now rule out the spurious self-supporting propagation in which E fails because of H and H fails because of E.

[4] We remark that such ordering constraints are needed only if the input PGFDS is cyclic, and only between components in the same strongly connected component of the dependency graph.

This would require that $o_{\text{H}} < o_{\text{E}}$ and $o_{\text{E}} < o_{\text{H}}$ are both true, which is clearly impossible in the theory of linear integer arithmetic (or, more generally, in any theory in which $<$ is interpreted as a strict ordering relation).

The propagations of $S$ mentioned in Example 3 correspond to the following assignments:

1. The propagation for the cut set $\{\text{G} \mapsto \top\}$ corresponds to an assignment $\mu$ such that $\mu \models I_{\text{G},\top} \wedge \neg I_{\text{E},\top} \wedge \neg I_{\text{H},\top} \wedge F_{\text{G},\top} \wedge \neg F_{\text{E},\top} \wedge \neg F_{\text{H},\top}$ and $\mu(o_{\text{G}}) = \mu(o_{\text{E}}) = \mu(o_{\text{H}}) = 0$.
2. The propagation for the cut set $\{\text{E} \mapsto \top\}$ corresponds to an assignment $\mu$ such that $\mu \models \neg I_{\text{G},\top} \wedge I_{\text{E},\top} \wedge \neg I_{\text{H},\top} \wedge F_{\text{G},\top} \wedge F_{\text{E},\top} \wedge F_{\text{H},\top}$ and $\mu(o_{\text{G}}) = 2$, $\mu(o_{\text{E}}) = 0$, $\mu(o_{\text{H}}) = 1$.
3. The propagation for the cut set $\{\text{H} \mapsto \top\}$ corresponds to an assignment $\mu$ such that $\mu \models \neg I_{\text{G},\top} \wedge \neg I_{\text{E},\top} \wedge I_{\text{H},\top} \wedge F_{\text{G},\top} \wedge F_{\text{E},\top} \wedge F_{\text{H},\top}$ and $\mu(o_{\text{G}}) = 2$, $\mu(o_{\text{E}}) = 1$, $\mu(o_{\text{H}}) = 0$.

These assignments are not unique; there are infinitely many choices for the values of the ordering variables $o_c$. Also note that there is no global causality ordering for the system: the causality ordering is different for different propagations. $\square$

Finally, we encode the fault-persistence constraint by stating that no component can be in two failure modes either in the initial state of the propagation or at any time during the propagation:

$$\varphi_{once} = \bigwedge_{\substack{c \in C \\ f, f' \in FM \setminus \{\bot\} \\ f \neq f'}} (\neg I_{c,f} \vee \neg I_{c,f'}) \wedge (\neg F_{c,f} \vee \neg F_{c,f'}) .$$

The final formula is then given by $\varphi_{cs}$:

$$\varphi_{cs} = \varphi_{TLE} \wedge \varphi_{next} \wedge \varphi_{once}.$$

As the following theorem shows, the formula $\varphi_{cs}$ for general systems encodes an *overapproximation* of the set $CS(S, TLE)$. The reason for this is that the encoding does not enforce failure mode of dependencies that are working, i.e., are in the failure mode $\bot$. Note that even an overapproximation of $CS(S, TLE)$ is useful for safety analysis; it can be used, for example, for computing an upper bound on the probability of failure of the system. Moreover, if the system $S$ is *subset-monotone*, which is often the case in practice, the formula $\varphi_{cs}$ is guaranteed to encode the set $CS(S, TLE)$ exactly.

To formulate the relationship precisely, we define the function that provides the correspondence between the models of $\mu$ and the cut sets of $S$. Observe that thanks to $\varphi_{once}$, each model $\mu$ of $\varphi_{cs}$ corresponds to a unique initial state *modelToState*$(\mu)$ of $S$ as defined below:

$$\mathit{modelToState}(\mu)(c) = \begin{cases} f, & \text{if } \{f' \in FM \setminus \{\bot\} \mid \mu(I_{c,f'}) = \mathbf{true}\} = \{f\}, \\ \bot, & \text{if } \{f' \in FM \setminus \{\bot\} \mid \mu(I_{c,f'}) = \mathbf{true}\} = \emptyset. \end{cases}$$

```
MCS-enumeration($\varphi_{cs}$, $modelToState$):
1.  solver := SMT-solver()
2.  res := $\emptyset$
3.  assert-formula(solver, $\varphi_{cs}$)
4.  for $I_{c,f} \in vars(\varphi_{cs})$:
5.      add-preferred-var(solver, $I_{c,f}$, false)
6.  while check-sat(solver):
7.      $\mu$ := get-model(solver)
8.      $\psi$ := true
9.      for $I_{c,f} \in vars(\varphi_{cs})$:
10.         if $\mu(I_{c,f}) = $ true:
11.             $\psi := \psi \wedge I_{c,f}$
12.     res := res $\cup \{modelToState(\mu)\}$
13.     assert-formula(solver, $\neg\psi$)
14.  return res
```

**Fig. 3.** SMT-based MCS enumeration algorithm.

**Theorem 1.** *For an arbitrary* PGFDS $S$ *and a top level event TLE,*

$$CS(S, TLE) \quad \subseteq \quad \{modelToState(\mu) \mid \mu \models \varphi_{cs}\}.$$

*Moreover, if $S$ is subset-monotone, these sets are equal.*

## 5 Enumeration of FDS-Minimal Cut Sets

In this section, we show how to efficiently enumerate FDS-minimal cut sets of subset-monotone systems using the formula $\varphi_{cs}$ and an SMT solver. We first consider a simplified case, in which the underlying FDS $D$ is Boolean. We then show how to generalize our solution to arbitrary FDSs.

### 5.1 Algorithm for Boolean FDSs

The pseudo-code of our procedure for the case when the underlying FDS is Boolean is shown in Fig. 3. Intuitively, the algorithm enumerates all the subset-minimal models of $\varphi_{cs}$ with respect to the set of variables of form $I_{c,f}$. These models are enumerated one by one and each enumerated model is, together with all its supermodels, blocked by the assertion on line 13, until the formula becomes unsatisfiable. Each model of the formula is converted to a cut set by the function *modelToState*.

The algorithm makes use of a DPLL(T)-based SMT solver that provides the following functionalities:

1. An assert-formula method that allows to add constraints incrementally;
2. A check-sat method to determine the satisfiability of the current set of constraints;

3. A get-model method that returns a model for the current asserted set of constraints, in case they are satisfiable;
4. An add-preferred-var method that allows to control the branching heuristics of the internal SAT engine of the solver, such that whenever a SAT decision needs to be performed, variables in the preferred set are always considered before the other variables for branching, and are assigned the value specified in the add-preferred-var call.[5]

The correctness for our algorithm is formalized by the theorem below.

**Theorem 2 (MCS enumeration over Boolean FDS).** *For a subset-monotone* PGFDS $S$ *over the Boolean* FDS*, the result of* $\mathit{MCS-enumeration}(\varphi_{cs}, \mathit{modelToState})$ *is the set of all* FDS*-minimal cut sets of* $S$*.*

*Proof.* Let $S = (C, \mathit{next})$ be a subset-monotone PGFDS. It was proven by Di Rosa et al. [15] that if branching heuristics of a CDCL-based SAT solver are modified to assign **false** to a subset $V$ of variables before branching on other variables (lines 4–5 of our pseudocode), the produced model is subset-minimal with respect to the set of variables $V$. This claim straightforwardly extends to DPLL(T)-based SMT solvers. In every iteration, the algorithm thus finds one subset-minimal model $\mu$ of $\varphi_{cs}$ with respect to the set of variables $I_{c,f}$ and adds a constraint that prevents enumerating any model $\mu'$ such that $\{I_{c,f} \in \mathit{vars}(\varphi_{cs}) \mid \mu(I_{c,f}) = \mathbf{true}\} \subseteq \{I_{c,f} \in \mathit{vars}(\varphi_{cs}) \mid \mu'(I_{c,f}) = \mathbf{true}\}$ in the following iterations. Therefore, the described algorithm enumerates, for each model $\overline{\mu}$ of the formula $\exists\{F_{c,f} \mid c \in C, f \in \mathit{FM}\}\, \exists\{o_c \mid c \in C\}\, (\varphi_{cs})$ that is subset-minimal with respect to the set of variables $I_{c,f}$, exactly one model $\mu$ of $\varphi_{cs}$ that agrees with $\overline{\mu}$ on all variables $I_{c,f}$.

Note that $\mathit{vars}(\varphi_{cs})$ does not contain the variable $I_{c,\perp}$ for any $c \in C$. For a Boolean FDS and models $\mu, \mu' \models \varphi_{cs}$, we thus have $\{I_{c,f} \in \mathit{vars}(\varphi_{cs}) \mid \mu(I_{c,f}) = \mathbf{true}\} \subseteq \{I_{c,f} \in \mathit{vars}(\varphi_{cs}) \mid \mu'(I_{c,f}) = \mathbf{true}\}$ if and only if $\mathit{modelToState}(\mu) \leq \mathit{modelToState}(\mu')$. Therefore, Theorem 1 implies that for subset-monotone $S$, subset-minimal models of $\varphi_{cs}$ with respect to the set of variables of form $I_{c,f}$ precisely correspond to FDS-minimal cut sets of $S$ and the correspondence is given by the function $\mathit{modelToState}$. □

## 5.2 Extension to Arbitrary FDSs

The algorithm of Fig. 3 does not work in general for arbitrary FDSs, but only for the FDSs in which all the failure modes different from $\perp$ are incomparable. The problem is that the assumption that a cut set is FDS-minimal iff the corresponding model of $\varphi_{cs}$ is subset-minimal with respect to the set of variables $I_{c,f}$ with $f \neq \perp$ does not hold in general with the encoding of Sect. 4, as can be seen on the following simple example.

[5] For example, calling add-preferred-var(solver, $v$, **true**) means that if the solver has to perform a case split, $v$ will be assigned before all non-preferred variables, and it will always be assigned to true by the branching heuristic.

$$\begin{tikzcd}
& & \{w, \mathit{fd}, \mathit{fu}\} \equiv w \land \neg \mathit{fs} \land \mathit{fd} \land \mathit{fu} \arrow[d, dash] \\
\{w, \mathit{fs}\} \equiv w \land \mathit{fs} \land \neg \mathit{fd} \land \neg \mathit{fu} \arrow[dr, dash] & & \{w, \mathit{fd}\} \equiv w \land \neg \mathit{fs} \land \mathit{fd} \land \neg \mathit{fu} \arrow[dl, dash] \\
& \{w\} \equiv w \land \neg \mathit{fs} \land \neg \mathit{fd} \land \neg \mathit{fu} &
\end{tikzcd}$$

**Fig. 4.** Hasse diagram of the ordered set $(W3F\downarrow, \subseteq)$ together with the encoding of the elements as formulas.

*Example 6.* Consider the FDS $D = (\{\bot, m, \top\}, \bot \leq m \leq \top, \bot)$ and the PGFDS $S = (\{c\}, \mathit{next})$ with $\mathit{next}(c)(s) = \emptyset$ for all $c$ and $s$. Intuitively, $S$ contains one component that cannot change its failure mode during the computation. Consider further the top-level event $\mathit{TLE} = \{\{c \mapsto m\}, \{c \mapsto \top\}\}$.

Both $\{c \mapsto \top\}$ and $\{c \mapsto m\}$ are cut sets, but only the latter is FDS-minimal. However, the algorithm of Fig. 3 will return both, since they both correspond to subset-minimal models with respect to the set of variables $I_{c,f}$. □

We can adapt the procedure of Fig. 3 to arbitrary FDSs by using an encoding in which the ordering of assignments to the $I_{c,f}$ variables corresponds to the severity ordering $\leq$ of the underlying FDS $D$. In order to do this, we exploit the isomorphism between $D = (\mathit{FM}, \leq, \bot)$ and the poset $D\downarrow$ of its lower subsets generated by single elements defined as $D\downarrow = \{\{f' \in \mathit{FM} \mid f' \leq f\} \mid f \in \mathit{FM}\}$ with partial order $\subseteq$ and the least element $\{\bot\}$. For example, the poset $(W3F\downarrow, \subseteq)$ for the FDS $W3F$ of Fig. 1 is shown in Fig. 4, together with an encoding of the elements as formulas.

With this isomorphism in mind, we define for each $c \in C$ and $f \in \mathit{FM}$ the formula $\psi_{c=f}$ that represents the failure mode $f$ of component $c$ by assigning the subset of variables $\{I_{c,\hat{f}} \mid \hat{f} \leq f\}$ to true:

$$\psi_{c=f} = \bigwedge_{\hat{f} \in \mathit{FM}, \hat{f} \leq f} I_{c,\hat{f}} \land \bigwedge_{\hat{f} \in \mathit{FM}, \hat{f} \not\leq f} \neg I_{c,\hat{f}}.$$

The important property of this definition is that for all $c \in C$, $f, f' \in \mathit{FM}$ and assignments $\mu \models \psi_{c=f}$ and $\mu' \models \psi_{c=f'}$, we have $f \leq f'$ if and only if $\{I_{c,\hat{f}} \mid \mu(I_{c,\hat{f}}) = \textbf{true}\} \subseteq \{I_{c,\hat{f}} \mid \mu'(I_{c,\hat{f}}) = \textbf{true}\}$.

We then modify the encoding $\varphi_{cs}$ of Sect. 4 as follows:

1. First, we modify $\varphi_{next}$ to encode the initial state by using $\psi_{c=f}$ instead of $I_{c,f}$. This ensures that the ordering of assignments to the initial variables reflects the ordering given by the underlying FDS. We also remove the mutual exclusion constraints on the variables $I_{c,f}$ from $\varphi_{once}$, because the mutual exclusion of initial failure modes is now guaranteed by the definition of $\psi_{c=f}$:

$$\varphi_{next} = \bigwedge_{\substack{c \in C \\ f \in FM \setminus \{\bot\}}} (F_{c,f} \rightarrow (\psi_{c=f} \vee (2))) \wedge (\psi_{c=f} \rightarrow F_{c,f}),$$

$$\varphi_{once} = \bigwedge_{\substack{c \in C \\ f,f' \in FM \setminus \{\bot\} \\ f \neq f'}} (\neg F_{c,f} \vee \neg F_{c,f'}) \,.$$

2. Then, we add domain constraints that ensure that the resulting formula represents only models with assignments to $I_{c,f}$ that correspond to elements of $D{\downarrow}$:

$$\varphi_{D\downarrow} = \bigwedge_{c \in C} \bigvee_{f \in FM} \psi_{c=f}.$$

The new encoding is then given by $\varphi_{cs}^{FM}$:

$$\varphi_{cs}^{FM} = \varphi_{TLE} \wedge \varphi_{next} \wedge \varphi_{once} \wedge \varphi_{D\downarrow}.$$

The modified encoding $\varphi_{cs}^{FM}$ represents the cut sets in a different way: instead of representing the failure modes directly by $I_{c,f}$ as in $\varphi_{cs}$, they are now represented by the subformulas $\psi_{c=f}$. Therefore, to prove correctness of the modified encoding, the function *modelToState* that maps models to cut sets also has to be changed. We define the initial state $\mathit{modelToState}^{FM}(\mu)$ corresponding to the model $\mu$ by $\mathit{modelToState}^{FM}(\mu)(c) = \max\{f \in FM \mid \mu(I_{c,f}) = \mathbf{true}\}$. Note that the maximum is guaranteed to exist because of the $\varphi_{D\downarrow}$ constraint.

**Theorem 3.** *For an arbitrary* PGFDS *$S$ and a top level event $TLE$,*

$$CS(S, TLE) \quad \subseteq \quad \{\mathit{modelToState}^{FM}(\mu) \mid \mu \models \varphi_{cs}^{FM}\}.$$

*Moreover, if $S$ is subset-monotone, these sets are equal.*

Therefore, the algorithm MCS-enumeration from Fig. 3 can be used to enumerate FDS-minimal cut sets of a subset-monotone PGFDS, given as the inputs the modified encoding $\varphi_{cs}^{FM}$ and the modified function $\mathit{modelToState}^{FM}$. This is formalized by the following theorem:

**Theorem 4 (MCS enumeration for general FDS).** *For a subset-monotone* PGFDS *$S$ over an* FDS *$D$, the result of MCS-enumeration($\varphi_{cs}^{FM}$, $\mathit{modelToState}^{FM}$) is the set of all* FDS*-minimal cut sets of $S$.*

Note that our encoding of FDS-minimality is general and does not depend on the algorithm for enumeration of subset-minimal models. Indeed, thanks to our encoding, any off-the-shelf minimal-model enumerator can be used to enumerate FDS-minimal models. Therefore, any improvements to minimal model enumeration directly translate to improved performance of our method for FDS-minimal cut set enumeration. From the opposite point of view, our encoding can in principle be employed by other tools to reduce FDS-minimal cut set enumeration to subset-minimal cut set enumeration.

## 6 Related Work

Finite Degradation Models (FDMs) [14] are an algebraic framework accommodating the concept of fault degradation, where faults may have different values organized into a semi-lattice. Using FDMs (probabilistic) safety analysis (fault trees and minimal cut sets) can be generalized from Boolean models to multi-state systems. Compared to FDMs, fault-persistent PGFDSs differ in two significant aspects: first, since the function *next* returns a set of possible next failure modes, PGFDSs allow non-determinism in the failure propagation, i.e., the failure of a component is not *uniquely* determined by the failure modes of its dependencies. Second, and more importantly, PGFDSs allow cyclic dependencies and give them well-defined and expected semantics. Since the work on FDMs is the closest to ours, we shall discuss it in detail below.

In [8] the authors present a framework for failure propagation which enables modeling sets of failure modes using a domain specific language. It is less expressive than FDMs, in that sets of failure modes cannot be related by degradation orders, which significantly simplifies the enumeration of MCSs. Finally, classical formalisms for failure propagation, but less expressive than FDS, include FPTN [9] and Hip-HOps [11].

TFPGs (Timed Failure Propagation Graphs) [1] extend fault propagation model by enabling the specification of time bounds and mode constraints on the propagation links. However, TFPGs do not consider degradation, and they do not support cyclic dependencies. Conversely, the PGFDS formalism can be easily extended to support time bounds, failure probabilities, mode constraints, and constraints on propagation delays similar to those available in TFPGs (e.g., following [5]). Moreover, once the minimal cut sets of a PGFDS are computed, the existing approach to computing probability of overall failure [5] can be used almost unchanged.

Finally, xSAP [3] is a safety analysis platform that supports library-based fault models and the generation of safety artifacts for fully general behavioral models, e.g., it can generate fault trees and minimal cut sets for arbitrary transition systems [6]. Currently, xSAP does not support FDS and degradation models.

### 6.1 Detailed Comparison with Finite Degradation Models

As outlined above, the formalism Finite Degradation Models (FDMs), introduced in [14], is closely related to our PGFDS. Here, we describe FDM in further detail and show that PGFDS are a strict generalization of FDS, obtained by (i) considering non-determinism in the propagation of failures, and (ii) by allowing cyclic dependencies among the components.

Each FDM has *state variables*, which correspond to the sources of failures in the system, and *flow variables*, which correspond to the propagated consequences of these failures. Each flow variable has an associated *equation*, which prescribes the failure mode of the corresponding flow variable based on the failure modes of state variables and other flow variables. We assume that the failure modes of all state and flow variables are modeled by the FDS $D = (FM, \leq, \bot)$.[6]

**Definition 9 (Finite Degradation Model** [14]**).** *Given an arbitrary* FDS $D = (FM, \leq, \bot)$*, a Finite Degradation Model (*FDM*) is a pair* $M = (\mathcal{V} = \mathcal{S} \uplus \mathcal{F}, \mathcal{E})$*, where*

- $\mathcal{S} = \{V_1, \ldots, V_m\}$ is a finite set of *state variables*,
- $\mathcal{F} = \{W_{m+1}, \ldots, W_{m+n}\}$ is a finite set of *flow variables*,
- $\mathcal{E} = \{W_{m+1} := \phi_{m+1}, \ldots W_{m+n} := \phi_{m+n}\}$ is a finite set of *equations*, where each $\phi_{m+i}$ for $1 \leq i \leq n$ is a function of type $FM^{\mathcal{V}} \rightarrow FM$.

We say that a flow variable $W_{m+i}$ *depends on* a variable $v$ if the function $\phi_{m+i}$ depends on $v$. An FDM is called acyclic if there are no cyclic dependencies among its flow variables, i.e., no flow variable transitively depends on itself. We stress out that in contrast to our definitions of PGFDS, the original paper [14] only deals with acyclic FDMs and does not provide semantics and necessary definitions for cyclic FDMs. We thus assume in the rest of the section that all FDMs are acyclic.

An assignment $\sigma\colon \mathcal{V} \rightarrow FM$ is called *admissible* if the failure modes assigned to the flow variables satisfy all the corresponding equations, i.e., $\sigma(W_{m+i}) = \phi_{m+i}(\sigma)$ for each $1 \leq i \leq n$. The assumption of acyclicity of FDMs, together with the fact that all equations are deterministic functions and not general relations, guarantees that in each admissible assignment, failure modes of the flow variables are uniquely determined by the failure modes of the state variables. This defines a function $[\![M]\!](\sigma) = \overline{\sigma}_M$, which maps each state variable assignment $\sigma$ to its unique admissible extension $\overline{\sigma}_M$ that assigns values to all variables. This is a stark contrast to PGFDS, where a single initial state can give rise to multiple different propagation paths.

A corresponding notion to our notion of *top level event* for FDM is the notion of *observer*. An observer is a pair $(R, U)$, where $R$ is a flow variable and $U \subseteq FM$ is a set of failure modes. Intuitively, the observer represents a set of dangerous failure modes of the given flow variable. A *cut set* is any assignment $\sigma\colon \mathcal{S} \rightarrow FM$ of failure modes to state variables such that $\overline{\sigma}(R) \in U$.

A notion related to our notion of *monotonicity* for FDM is *coherence*. The observer is coherent if for all assignments $\sigma, \sigma'\colon \mathcal{S} \rightarrow FM$ such that $\sigma$ is a cut set and $\sigma \leq \sigma'$, the assignment $\sigma'$ is also a cut set.

Each FDM $M$ can be translated to a PGFDS $S_M$ such that the cut sets of $M$ correspond to the cut sets of $S_M$. Moreover, if the FDM $M$ is coherent, the resulting PGFDS $S_M$ is guaranteed to be subset-monotone. This enables efficient analysis of coherent FDMs by our SMT-based technique. Intuitively, the PGFDS $S_M$ has one component for each state variable of $M$ and an additional component $R$ for the observer flow variable $R$. The *next* function is defined in a way that the failure modes of all the components that correspond to state variables cannot

[6] Both FDMs and our PGFDS can be defined over multiple different FDSs for different variables. Such generalization is straightforward, but it complicates the notation and the exposition significantly.

**Table 1.** Classes of PGFDS and their traces that each of the compared tools can handle precisely.

| Tool | FDS | Cycles | Nondeterministic | Not fault-persistent |
|---|---|---|---|---|
| Emmy | **Arbitrary** | No | No | No |
| xSAP | Boolean | **Yes** | **Yes** | **Yes** |
| SMT-PGFPS | **Arbitrary** | **Yes** | **Yes** | No |

change and that the component $R$ can switch to a predefined set of failure modes if $\overline{\sigma}(R) \in U$. This is achieved by composing all equations for the flow variables. If local variables are used in the symbolic encoding[7], the size of the result is guaranteed to be polynomial.

## 7 Experimental Evaluation

To evaluate the performance and scalability of our approach, we have implemented the proposed algorithm MCS-enumeration in a simple Python tool that uses the solver MathSAT [7], which supports all the required functionalities that are described in Sect. 5.1. In this section, we refer to the tool as SMT-PGFPS.

As a comparison, we have used Emmy [13], a tool based on decision diagrams for the enumeration of FDS-minimal cut sets of FDMS, and xSAP [3], a tool for safety assessment for arbitrary transition systems. Each of these tools only supports a subset of the capabilities of our approach, as summarized in Table 1.

**Emmy** supports minimal cut set enumeration with respect to an arbitrary ordering of failure modes given by an FDS, but only for acyclic and deterministic FDMS;

**xSAP** supports analysis of arbitrary transition systems with cycles, given that it internally relies on the nuXmv model checker. However, it cannot enumerate FDS-minimal cut sets, but only subset-minimal ones. Note that for computation of subset-minimal cut sets, xSAP is more general than our approach, as it supports general transition systems and arbitrary temporal properties. However, we use xSAP as a baseline to compare performance and scalability of our approach for cyclic PGFDS because it is a subcase of general transition systems that is important in practice. In the evaluation, we use the IC3-based engine described in [6] (denoted as xSAP-IC3). Note that this algorithm assumes that the verified property is monotone and leverages this assumption for efficiency.

[7] For example, let-expressions of form `(let ((var definition) ...) body)` in SMT-LIB.

For the comparison, we have created three sets of benchmarks:

**Scalable acyclic benchmarks** consisting of linear structures extended by a triple modular redundancy scheme. The basic architecture of these structures is parameterized by its size $n$ and the system contains $6n$ components: $3n$ modules and $3n$ voters. These benchmarks use the FDS W2F, which is a restriction of the FDS W3F of Fig. 1 to failure modes $\{w, fd, fu\}$, with the ordering $w < fd < fu$.
Note that FDS-minimal cut sets of these benchmarks cannot be enumerated by xSAP, as the benchmarks use a non-Boolean FDS.

**Randomly generated systems with cycles over Boolean FDS** which share some structural properties with real-world systems. In particular, we generated random systems that have a similar distribution of in-degrees and out-degrees of the components as our proprietary systems, which we cannot disclose. We have generated 950 such systems of sizes ranging between 50 and 1000 components. We have used the Boolean FDS for these benchmarks, so that they can be precisely analyzed also by xSAP.
Note that these benchmarks cannot be solved by Emmy, as they contain cyclic dependencies among the components.

**Randomly generated systems over W2F** which are created from the above-mentioned randomly generated systems by using the FDS W2F instead of the Boolean one. Although this does not change the overall structure of the system, it makes the transition relation more complicated and significantly increases number of minimal cut sets.
In the evaluation, we only used systems of size at most 400, as both the compared approaches timed out on the vast majority of larger systems.
Note that these benchmarks cannot be solved by Emmy, as they contain cyclic dependencies among the components. They can be solved by xSAP, but the generated cut sets are only subset-minimal with respect to fault variables, and not (in general) FDS-minimal.

For the scalable benchmarks, we have generated encodings in the SMT format described in this paper and in the FDS-ML format used by Emmy. For the randomly generated cyclic benchmarks, we have generated encodings in the SMT format and in the SMV format used by xSAP. The SMV encodings also include the assumption of *fault-persistence*. All the used benchmarks are *subset-monotone*, and therefore our SMT-based approach can be used to compute the set of minimal cut sets correctly.

We have used wall time limit of 30 min for each solver-benchmark pair. All experiments were performed on a Linux laptop with Intel Core i7-8665U CPU and 32 GiB of RAM.

A comparison of SMT-PGFPS and Emmy on the scalable acyclic benchmarks can be seen in Table 2. It shows that Emmy times out already on systems of size 5, i.e., on systems with 30 components. On the other hand, our approach is able to scale to systems with three thousand components.

A comparison against the sequential approach of xSAP on cyclic benchmarks can be seen in Fig. 5. Figures 5a and 5b show that on random systems over

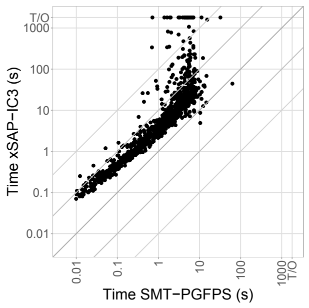
**Table 2.** Numbers of minimal cut sets (*#MCS*) and solving times for top level events *failed detected (fd)* and *failed undetected (fu)* on linear systems extended with triple-modular redundancy scheme. Note that the system with size $n$ consists of $6n$ components (column *#Comp*).

| | | Failed detected | | | Failed undetected | | |
|---|---|---|---|---|---|---|---|
| Size | #Comp | #MCS | Emmy (s) | SMT (s) | #MCS | Emmy (s) | SMT (s) |
| 1 | 6 | 4 | 0.051 | 0.001 | 7 | 0.071 | 0.001 |
| 2 | 12 | 16 | 0.137 | 0.002 | 31 | 0.172 | 0.003 |
| 3 | 18 | 28 | 3.052 | 0.004 | 55 | 3.141 | 0.007 |
| 4 | 24 | 40 | 160.493 | 0.006 | 79 | 163.556 | 0.013 |
| 5 | 30 | 52 | >1800 | 0.009 | 103 | >1800 | 0.017 |
| 10 | 60 | 112 | >1800 | 0.032 | 223 | >1800 | 0.063 |
| 100 | 600 | 1192 | >1800 | 3.456 | 2383 | >1800 | 6.748 |
| 500 | 3000 | 5992 | >1800 | 171.042 | 11983 | >1800 | 328.737 |

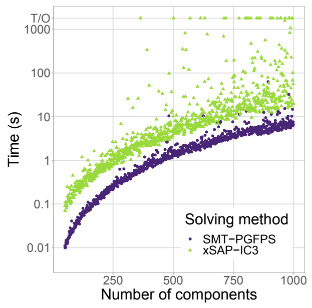
Boolean FDSs, our approach significantly outperforms the sequential approach of xSAP. As the size of the system grows, the difference can be up to several orders of magnitude. Both xSAP and SMT-PGFPS compute exactly the same minimal cut sets. Hence, the dramatic difference in performance can be justified by the reduction to the combinational case, which prevents the unrolling of the transition relation by implicitly encoding the propagations in the total ordering(s) found by the SMT solver.

The performance difference on the systems over the FDS W2F, shown in Figures 5c and 5d, is even more pronounced. This can be caused by two additional factors. First, the systems over the FDS W2F have more complicated transition relation, more minimal cut sets, and are in general harder. Thus, the unrolling performed by xSAP is even more costly. Second, xSAP has to enumerate more cut sets, because it is enumerating all subset-minimal cut sets and not only FDS-minimal cut sets. However, this cannot be the main source of the observed performance gap: on 35 from the 113 benchmarks on which both xSAP and SMT-PGFPS finished before timeout, the number of cut sets are the same; on the remaining 78 benchmarks, xSAP enumerates on average 6% more cut sets and at most 62% more cut sets. In order to obtain FDS-minimal cut sets from xSAP, the produced subset-minimal cut sets would have to be filtered or explicitly minimized, which would add yet another performance penalty for xSAP.

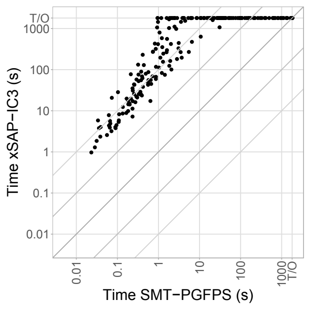
Overall, the SMT-based techniques presented in this paper yield a fundamental advancement with respect to the state of the art, both in terms of expressiveness as well as in terms of performance.



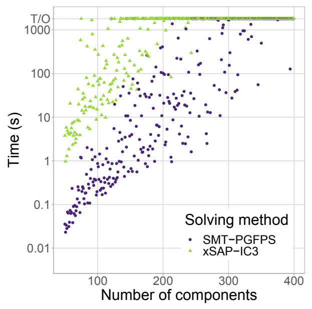
(a) Scatter plot of solving times over Boolean FDS.



(b) Dependence of solving time on the number of components over Boolean FDS.



(c) Scatter plot of solving times over FDS W2F.



(d) Dependence of solving time on the number of components over FDS W2F.

**Fig. 5.** Comparison of SMT-PGFPS and xSAP-IC3 on random cyclic systems.

## 8 Conclusions and Further Work

We tackled the problem of supporting the Preliminary Safety Assessment phase of aircraft design. Specifically, we defined an expressive framework for modeling failure propagation over components with multiple levels of degradation, with nondeterminism and cyclic dependencies. We presented a sequential semantics and proved that the problem can be tackled by means of minimal models enumeration in SMT. The framework is more expressive than the state of the art, and the proposed method outperforms the BDD-based techniques from [14] on acyclic benchmarks over generic FDSs, and the model checking techniques of [6] on cyclic benchmarks.

In the future, we are going to introduce timing constraints and analyze redundancy architectures. We also investigate ways to relax the monotonicity and fault-persistence assumptions to explore recovery mechanisms and to further extend the reach of our approach. We are also working on encoding the causality constraints in the frameworks of SAT modulo acyclicity [10] and ASP modulo acyclicity [4], which could improve the performance of our approach even further.

## References

1. Abdelwahed, S., Karsai, G., Mahadevan, N., Ofsthun, S.C.: Practical implementation of diagnosis systems using timed failure propagation graph models. IEEE Trans. Instrum. Meas. **58**(2), 240–247 (2009)
2. Barrett, C.W., Sebastiani, R., Seshia, S.A., Tinelli, C.: Satisfiability modulo theories. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability, volume 185 of Frontiers in Artificial Intelligence and Applications, pp. 825–885. IOS Press (2009)
3. Bittner, B., et al.: The xSAP safety analysis platform. In: Chechik, M., Raskin, J.-F. (eds.) TACAS 2016. LNCS, vol. 9636, pp. 533–539. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49674-9_31
4. Bomanson, J., Gebser, M., Janhunen, T., Kaufmann, B., Schaub, T.: Answer set programming modulo acyclicity. Fundamenta Informaticae **147**(1), 63–91 (2016)
5. Bozzano, M., Cimatti, A., Gario, M., Micheli, A.: SMT-based validation of timed failure propagation graphs. In: Bonet, B., Koenig, S. (eds.) Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, Texas, USA, 25–30 January 2015, pp. 3724–3730. AAAI Press (2015)
6. Bozzano, M., Cimatti, A., Griggio, A., Mattarei, C.: Efficient anytime techniques for model-based safety analysis. In: Kroening, D., Păsăreanu, C.S. (eds.) CAV 2015. LNCS, vol. 9206, pp. 603–621. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21690-4_41
7. Cimatti, A., Griggio, A., Schaafsma, B.J., Sebastiani, R.: The MathSAT5 SMT solver. In: Piterman, N., Smolka, S.A. (eds.) TACAS 2013. LNCS, vol. 7795, pp. 93–107. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36742-7_7
8. Delmas, K., Delmas, R., Pagetti, C.: SMT-based architecture modelling for safety assessment. In: 12th IEEE International Symposium on Industrial Embedded Systems, SIES 2017, Toulouse, France, 14–16 June 2017, pp. 1–8. IEEE (2017)
9. Fenelon, P., McDermid, J.A.: An integrated tool set for software safety analysis. J. Syst. Softw. **21**(3), 279–290 (1993)
10. Gebser, M., Janhunen, T., Rintanen, J.: SAT modulo graphs: acyclicity. In: Fermé, E., Leite, J. (eds.) JELIA 2014. LNCS (LNAI), vol. 8761, pp. 137–151. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11558-0_10
11. Papadopoulos, Y., McDermid, J.A.: Hierarchically performed hazard origin and propagation studies. In: Felici, M., Kanoun, K. (eds.) SAFECOMP 1999. LNCS, vol. 1698, pp. 139–152. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48249-0_13
12. Rauzy, A.: Mathematical foundations of minimal cutsets. IEEE Trans. Reliab. **50**(4), 389–396 (2001)
13. Rauzy, A., Yang, L.: Decision diagram algorithms to extract minimal cutsets of finite degradation models. Information **10**(12), 368 (2019)
14. Rauzy, A., Yang, L.: Finite degradation structures. FLAP **6**(6), 1447–1474 (2019)
15. Di Rosa, E., Giunchiglia, E., Maratea, M.: Solving satisfiability problems with preferences. Constraints Int. J. **15**(4), 485–515 (2010). https://doi.org/10.1007/s10601-010-9095-y
16. SAE: ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment, December 1996
17. SAE: ARP4754A Guidelines for Development of Civil Aircraft and Systems, December 2010

18. Wang, P.: Civil Aircraft Electrical Power System Safety Assessment: Issues and Practices. Butterworth-Heinemann, Oxford (2017)
19. Yang, L., Rauzy, A., Haskins, C.: Finite degradation structures: a formal framework to support the interface between MBSE and MBSA. In: IEEE International Systems Engineering Symposium (ISSE), Rome, Italy, pp. 1–6 (2018)