Safety-Complete Test Suites

Wen-ling Huang[2] and Jan Peleska[1,2(✉)]

[1] Verified Systems International GmbH, Bremen, Germany
[2] Department of Mathematics and Computer Science,
University of Bremen, Bremen, Germany
{huang,jp}@cs.uni-bremen.de

Abstract. In this paper, a novel safety-related variant of complete test suites for finite state machines is introduced. Under certain hypotheses which are similar to the ones used in the well-known W-Method or the Wp-Method, the new method guarantees to uncover every safety violation, while erroneous outputs without safety-relevance may remain undetected. In well-defined situations that can be precisely pre-determined from the reference model, this leads to a substantial reduction of test cases in comparison to the size of the analogous Wp-test suites. We advocate this new test suite for situations, where exhaustive testing of the complete system is too expensive. In these cases, strong guarantees with respect to fault coverage should only be given for the errors representing safety violations, while it is considered as acceptable if less critical errors remain undetected.

Keywords: Model-based testing · Complete testing theories · Safety

1 Introduction

Motivation. Complete test suites guarantee to uncover all conformance violations of the implementation under test checked against a given reference model, provided that certain hypotheses – typically captured in a fault model – are fulfilled. This ideal test strength has attracted many researchers over the last 50 years, so that a large variety of contributions exists (a comprehensive overview has been given in [4, Sect. 5]). On the other hand, the often infeasible size of the test suites involved has frequently prevented their practical application. As a result, there is a considerable interest in testing strategies allowing to focus the effort on certain critical properties, while requiring lesser fault coverage for non-critical ones; we name [7] as one example among a multitude of publications in this field which is typically denoted as *property-oriented testing*.

Main Contributions. A novel contribution to property-oriented testing for the domain of finite state machines is presented. Our approach modifies the well-known Wp-Method in such a way, that complete coverage for output and

N. Yevtushenko et al. (Eds.): ICTSS 2017, LNCS 10533, pp. 145–161, 2017.
DOI: 10.1007/978-3-319-67549-7_9

transition faults (including addition of new states) is guaranteed, if these lead to erroneous outputs representing safety-violations. To this end, an abstraction concept for outputs is introduced, so that it can be formally captured whether an erroneous replacement of another output for the expected one presents a safety violation or just a non-critical deviation. In contrast to other publications in this field, we formally prove that our strategy is complete with respect to this safety-related fault coverage. We show by means of examples, that applying this *Safety-complete Wp-Method* can lead to significantly reduced test suites in comparison to the Wp-Method, though this is not guaranteed, but depends on the nature of the reference model and its safety-related abstraction.

Overview. In Sect. 2, basic terms and concepts are introduced, so that this paper remains sufficiently self-contained. In Sect. 3, the Safety-complete Wp-Method is introduced, and its completeness properties are proven. In Sect. 4, three small case studies are presented that provide some insight into the situations where the new method leads to a significant test case reduction. Section 5 presents the conclusion.

2 Notation and Technical Background

A *deterministic finite state machine (DFSM)* is a tuple $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$ denoting the finite state space Q, initial state $\underline{q} \in Q$, finite input and output alphabets Σ_I and Σ_O, and the transition relation $h \subseteq Q \times \Sigma_I \times \Sigma_O \times Q$. For deterministic machines, pre-state q and input x uniquely determine the associated output y and the post-state q', such that $h(q, x, y, q')$ holds. We assume that all DFSMs are *completely specified*. This means that for every q and every x, there exists y and q' such that $h(q, x, y, q')$.

The *after operator* q-after-$\overline{x}$ maps a pre-state q and a finite sequence $\overline{x}$ of inputs to the uniquely determined post-state q' resulting from repetitive application of h. The *language* of a DFSM is the set of finite input/output traces $\overline{x}/\overline{y} \in \Sigma_I^* \times \Sigma_O^*$ resulting from applying all $\overline{x} \in \Sigma_I^*$ to the initial state $\underline{q}$ and associating the output trace $\overline{y}$ which is uniquely determined by $\underline{q}$, $\overline{x}$, and h. Two DFSMs are *I/O-equivalent* ($M \sim M'$) if they produce the same language. The language of a state q is the set of $\overline{x}/\overline{y}$ generated by applying all $\overline{x} \in \Sigma_I^*$ to q. The *prime machine* $\mathbf{prime}(M)$ of a DFSM M is the minimal DFSM producing the same language as M.

A *test suite* is a subset $\mathrm{TS} \subseteq \Sigma_I^*$, each $\overline{x} \in \mathrm{TS}$ is a *test case*. This simplified notation is possible, since only deterministic machines are considered, so that the input trace $\overline{x}$ uniquely determines the output trace to be expected according to the reference DFSM. An implementation *passes* a test case $\overline{x}$ if the application of this input sequence produces an output sequence $\overline{y}$, such that $\overline{x}/\overline{y}$ is in the language of the reference DFSM.

For sets of input traces $A, B \subseteq \Sigma_I^*$, the expression $A.B$ denotes the set of all input traces resulting from concatenating a trace $\overline{x} \in A$ with a trace $\overline{x}' \in B$. Given a collection of sets of input traces indexed over the states of a DFSM M,

say, $W_q \subseteq \Sigma_I^*, q \in Q$, the notation $A \oplus \{W_q \mid q \in Q\}$ is used to denote the set of all input traces $\overline{x}.\overline{x}'$ where $\overline{x} \in A$ and $\overline{x}' \in W_{(\underline{q}\text{-after-}\overline{x})}$. Σ_I^k denotes the set of all input traces of length $k \geq 0$. For input or output traces $\overline{z} = z_1 \ldots z_k$, the following notation is used for trace sections.

$$\overline{z}^{[i,j]} = z_i.z_{i+1} \ldots z_j \text{ where } 1 \leq i \leq j \leq k.$$

Given a reference DFSM M, the *W-Method* defines the test suite

$$\mathcal{W}(M) = V.\bigcup_{i=0}^{m-n+1} \Sigma_I^i.W,$$

where V is a *state cover* and W is a *characterisation set*. A state cover is a set of input traces, such that every state of M can be reached by $\underline{q}$-after-$\overline{x}$ for some $\overline{x} \in V$. V contains the empty trace ε which "reaches" the initial state of M. It is assumed that $\mathbf{prime}(M)$ has n states and that the prime machine representing the true behaviour of the SUT has at most $m \geq n$ states. A characterisation set W contains input traces distinguishing all states of $\mathbf{prime}(M)$. This means that for each pair of distinct states q, q' of $\mathbf{prime}(M)$, there exists an $\overline{x} \in W$ such that $\overline{x}$ applied to q produces an output trace which differs from the one resulting from application of $\overline{x}$ to q'. It is shown in [1,10] that test suite $\mathcal{W}(M)$ uncovers every violation of I/O-equivalence, provided that the prime machine representing the true behaviour of the implementation does not have more than m states.

The *Wp-Method* [2,8] is an alternative test strategy which has the same test strength as the W-Method, but requires fewer test cases.

$$\mathcal{W}_p(M) = V.W \cup \left(V.\bigcup_{i=0}^{m-n} \Sigma_I^i.W\right) \cup \left(V.\Sigma_I^{m-n+1} \oplus \{W_q \mid q \in Q(\mathbf{prime}(M))\}\right)$$

Here V, W are defined as above. The *state identification sets* W_q are subsets of W, such that each W_q contains sufficient input traces to distinguish q from every other state in $\mathbf{prime}(M)$.

3 A Safety-Complete Wp-Method

3.1 Safety-Related Output Abstractions

Let $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$ be a deterministic completely specified FSM. Then any reflexive and transitive relation $\leq_s \subseteq \Sigma_O \times \Sigma_O$ is called a *safety-related output abstraction*. The intuition behind this definition is that $y \leq_s y'$ indicates that an erroneous output of y' instead of an expected output y does not induce a safety violation. Reflexivity just indicates that the occurrence of the output expected according to the reference model M can never be a safety violation. Transitivity implies that output z must also be a safe replacement of w, if $w \leq_s y \wedge y \leq_s z$ holds. Relation $\leq_s$ induces an equivalence relation $\sim_s$ on $\Sigma_O \times \Sigma_O$ by defining

$$y_1 \sim_s y_2 \equiv y_1 \leq_s y_2 \wedge y_2 \leq_s y_1$$

Example 1. Consider a train onboard controller which compares actual train speed against the allowed speed and progressively outputs

$$\Sigma_O = \{\texttt{ok}, \texttt{warning}, \texttt{ServiceBrakeTrigger}, \texttt{EmergencyBrakeTrigger}\},$$

depending on how much the train is overspeeding. The outputs `ok` and `warning` are shown on the display unit of the train engine driver, whereas the outputs `ServiceBrakeTrigger` and `EmergencyBrakeTrigger` directly act on the train's braking system. The service brake slows the train down with lower braking force than the emergency brake, so that the latter is used only as the "last resort", when warnings and service brake interventions do not suffice. These considerations induce a safety-related output abstraction $\leq_s$ as the reflexive and transitive closure of

$$\texttt{ok} \leq_s \texttt{warning} \leq_s \texttt{ServiceBrakeTrigger} \leq_s \texttt{EmergencyBrakeTrigger}$$

The intuition behind this definition is that a warning or even a braking intervention performed by the controller is an acceptable substitute for an expected `ok`-output from the safety perspective: the substitute output may be a nuisance (a spurious warning when the speed is within range) or even a severe reduction of reliability (triggering the emergency brake without need), but it does not introduce a safety threat. The same holds for situations where the service brake should be triggered but instead, the emergency brakes are activated.

When an intervention by service brakes or emergency brakes is expected, however, an output `ok` or `warning` would certainly be regarded as a safety hazard.

Next, suppose that the outputs to the train engine driver are extended by status messages

$$\Sigma'_O = \{\mathtt{s}_1, \ldots, \mathtt{s}_n\}.$$

Since these informative messages have no safety-relevance at all, we wish to extend the relation $\leq_s$ in a way expressing that each status message can be replaced by any other output of $\Sigma_O \cup \Sigma'_O$ without causing any safety hazard. This is achieved by extending $\leq_s$ according to the rules

$$\mathtt{s} \sim_s \mathtt{s}' \text{ for all } \mathtt{s}, \mathtt{s}' \in \Sigma'_O$$
$$\mathtt{s} \leq_s \mathtt{e} \text{ for all } \mathtt{s} \in \Sigma'_O, \mathtt{e} \in \Sigma_O$$

Finally consider a design extension, where the onboard controller operates in a de-centralised distributed train control environment, so that it switches its own points

$$\Sigma''_O = \{\mathtt{p}_i^+,\ \mathtt{p}_i^- \mid i = 1, \ldots, m\}$$

along the route (such a system has been investigated, for example, in [3]). Notation $\mathtt{p}_i^+$ stands for switching point number i into the straight position, $\mathtt{p}_i^-$ for switching the point into the branching position. From the safety-perspective, switching a point into the desired position cannot be replaced by any other event without introducing a safety hazard. Therefore we extend $\leq_s$ this time as follows.

$$p \leq_s p \text{ for all } \mathtt{p} \in \Sigma_O''$$
$$\mathtt{s} \leq_s \mathtt{p} \text{ for all } \mathtt{s} \in \Sigma_O', \mathtt{p} \in \Sigma_O'' \qquad \square$$

Given a safety-related output abstraction $\leq_s$ on Σ_O, this is extended in the natural way to a reflexive and transitive relation (again denoted by $\leq_s$) on output traces $\iota, \pi \in \Sigma_O^*$ by setting

$$\iota \leq_s \pi \equiv \big(\#\iota = \#\pi \wedge \forall i \in \{1, \ldots, \#\iota\} : \iota(i) \leq_s \pi(i)\big)$$

for $\iota, \pi \in \Sigma_O^*$.

Now let q, q' be two states of the same state machine or of different state machines over the same input/output alphabet (Σ_I, Σ_O). In the latter case, it is assumed without loss of generality that their states come from disjoint sets Q, Q'. Then it is possible to specify a joint output function $\omega : (Q \cup Q') \times \Sigma_I \to \Sigma_O$ which is extended in the natural way to operate on sequences of inputs, i.e. $\omega : (Q \cup Q') \times \Sigma_I^* \to \Sigma_O^*$. Let $\overline{x} \in \Sigma_I^*$ be an input trace. We define

$$q' \overset{\overline{x}}{\leq_s} q \equiv \big(\omega(q', \overline{x}) \leq_s \omega(q, \overline{x})\big).$$

Intuitively speaking, $q' \overset{\overline{x}}{\leq_s} q$ states that applying input trace $\overline{x}$ to state q produces an output sequence $\omega(q, \overline{x})$ which is an admissible substitute to the output sequence $\omega(q', \overline{x})$ expected when applying the same input sequence to q'.

Relation $\overset{\overline{x}}{\leq_s}$ induces an equivalence relation on states by defining

$$q' \overset{\overline{x}}{\sim_s} q \equiv \big(q' \overset{\overline{x}}{\leq_s} q \wedge q \overset{\overline{x}}{\leq_s} q'\big)$$

These relations can be extended to sets of input traces in the natural way by defining

$$q' \overset{W}{\leq_s} q \equiv \big(\forall \overline{x} \in W : q' \overset{\overline{x}}{\leq_s} q\big)$$
$$q' \overset{W}{\sim_s} q \equiv \big(\forall \overline{x} \in W : q' \overset{\overline{x}}{\sim_s} q\big)$$

for arbitrary $W \subseteq \Sigma_I^*$. Finally, the specific case where $W = \Sigma_I^*$ is written in the simplified notation

$$q' \leq_s q \equiv \big(q' \overset{\Sigma_I^*}{\leq_s} q\big)$$
$$q' \sim_s q \equiv \big(q' \overset{\Sigma_I^*}{\sim_s} q\big).$$

If $q' \sim_s q$ holds, any input trace applied to q' will lead to an output trace which – regarded from the safety perspective – is an admissible replacement of the outputs expected when applying the same inputs to q and vice versa. If the initial states $\underline{q}$ and $\underline{q'}$ of two state machines M, M' are s-equivalent ($\underline{q'} \sim_s \underline{q}$), we denote this by $M' \sim_s M$.

We call $W_s \subseteq \Sigma_I^*$ an *s-characterisation set of DFSM* M, if and only if

$$q' \overset{W_s}{\sim_s} q \Leftrightarrow q' \sim_s q$$

holds. For any $q \in Q$, $W_{s_q} \subseteq W_s$ is called an *s-state identification set of* q, if and only if

$$\forall q' \in Q : \left(q' \overset{W_{s_q}}{\sim_s} q \Leftrightarrow q' \sim_s q\right)$$

holds. The sets of input traces in W_s and W_{s_q}, respectively, allow to distinguish states from the perspective of their safety-relevant outputs. Conversely, different states are indistinguishable by W_s and W_{s_q}, if their safety-relevant outputs are equivalent, while the non-relevant outputs may differ for certain input traces.

Note that W_s and W_{s_q} coincide with the conventional characterisation sets and state identification sets introduced in [8], if we choose $\leq_s$ to be the reflexive and transitive relation defined by the *diagonal of* Σ_O, that is,

$$\leq_s = \mathbf{diag}(\Sigma_O) = \{(y, y) \mid y \in \Sigma_O\},$$

where every output is only comparable to itself.

The following lemma states an obvious but useful fact about the $\leq_s$-relation, input prefixes, and input suffixes.

Lemma 1. *Let* $\overline{x} = x_1 \ldots x_k$ *and* $1 \leq i < k$. *Let* $q, q' \in Q \cup Q'$ *satisfying* $q' \overset{\overline{x}}{\leq_s} q$. *Define states*

$$q_i = q\text{-}after\text{-}\overline{x}^{[1,i]}$$
$$q'_i = q'\text{-}after\text{-}\overline{x}^{[1,i]}$$

Then $q'_i \overset{\overline{x}^{[i+1,k]}}{\leq_s} q_i$ *holds.* □

3.2 A Safety-Complete Variant of the Wp-Method

Throughout this section, let $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$, $M' = (Q', \underline{q}', \Sigma_I, \Sigma_O, h')$ be completely specified, deterministic, and minimised FSMs over the same input/output alphabet $\Sigma = \Sigma_I \times \Sigma_O$ with $|Q| = n$, $|Q'| \leq m$ and $m \geq n$.

Definition 1 (Safety-related Fault Model). *Let* $\leq_s \subseteq \Sigma_O \times \Sigma_O$ *be a safety-related output abstraction with associated equivalence relation* $\sim_s$. *A* safety-related fault model

$$\mathcal{F} = (M, \sim_s, \mathcal{D}(m))$$

is composed of

1. *the* reference model M,
2. *the* conformance relation $\sim_s$, *and*
3. *the* fault domain $\mathcal{D}(m)$ *consisting of all finite, completely specified, deterministic, and minimised state machines* M' *over input/output alphabet* Σ, *such that* $|Q'| \leq m$ *and* $m \geq n$. □

Definition 2 (Safety-complete Test Suite). *With the definitions above, let $TS \subseteq \Sigma^*$ be a test suite.*

1. *TS is called* sound *w.r.t. fault model $\mathcal{F}$, if and only if every member $M' \in \mathcal{D}(m)$ which is I/O-equivalent to M ($M' \sim M$) passes the test suite.*
2. *TS is called* safety-exhaustive *w.r.t. fault model $\mathcal{F}$, if and only if every member $M' \in \mathcal{D}(m)$ which is not safety-equivalent to M ($M' \not\sim_s M$) fail at least one test case in TS.*
3. *TS is called* safety-complete *w.r.t. fault model $\mathcal{F}$, if it is both sound and safety-exhaustive.* □

Theorem 1. *Let $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$, $M' = (Q', \underline{q}', \Sigma_I, \Sigma_O, h')$ be two completely specified, deterministic, mimimal FSMs with $|Q| = n$, $|Q'| \leq m$ and $m \geq n$. Let $\leq_s \subseteq \Sigma_O \times \Sigma_O$ be a safety-related output abstraction. Suppose that*

1. *$\varepsilon \in V \subseteq \Sigma_I^*$ is a state cover of M,*
2. *$W \subseteq \Sigma_I^*$ is a characterisation set of M, and*
3. *$W_s \subseteq \Sigma_I^*$ is an s-characterisation set of M.*

Define

$$A = V.W \text{ and } B = V. \bigcup_{i=0}^{m-n+1} \Sigma_I^i.W_s$$

Then

$$\underline{q}' \overset{A}{\sim} \underline{q} \wedge \underline{q}' \overset{B}{\sim}_s \underline{q}$$

implies $\underline{q}' \sim_s \underline{q}$ and therefore $M' \sim_s M$.

Proof. We prove by induction over $|\overline{x}|$ that for any $\overline{x} \in \Sigma_I^*$,

1. $\underline{q}' \overset{\overline{x}}{\sim}_s \underline{q}$.
2. $\underline{q}'$-after-$\overline{x} \overset{W_s}{\sim}_s \underline{q}$-after-$\overline{x}$

Statement 2 is an auxiliary assertion needed to prove Statement 1. The latter directly implies the statement of the theorem.

Induction Base. Statements 1 and 2 trivially hold for $\overline{x} = \varepsilon$.

Induction Hypothesis. Suppose that Statements 1 and 2 are true for some $k \geq 0$.

Induction Step. Let $\overline{x}.x \in \Sigma_I^*$ be any input trace with $|\overline{x}| = k$ and $x \in \Sigma_I$. Let

$$q = \underline{q}\text{-after-}\overline{x}/\overline{y},\ q_1 = \underline{q}\text{-after-}(\overline{x}/\overline{y}).(x/y)$$

and

$$q' = \underline{q}'\text{-after-}\overline{x}/\overline{y}', q_1' = \underline{q}'\text{-after-}(\overline{x}/\overline{y}').(x/y').$$

From induction hypothesis we have $q' \overset{W_s}{\sim}_s q$.

Since $\underline{q}' \overset{V.W}{\sim} \underline{q}$ and since W is a characterisation set of M, the set

$$\{\underline{q}'\text{-after-}\overline{x} \mid \overline{x} \in V\}$$

contains $n = |M|$ states of M'. Consequently,

$$V' = V.\bigcup_{i=0}^{m-n} \Sigma_I^i$$

is a state cover of M'. Therefore, there exists some input trace $\pi \in V'$ such that $\underline{q}'\text{-after-}\pi = q'$ (note that it is not necessarily the case that $\overline{x} \in V'$). Let $q_2 = \underline{q}\text{-after-}\pi$. We wish to show that $q \overset{W_s}{\sim}_s q_2$ holds.

Assume that $\pi \in V.\Sigma_I^i$ for some $i \in \{0, \ldots, m-n\}$. Now assumption $\underline{q}' \overset{B}{\sim}_s \underline{q}$ of the theorem, together with Lemma 1 implies again that $q' \overset{W_s}{\sim}_s q_2$.

This fact is now combined with the induction hypothesis which implies that $q' \overset{W_s}{\sim}_s q$. From these facts we can conclude that $q \overset{W_s}{\sim}_s q_2$. Now W_s is an s-characterisation set of M, so $q \overset{W_s}{\sim}_s q_2$ implies $q \sim_s q_2$.

Let $q_3 = q_2\text{-after-}(x/y_1)$. Then $q \sim_s q_2$, $\omega(q, x) = y$, and $\omega(q_2, x) = y_1$ implies $y \sim_s y_1$. From Lemma 1 we have $q_1 = q\text{-after-}x \sim_s q_3 = q_2\text{-after-}x$. Since $\pi.x \in V.\bigcup_{i=1}^{m-n+1} \Sigma_I^i$ we have $q_1' \overset{W_s}{\sim}_s q_3$ and $y \sim_s y'$. Hence we have $\underline{q}' \overset{\overline{x}.x}{\sim}_s \underline{q}$ and $\underline{q}'\text{-after-}(\overline{x}.x) \overset{W_s}{\sim}_s \underline{q}\text{-after-}(\overline{x}.x)$, which proves the induction step. □

Theorem 2. *Let $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$, $M' = (Q', \underline{q}', \Sigma_I, \Sigma_O, h')$ be two completely specified, deterministic, minimal FSMs with $|Q| = n$, $|Q'| \leq m$ and $m \geq n$. Let $\leq_s \subseteq \Sigma_O \times \Sigma_O$ be a safety-related output abstraction. Suppose that*

1. *$\varepsilon \in V \subseteq \Sigma_I^*$ is a state cover of M,*
2. *$W \subseteq \Sigma_I^*$ is a characterisation set of M,*
3. *$W_s \subseteq \Sigma_I^*$ is an s-characterisation set of M, and*
4. *$W_{s_q} \subseteq W_s$ are s-identification sets of M for all $q \in Q$.*

Define

$$A = V.W, \quad C = V.\bigcup_{i=0}^{m-n} \Sigma_I^i.W_s, \text{ and } D = V.\Sigma_I^{m-n+1} \oplus \{W_{s_q} \mid q \in Q\}.$$

Then

$$\underline{q}' \overset{A}{\sim} \underline{q} \wedge \underline{q}' \overset{C}{\sim}_s \underline{q} \wedge \underline{q}' \overset{D}{\sim}_s \underline{q}$$

implies $\underline{q}' \sim_s \underline{q}$, and therefore $M' \sim_s M$.

Proof. From Theorem 1 we conclude that it suffices to prove that the assumptions of Theorem 2 imply the validity of $\underline{q}' \overset{B}{\sim}_s \underline{q}$, with $B = V.\bigcup_{i=0}^{m-n+1} \Sigma_I^i.W_s$

as specified in Theorem 1. Since Theorem 2 is already based on the assumption $\underline{q}' \overset{C}{\sim}_s \underline{q}$, it suffices to prove that

$$\underline{q}' \overset{V.\Sigma_I^{m-n+1}.W_s}{\sim_s} \underline{q}$$

holds.

Let $\overline{x} \in V.\Sigma_I^{m-n+1}$ and define $q = \underline{q}$-after-$\overline{x}$ and $q' = \underline{q}'$-after-$\overline{x}$. We need to show that $q' \overset{W_s}{\sim}_s q$ follows from the assumptions of the theorem.

From assumption $\underline{q}' \overset{D}{\sim}_s \underline{q}$ and Lemma 1, we already have $q' \overset{W_{s_q}}{\sim}_s q$. Since $V' = V.\bigcup_{i=0}^{m-n} \Sigma_I^i$ is a state cover of M' (this has been established in the proof of Theorem 1), there is some $\overline{x}' \in V'$ such that $\underline{q}'$-after-$\overline{x}' = q'$. Let $q_2 = \underline{q}$-after-$\overline{x}'$. Then $q' \overset{W_s}{\sim}_s q_2$ follows from assumption from $\underline{q}' \overset{C}{\sim}_s \underline{q}$ (this has also been shown in detail in the proof of Theorem 1). Since $W_{s_q} \subseteq W_s$, the fact that $q' \overset{W_s}{\sim}_s q_2$ holds implies that $q' \overset{W_{s_q}}{\sim}_s q_2$ holds as well. In combination with $q' \overset{W_{s_q}}{\sim}_s q$, this implies $q_2 \overset{W_{s_q}}{\sim}_s q$, Therefore, $q_2 \sim_s q$ and $q_2 \overset{W_s}{\sim}_s q$. From $q' \overset{W_s}{\sim}_s q_2$ and $q_2 \overset{W_s}{\sim}_s q$, we conclude that $q' \overset{W_s}{\sim}_s q$. This completes the proof. □

The theorem above induces a safety-complete test suite, this time it is based on the original Wp-Method.

Corollary 1 (Safety-complete Wp-Method). *Let $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$ be a completely specified, deterministic, minimal FSM with $|Q| = n$, and let m be a fixed integer satisfying $m \geq n$. Let $\leq_s \subseteq \Sigma_O \times \Sigma_O$ be a safety-related output abstraction. Using the notation and terms introduced in Definitions 1 and 2, and Theorem 2. Then the test suite*

$$TS = V.W \cup \left(V. \bigcup_{i=0}^{m-n} \Sigma_I^i.W_s\right) \cup \left(V.\Sigma_I^{m-n+1} \oplus \{W_{s_q} \mid q \in Q\}\right)$$

is safety-complete with respect to fault model $\mathcal{F} = (M, \sim_s, \mathcal{D}(m))$. □

3.3 Implementation

For implementing an algorithm calculating the safety-complete test suite according to Corollary 1, we proceed as follows.

FSM Abstraction. Given a completely specified, deterministic, minimal FSM $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$, every safety-related output abstraction $\leq_s \subseteq \Sigma_O \times \Sigma_O$ induces an abstraction α_s of the alphabet by mapping each output $y \in \Sigma_O$ to the set of outputs $y' \in \Sigma_O$ that are greater or equal to y according to $\leq_s$.

$$\alpha_s : \Sigma_O \to \mathbb{P}(\Sigma_O); \quad y \mapsto \{y' \in \Sigma_O \mid y \leq_s y'\}$$

The image $\Sigma_O^s = \alpha_s(\Sigma_O)$ is again finite, therefore it can be used as a new output alphabet of a state machine M_s which is an abstraction of M with respect to $\leq_s$ in the following sense.

$$M_s = \mathbf{prime}(Q, \underline{q}, \Sigma_I, \Sigma_O^s, h_s)$$
$$h_s = \{(q, x, \alpha_s(y), q') \mid (q, x, y, q') \in h\}$$

Though M is assumed to be already minimised, the abstracted machine $(Q, \underline{q}, \Sigma_I, \Sigma_O^s, h_s)$ will not be minimised in general, because the output abstraction may result in fewer states of Q being distinguishable. Therefore M_s is specified as the prime machine of $(Q, \underline{q}, \Sigma_I, \Sigma_O^s, h_s)$.

By construction, two different states (q, q') in M_s produce outputs for certain input traces that differ in Σ_O^s. As a consequence, $q \not\sim_s q'$ holds. Therefore, the characterisation set of M_s equals the s-characterisation set W_s of M, as specified in Sect. 3.1. Analogously, the state identification sets of M_s are exactly the s-state identification sets of M. As a consequence, s-characterisation sets and s-state identification sets can be calculated by using the existing algorithms for characterisation sets and state identification sets, but the calculation needs to be performed on the abstracted FSM M_2.

Algorithm. With the FSM abstraction at hand, the algorithm for calculating the safety-complete test suite works as follows.

1. **Input 1.** Reference model $M = (Q, \underline{q}, \Sigma_I, \Sigma_O, h)$ with $|Q| = n$.
2. **Input 2.** Integer m satisfying $m \geq n$.
3. **Input 3.** Deterministic, completely specified, minimised FSM $M_s = (Q_s, \underline{q}_s, \Sigma_I, \Sigma_O^s, h_s)$ resulting from the abstraction of M with respect to $\leq_s$.
4. **Output.** Test suite TS which is safety-complete with respect to fault model $\mathcal{F} = (M, \sim_s, \mathcal{D}(m))$.
5. Calculate state cover V from M.
6. Calculate characterisation set W from M.
7. Calculate characterisation set W_s from M_s.
8. For all $q \in Q_s$, calculate state identification sets W_{s_q} from M_s.
9. Calculate $V.\Sigma_I^{m-n+1} \oplus \{W_{s_q} \mid q \in Q_s\}$ from M_s.
10. Set

$$\mathrm{TS} = V.W \cup \left(V. \bigcup_{i=0}^{m-n} \Sigma_I^i.W_s\right) \cup \left(V.\Sigma_I^{m-n+1} \oplus \{W_{s_q} \mid q \in Q_s\}\right)$$

11. Remove test cases from TS that are prefixes of longer test cases.
12. Return TS.

FSM Open Source Library. We have published the open source C++ library fsmlib-cpp[1] which contains all algorithms needed for implementing the algorithm above. This library also provides essential methods for minimising DFSMs

[1] https://github.com/agbs-uni-bremen/fsmlib-cpp.git.

and for making nondeterministic FSMs observable. Moreover, a generator main program is provided which uses these methods to calculated W-Method, Wp-Method, and Safety-complete Wp-Method test suites. An overview over this library is given in the lecture notes [9, Appendix B].

4 Case Studies and Strategy Evaluation

4.1 Control of Fasten Seat Belt and Return-to-Seat Signs in the Aircraft Cabin

Application. The following example is a (slightly simplified) real-world example concerning safety-related and uncritical indications in an aircraft cabin. A *cabin controller* in a modern aircraft switches the *fasten seat belt (FSB) signs* located above the passenger seats in the cabin and the *return to seat (RTS) signs* located in the lavatories according to the rules modelled in the DFSM shown in Table 1.

As inputs, the cabin controller reads the actual position of the fasten seat belts switch in the cockpit, which has the position **f0** (OFF), **f1** (ON), and **f2** (AUTO). Further inputs come from the cabin pressure control system which indicates "cabin pressure low" by event **d1** and "cabin pressure ok" by **d0**. This controller also indicates "excessive altitude" by **e1** or "altitude in admissible range" by **e0**. Another sub-component of the cabin controller determines whether the so-called AUTO condition is true (event **a1**) or false (**a0**).

The cabin controller switches the fasten seat belt signs and return to seat signs on and off, depending on the actual input change and its current internal state. As long as the cabin pressure and the cruising altitude are ok (after initialisation of the cabin controller or if last events from the cabin pressure controller were **d0, e0**), the status of the FSB and RTS signs is determined by the cockpit switch and the AUTO condition: if the switch is in the ON position, both FSB and RTS signs are switched on (output 11 in Table 1). Turning the switch into the OFF position switches the signs off. If the switch is in the AUTO position, both FSB and RTS signs are switched on if the AUTO condition becomes true with event **a1**, and they are switched off again after event **a0**. The AUTO condition may depend on the status of landing gears, slats, flaps, and oil pressure, these details are abstracted to **a1, a0** in our example.

As soon as there occurs a loss of pressure in the cabin (event **d1**) or an excessive altitude is reached, the FSB signs must be switched on and remain in this state, regardless of the actual state of the cockpit switch and the AUTO condition. The RTS signs, however, need to be switched off, because passengers should not be encouraged to leave the lavatories in a low pressure or excessive altitude situation.

After the cabin pressure and the altitude are back in the admissible range, the FSB and RTS signs shall automatically resume their state as determined by the "normal" inputs from cockpit switch and AUTO condition.

Table 1. State-transition table of DFSM specifying the control of FSB signs and RTS signs in an aircraft cabin.

	f0	**f1**	**f2**	**d1**	**d0**	**e1**	**e0**	**a1**	**a0**
s0	$s_0/00$	$s_1/11$	$s_2/00$	$s_3/10$	$s_0/00$	$s_6/10$	$s_0/00$	$s_{12}/00$	$s_0/00$
s1	$s_0/00$	$s_1/11$	$s_2/00$	$s_4/10$	$s_1/11$	$s_7/10$	$s_1/11$	$s_{13}/11$	$s_1/11$
s2	$s_0/00$	$s_1/11$	$s_2/00$	$s_5/10$	$s_2/00$	$s_8/10$	$s_2/00$	$s_{14}/11$	$s_2/00$
s3	$s_3/10$	$s_4/10$	$s_5/10$	$s_3/10$	$s_0/00$	$s_9/10$	$s_3/10$	$s_{15}/10$	$s_3/10$
s4	$s_3/10$	$s_4/10$	$s_5/10$	$s_4/10$	$s_1/11$	$s_{11}/10$	$s_4/10$	$s_{16}/10$	$s_4/10$
s5	$s_3/10$	$s_4/10$	$s_5/10$	$s_5/10$	$s_2/00$	$s_{11}/10$	$s_5/10$	$s_{17}/10$	$s_5/10$
s6	$s_6/10$	$s_7/10$	$s_8/10$	$s_9/10$	$s_6/10$	$s_6/10$	$s_0/00$	$s_{18}/10$	$s_6/10$
s7	$s_6/10$	$s_7/10$	$s_8/10$	$s_{10}/10$	$s_7/10$	$s_7/10$	$s_1/11$	$s_{19}/10$	$s_7/10$
s8	$s_6/10$	$s_7/10$	$s_8/10$	$s_{11}/10$	$s_8/10$	$s_8/10$	$s_2/00$	$s_{20}/10$	$s_8/10$
s9	$s_9/10$	$s_{10}/10$	$s_{11}/10$	$s_9/10$	$s_6/10$	$s_9/10$	$s_3/10$	$s_{21}/10$	$s_9/10$
s10	$s_9/10$	$s_{10}/10$	$s_{11}/10$	$s_{10}/10$	$s_7/10$	$s_{10}/10$	$s_4/10$	$s_{22}/10$	$s_{10}/10$
s11	$s_9/10$	$s_{10}/10$	$s_{11}/10$	$s_{11}/10$	$s_8/10$	$s_{11}/10$	$s_5/10$	$s_{23}/10$	$s_{11}/10$
s12	$s_{12}/00$	$s_{13}/11$	$s_{14}/11$	$s_{15}/10$	$s_{12}/00$	$s_{18}/10$	$s_{12}/00$	$s_{12}/00$	$s_0/00$
s13	$s_{12}/00$	$s_{13}/11$	$s_{14}/11$	$s_{16}/10$	$s_{13}/11$	$s_{19}/10$	$s_{13}/11$	$s_{13}/11$	$s_1/11$
s14	$s_{12}/00$	$s_{13}/11$	$s_{14}/11$	$s_{17}/10$	$s_{14}/11$	$s_{20}/10$	$s_{14}/11$	$s_{14}/11$	$s_2/00$
s15	$s_{15}/10$	$s_{16}/10$	$s_{17}/10$	$s_{15}/10$	$s_{12}/00$	$s_{21}/10$	$s_{15}/10$	$s_{15}/10$	$s_3/10$
s16	$s_{15}/10$	$s_{16}/10$	$s_{17}/10$	$s_{16}/10$	$s_{13}/11$	$s_{22}/10$	$s_{16}/10$	$s_{16}/10$	$s_4/10$
s17	$s_{15}/10$	$s_{16}/10$	$s_{17}/10$	$s_{17}/10$	$s_{14}/11$	$s_{23}/10$	$s_{17}/10$	$s_{17}/10$	$s_5/10$
s18	$s_{18}/10$	$s_{19}/10$	$s_{20}/10$	$s_{21}/10$	$s_{18}/10$	$s_{18}/10$	$s_{12}/00$	$s_{18}/10$	$s_6/10$
s19	$s_{18}/10$	$s_{19}/10$	$s_{20}/10$	$s_{22}/10$	$s_{19}/10$	$s_{19}/10$	$s_{13}/11$	$s_{19}/10$	$s_7/10$
s20	$s_{18}/10$	$s_{19}/10$	$s_{20}/10$	$s_{23}/10$	$s_{20}/10$	$s_{20}/10$	$s_{14}/11$	$s_{20}/10$	$s_8/10$
s21	$s_{21}/10$	$s_{22}/10$	$s_{23}/10$	$s_{21}/10$	$s_{18}/10$	$s_{21}/10$	$s_{15}/10$	$s_{21}/10$	$s_9/10$
s22	$s_{21}/10$	$s_{22}/10$	$s_{23}/10$	$s_{22}/10$	$s_{19}/10$	$s_{22}/10$	$s_{16}/10$	$s_{22}/10$	$s_{10}/10$
s23	$s_{21}/10$	$s_{22}/10$	$s_{23}/10$	$s_{23}/10$	$s_{20}/10$	$s_{23}/10$	$s_{17}/10$	$s_{23}/10$	$s_{11}/10$

First column defines the states (initial state s_0)
First row defines the inputs
Fields s/y denote 'Post-state/Output'

Inputs:
f0, f1, f2 : FSB switch in position OFF, ON, AUTO
d1, d0 : Cabin decompression true, false
e1, e0 : Excessive altitude true, false
a1, a0 : Auto condition true, false
Outputs:
00 denotes (FSB,RTS)=(0,0)
11 denotes (FSB,RTS)=(1,1)
10 denotes (FSB,RTS)=(1,0)

Safety Considerations. Analysing the outputs

$$(\text{FSB}, \text{RTS}) \in \Sigma_O = \{00, 10, 11, 01\}$$

from the safety-perspective, leads to the identification of one safety-critical output $(\text{FSB}, \text{RTS}) = (1, 0)$, which should be set whenever cabin decompression or excessive altitude occurs. If the other outputs $\{00, 11, 01\}$ are changed due to an application error, this is certainly undesirable, but does not represent a safety hazard. Note that the output combination 01 should never occur at all.

These considerations lead to an abstraction function

$$\begin{aligned} \alpha_s \ &: \ \Sigma_O \to \mathbb{P}(\Sigma_O) \\ 00 &\mapsto \{00, 10, 11, 01\} \\ 11 &\mapsto \{00, 10, 11, 01\} \\ 01 &\mapsto \{00, 10, 11, 01\} \\ 10 &\mapsto \{10\} \end{aligned}$$

as introduced in Sect. 3.3, and the abstracted FSM described there is obtained by replacing outputs 00, 11 by $\text{YY} = \{00, 10, 11, 01\}$, while leaving every occurrence of output 10 unchanged.

Comparison Wp-Method Versus Safety-Complete Wp-Method. The reference FSM with 24 states as specified in Table 1 is already minimal, and a characterisation set has 4 elements. The minimised version of the FSM abstraction only has 4 states and a characterisation set with 3 elements.

These figures motivate why the Wp-Method requires 549 test cases if the minimised machine representing the implementation has the same number of states as the reference model ($m = n$). The Safety-complete Wp-Method only requires 468 test cases in this situation; this corresponds to a test case reduction of approx. 15%.

4.2 Synthetic Example

Application. The following example does not come from a practical application, but has been constructed to illustrate that the reduction of test cases in comparison to the original Wp-Method can be quite significant. The reference state machine is shown in Table 2.

Safety Considerations. We assume that outputs 1 and 2 can be considered as non-critical, so that they can be abstracted to a single output Y. Output 0 is considered as critical.

Table 2. Example showing the effectiveness of the safety-complete Wp-Method

	a	**b**	**c**	**d**	**e**
s0	s1/1	s3/2	s2/0	s4/1	s5/1
s1	s1/1	s3/1	s2/0	s4/2	s5/1
s2	s1/1	s3/1	s2/0	s4/1	s5/1
s3	s1/2	s3/2	s1/0	s4/1	s5/1
s4	s1/2	s3/2	s2/0	s4/1	s5/1
s5	s1/0	s3/1	s0/0	s4/1	s6/1
s6	s1/0	s3/1	s2/0	s4/1	s5/1

First column defines the states (initial state s_0)
First row defines the inputs
Fields s/y denote 'Post-state/Output'

Comparison Wp-Method Versus Safety-Complete Wp-Method. The reference machine in Table 2 with its 7 states is already minimal, but the minimised abstracted FSM only has 2 states. As a consequence, both characterisation set and state identification sets of the abstracted machine are significantly smaller. Therefore, the Wp-Methods with assumption $m = n$ requires 87 test cases, while the Safety-complete Wp-Method only requires 41, this corresponds to a test case reduction of approx. 53%.

While this example is of no practical value, it illustrates effectively that test case reductions to less than half of the cases required for the Wp-Method are possible when using the Safety-complete Wp-Method.

4.3 Garage Door Controller

Application. This example has been originally proposed in [6]. We use it here as a negative example: it is not guaranteed that the Safety-complete Wp-Method will always require fewer test cases than the Wp-Method, though the former has lesser test strength than the latter. Therefore it is important to compare the required test case numbers beforehand – for example, by using the algorithms made available in the FSM Library described in [9, Appendix B] – before deciding which test suites to run against the system under test.

The garage door controller uses inputs from a remote control, two sensors indicating whether the door has reached the up position or the down position, respectively, and a light sensor indicating whether the door area is crossed while the door is closing or opening. The controller commands the motor to go down, up, stop, or to reverse the down direction to the up direction. Its detailed behaviour is specified in Table 3.

Safety Considerations. The only output considered as safety-critical is the command to reverse the down-direction to the up direction. All other outputs can be abstracted to some value Y.

Comparison Wp-Method Versus Safety-Complete Wp-Method. Both the reference model in Table 3 and its abstraction are not minimal. It turns out, however, that the minimised abstracted model still has as many states as the minimised reference model. Moreover, the characterisation set and the state identification sets of the abstracted model are larger than the equivalent sets derived from the minimised reference model.

As a consequence, the Wp-Method requires only 17 test cases for $m = n$, while the Safety-complete Wp-Method requires 33 cases.

Table 3. DFSM modelling the garage door controller.

	e1	**e2**	**e3**	**e4**
DU	DC/a1	DU/a3	DU/a3	DU/a3
DD	DO/a2	DD/a3	DD/a3	DD/a3
DSD	DC/a1	DSD/a3	DSD/a3	DSD/a3
DSU	DO/a2	DSU/a3	DSU/a3	DSU/a3
DC	DSD/a3	DD/a3	DC/a1	DO/a4
DO	DSU/a3	DO/a2	DU/a3	DO/a2

Inputs:
e1 : Remote control has been pressed
e2 : Sensor indicates “door reaches down position”
e3 : Sensor indicates “door reaches up position”
e4 : Sensor indicates “light beam crossed”
Outputs:
a1 : Command “start down movement” to motor
a2 : Command “start up movement” to motor
a3 : Command “stop movement” to motor
a4 : Command “reverse down movement to up” to motor
States:
DU : Door is in up position
DD : Door is in down position
DSD : Door is stopped while going down
DSU : Door is stopped while going up
DC : Door is closing
DO : Door is opening

5 Conclusion

We have presented a testing strategy which guarantees to uncover every safety violation when testing an implementation against a deterministic finite state machine reference model. These guarantees hold under the assumption that the true behaviour of the implementation, when expressed by a minimised state machine, does not exceed a certain maximum of states m, in comparison to the number n of states in the minimised reference model. Safety criticality has been modelled by means of a safety-related output abstraction which allows to express that certain outputs can be exchanged by certain others without introducing a safety violation. The new strategy has been derived from the well-known Wp-Method. A proof has been presented which shows that – while no longer guaranteeing to uncover *every* violation of input/output equivalence – the strategy will uncover every failure which ends in an erroneous output representing a safety violation.

First experiments have shown that this Safety-complete Wp-Method may require significantly fewer test cases than the Wp-Method (reductions between 15% and 50% have been observed). It has been indicated by another example, however, that this reduction is not guaranteed.

The concept described here can be extended to more complex systems whose behaviour can be represented by a certain class of Kripke structures over infinite input domains, but with finite domains for internal states and outputs. It has been shown in [5] that a specific input equivalence class construction technique can be applied, so that any complete testing theory valid for FSMs can be translated to a likewise complete equivalence class partition testing strategy for these systems with Kripke semantics.

References

1. Chow, T.S.: Testing software design modeled by finite-state machines. IEEE Trans. Software Eng. **4**(3), SE-178–186 (1978)
2. Fujiwara, S., Bochmann, G.V., Khendek, F., Amalou, M., Ghedamsi, A.: Test selection based on finite state models. IEEE Trans. Software Eng. **17**(6), 591–603 (1991)
3. Haxthausen, A.E., Peleska, J.: Formal development and verification of a distributed railway control system. IEEE Trans. Softw. Eng. **26**(8), 687–701 (2000)
4. Huang, W., Peleska, J.: Complete model-based equivalence class testing. STTT **18**(3), 265–283 (2016). http://dx.doi.org/10.1007/s10009-014-0356-8
5. Huang, W.I., Peleska, J.: Complete model-based equivalence class testing for nondeterministic systems. Formal Aspects Comput. **29**(2), 335–364 (2017). http://dx.doi.org/10.1007/s00165-016-0402-2
6. Jorgensen, P.C.: The Craft of Model-Based Testing. CRC Press, Boca Raton (2017)
7. Li, S., Qi, Z.: Property-oriented testing: An approach to focusing testing efforts on behaviours of interest. In: SOQUA/TECOS (2004)
8. Luo, G., von Bochmann, G., Petrenko, A.: Test selection based on communicating nondeterministic finite-state machines using a generalized wp-method. IEEE Trans. Software Eng. **20**(2), 149–162 (1994). http://doi.ieeecomputersociety.org/10.1109/32.265636

9. Peleska, J., Huang, W.l.: Test Automation - Foundations and Applications of Model-based Testing. University of Bremen (2017). Lecture notes http://www.informatik.uni-bremen.de/agbs/jp/papers/test-automation-huang-peleska.pdf
10. Vasilevskii, M.P.: Failure diagnosis of automata. Kibernetika (Transl.) **4**, 98–108 (1973)