FSM quasi-equivalence testing via reduction and observing absences

Robert M. Hierons

Department of Computer Science, The University of Sheffield, UK

Abstract

There has been significant interest in automatically generating test cases from a non-deterministic finite state machine (FSM). Most approaches check that the behaviours of the system under test (SUT) are allowed by the specification FSM; they therefore test for *reduction*. However, sometimes one wants all of the behaviours, and so features, of the specification to be implemented and then one is testing for *equivalence*. In this paper we first note that in order to test for equivalence one must effectively be able to observe the SUT not being able to produce an output y in response to an input x after trace $\bar{\sigma}$; we model this as the absence of an output. We prove that the problem of testing for equivalence to FSM M can be mapped to testing for reduction to an FSM $\mathcal{R}(M)$ that extends M with absences. Thus, one can use techniques developed for testing for reduction when testing for equivalence. We then consider the case where the specification is partial, generalising the result to quasi-equivalence. These results are proved for observable specifications and so we also show how a partial FSM can be mapped to an observable partial FSM from which we can test.

Keywords: conformance testing, partial finite state machine, testing for equivalence, observable finite state machine.

1. Introduction

There has been significant interest in the problem of automatically generating a test suite from a finite state machine (FSM) specification M, with work considering deterministic FSMs [1, 2, 3, 4, 5, 6], non-deterministic FSMs [7, 8, 9, 10, 11], and partial (deterministic or non-deterministic) FSMs [12, 13, 14, 15, 16, 17]. Much of the FSM testing literature assumes that FSM specifications are completely-specified, where FSM M is completely-specified if for every state s and input x, the response to x in state s is defined. However, in practice specifications often are not completely-specified, they are partial (see, for example, [18, 19, 16]). For example, Kushik et al. [20] note that protocol

Email address: r.hierons@sheffield.ac.uk (Robert M. Hierons)

specifications are usually partial and also observe that often it is not possible to complete the corresponding FSMs used as the basis of testing. In addition, work on testing from partial FSMs looked at a set of 59 FSMs based on the designs of circuits (the ACM/SIGDA benchmarks [21]) and found that just over 25% of these FSMs from industry were partial [22].

In order to reason about test effectiveness, it is normal to assume that the *system under test (SUT)* behaves like an *unknown* FSM $\mathcal{I}$. It is then possible to define an *implementation relation* (between FSMs) that says what it means for an unknown FSM model $\mathcal{I}$ of the SUT to be a correct implementation of the FSM specification $\mathcal{S}$. The implementation relation tells us what behaviours are allowed, for a given specification $\mathcal{S}$, and so allows us to determine the verdict (pass or fail) when we test the SUT. In addition, if we assume that the FSM $\mathcal{I}$ must lie in some given set $\mathcal{F}$ (a fault domain), then there is potential to produce test suites that are complete in the following sense: if the SUT passes the test suite then, under the assumption that $\mathcal{I} \in \mathcal{F}$, we know that the SUT is a correct implementation of $\mathcal{S}$. The notion of a fault domain is closely related to the test hypotheses introduced by Gaudel [23].

There are two standard implementation relations for testing from a completely-specified FSM specification $\mathcal{S}$, reduction and equivalence. Given specification $\mathcal{S}$, that is a completely-specified FSM, the implementation $\mathcal{I}$ is a *reduction* of $\mathcal{S}$ if all sequences of input/output pairs (*traces*) of $\mathcal{I}$ are also traces of $\mathcal{S}$. This corresponds to *language inclusion*: the set of traces of $\mathcal{I}$ is a subset of the set of traces of $\mathcal{S}$. The notion of *equivalence* strengthens this to require that $\mathcal{I}$ and $\mathcal{S}$ have the same set of traces and so corresponds to *language equivalence.* Most work has looked at the problem of testing for reduction but sometimes one might want the SUT to be behaviourally equivalent to the specification FSM M. This might be the case, for example, when one wishes to ensure that all of the features of the specification are present.

Naturally, one needs to adapt implementation relations to the case where the specification is partial. Work on testing from a partial FSM specification $\mathcal{S}$ normally assumes that if no response to input x is defined after trace $\bar{\sigma}$ then all behaviours are allowed. As result, there are two implementation relations for partial FSMs: *quasi-equivalence* and *quasi-reduction*, and some work on testing from partial FSMs [15, 16, 17, 24, 25]. However, until recently these implementation relations were only defined for specifications that have *harmonised traces*[1].

The work in this paper builds on recent research that generalises the implementation relations for partial FSMs (quasi-reduction and quasi-equivalence) to remove the requirement that the specification has harmonised traces. This previous work showed that one can test for quasi-reduction to specification $\mathcal{S}$ by completing $\mathcal{S}$ in an appropriate manner and then using any technique for testing for reduction [26]. The practical benefit of this is that when testing

[1]FSM $\mathcal{S}$ has harmonised traces if there is no input sequence $\bar{x}$ that can take $\mathcal{S}$ to states s and s' such that there is an input x defined in s but not s'.

whether the SUT is a quasi-reduction of the partial FSM specification $\mathcal{S}$, one can use any one of the many test generation algorithms that have been devised for testing whether an SUT is a reduction of a completely-specified FSM [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]. However, the work also showed that the approach given cannot be used with quasi-equivalence. The work in this paper was motivated by the desire to map the problem of testing for quasi-equivalence to one where there was a more substantial body of techniques (e.g. testing for reduction or equivalence when the specification is completely-specified).

This paper starts by considering how one can test for equivalence to a completely-specified FSM specification $\mathcal{S}$; this is a problem that has been considered before [27] but we take a rather different approach. In order to be able to test for equivalence, one must be able to test in a manner that allows one to conclude that a given trace $\bar{\sigma}$ of $\mathcal{S}$ has *not* been implemented by the SUT $\mathcal{I}$; the failure to implement $\bar{\sigma}$ is one of the ways in which the SUT can fail. Typically, this is achieved by making a fairness assumption: for a given integer k, if one applies a test case t a total of k times then one is (assumed to be) guaranteed to observe all possible traces of the SUT that can result from t. The choice of a value for k provides a cost/effectiveness trade-off and might be based on domain knowledge.

Observing that $\bar{\sigma}$ is not a trace of the SUT is similar to observing a *refusal*, which typically corresponds to the ability to observe that the SUT cannot engage in an action a (or a set of actions) [28]. We instead use the term 'absence' since, as we explain later, typically it does not make sense to refuse an output in testing (see, for example, [29]). For each output y we add a_y to the output alphabet, with a_y representing the situation in which the FSM cannot produce output y in response to a given input. We show how we can add such observations to an FSM M to form an FSM $\mathcal{R}(M)$.

Having introduced the notion of the absence of an output, we prove that it is possible to rephrase testing for equivalence to $\mathcal{S}$ in terms of testing that the SUT is a reduction of the completely-specified FSM $\mathcal{R}(\mathcal{S})$. As a result, we can use any technique for testing for reduction when testing for equivalence. Having established this result, we consider the problem of testing from a partial FSM $\mathcal{S}$. We show that one can test for quasi-equivalence to $\mathcal{S}$ by testing that the SUT is a quasi-reduction of $\mathcal{R}(\mathcal{S})$. It is known that one can use any one of the many techniques devised for testing for reduction against a completely-specified FSM when testing for quasi-reduction [26]; the combination of these results tells us that we can use any technique for testing for reduction in order to test for quasi-equivalence.

These result are proved for the case where the specification $\mathcal{S}$ is observable[2]. This is not a significant restriction if $\mathcal{S}$ is completely-specified since one can map a completely-specified FSM to an equivalent observable completely-specified FSM using any technique that maps a non-deterministic finite state

[2]A finite state machine $\mathcal{S}$ is observable if whenever $\bar{\sigma}$ is a trace of $\mathcal{S}$, there is only one state of $\mathcal{S}$ that can be reached by a path with label $\bar{\sigma}$.

automaton to an equivalent deterministic finite state automaton. However, it is known that such techniques cannot in general be used with partial FSMs[3]. We therefore consider the problem of mapping a partial FSM $\mathcal{S}$ that is not observable to an observable partial FSM $\mathcal{S}'$ from which one can test. We start by examining the recently defined notions of correctness (generalisations of quasi-reduction and quasi-equivalence) and show that these can be slightly weakened when the specification is not observable. We then devise an approach that adds events that represent the refusal of *inputs*, then makes the resultant FSM observable, and finally removes certain transitions.

This paper makes the following main contributions.

1. It proves that testing for equivalence/quasi-equivalence can be rephrased as testing for reduction.
2. It shows that the recent generalised definitions of quasi-equivalence and quasi-reduction can be slightly weakened when the specification is not observable.
3. It solves the problem of how one can map a partial FSM $\mathcal{S}$ that is not observable to an observable partial FSM $\mathcal{S}'$ from which one can test.

The primary practical consequence is that, when testing whether the SUT is quasi-equivalent to partial FSM specification $\mathcal{S}$, one can generate a completely-specified FSM $\mathcal{S}_1$ and use any one of the many automated test generation techniques that generate test suites from a completely-specified FSM [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

The paper is structured as follows. Section 2 defines associated notation and concepts. In Section 3 we prove the results for completely-specified FSMs and in Section 4 we generalise this to partial FSMs. Section 5 then shows how we can map a partial FSM to an observable partial FSM from which we can test. Practical implications are discussed in Section 6 and Section 7 reviews some related work. Finally, in Section 8 we summarise the results and discuss possible lines of future work.

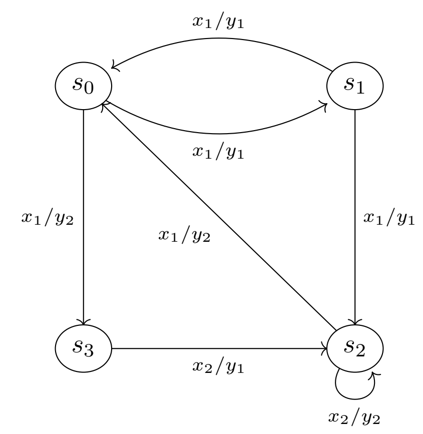
2. Preliminaries

This section defines finite state machines (*Mealy machines*) and the associate concepts and notation used in the paper.

Definition 1. *A partial non-deterministic finite state machine (PFSM) is defined by a tuple* (S, s_0, X, Y, h) *in which:* S *is the finite set of states;* $s_0 \in S$ *is the initial state;* X *is the finite input alphabet;* Y *is the finite output alphabet; and* h *is the transition relation of type* $S \times X \times S \times Y$.

Throughout this paper M will denote such a PFSM and we will fix the input alphabet to be X and the output alphabet to be Y unless otherwise

[3]In Section 5 we given an example that demonstrates this.



$$\begin{tikzcd}
s_0 \arrow[r, bend right, "x_1/y_1"'] \arrow[d, "x_1/y_2"'] & s_1 \arrow[l, bend right, "x_1/y_1"'] \arrow[d, "x_1/y_1"] \\
s_3 \arrow[r, "x_2/y_1"'] & s_2 \arrow[ul, "x_1/y_2"] \arrow[loop below, "x_2/y_2"]
\end{tikzcd}$$

Figure 1: Partial FSM $\mathcal{S}_1$

stated. Figure 1 gives an example of a PFSM $\mathcal{S}_1$ (from [26]) with input alphabet $\{x_1, x_2\}$ and output alphabet $\{y_1, y_2\}$. Here, nodes represent states and an arc from s_i to s_j with label x/y denotes $(s_i, x, s_j, y) \in h$; the PFSM can move from s_i to s_j with input x and output y. We can associate a PFSM with its initial state and so at times we will use the terms PFSM and state interchangeably.

An input x is *defined* in state s of M if there is at least one state s' and output y such that $(s, x, s', y) \in h$. Further, we let *dom h* be the set of (s, x) such that x is defined in state s. Given $(s, x) \in \textit{dom } h$, we let $h(s, x) = \{(s', y)|(s, x, s', y) \in h\}$. We also use $\Omega_M(s)$ to denote the set of inputs that are defined in state $s \in S$ and so $\Omega_M(s) = \{x \in X|(s, x) \in \textit{dom } h\}$. If s is the initial state of M then we also denote this $\Omega(M)$ (i.e. $\Omega(M) = \Omega_M(s_0)$). If we consider $\mathcal{S}_1$, we have that $\Omega_{\mathcal{S}_1}(s_2) = \{x_1, x_2\}$ and $\Omega(\mathcal{S}_1) = \{x_1\}$. Given $(s', y) \in h(s, x)$, $t = (s, s', x/y)$ is a *transition* of M. For example, $(s_1, s_2, x_1/y_1)$ is a transition of $\mathcal{S}_1$. Given transition $t = (s, s', x/y)$, s is the *starting state* of t, s' is the *ending state* of t, and x/y is the *label* of t.

Given PFSM M, if $|h(s, x)| \geq 1$ for all $s \in S$ and $x \in X$ then M is a *completely-specified* finite state machine (*FSM*). We use the term PFSM where a finite state machine might be partial or completely-specified and use FSM when the finite state machine must be completely-specified. Clearly $\mathcal{S}_1$ is partial since, for example, there is no transition from s_1 with input x_2. PFSM M is said to be *observable* if there is no pair of transitions that have the same starting state, input and output but different ending states (for all $(s, x) \in \textit{dom } h$ and $y \in Y$ there is at most one s' for which we have that $(s', y) \in h(s, x)$). It is straightforward to see that $\mathcal{S}_1$ is not observable since, for example, from state s_1 there are two transitions with label x_1/y_1 and these reach different states.

In this paper we assume that the specification is a PFSM $\mathcal{S}$. When testing from such a PFSM, it is normal to assume that the SUT behaves like an *unknown* completely-specified FSM $\mathcal{I}$ with the same input and output alphabets as $\mathcal{S}$. It is then possible to formally define what it means for the SUT to be a correct implementation of $\mathcal{S}$ in terms of an implementation relation between FSMs. The assumption that the SUT is completely-specified is one normally made in the testing literature since most systems will either respond to an input, when applied, or provide observable evidence that the input cannot be applied (this evidence can be represented by an output).

The execution of PFSM M leads to a *path*; a sequence $\bar{\rho} = (s_1, s_2, x_1/y_1)\dots(s_k, s_{k+1}, x_k/y_k)$ of consecutive transitions. We say that s_1 is the *starting state* of $\bar{\rho}$, s_{k+1} is the *ending state* of $\bar{\rho}$, and $\bar{\sigma} = x_1/y_1 \dots x_k/y_k$ is the *label* of $\bar{\rho}$. Here, $\bar{\sigma}$ is an input/output sequence, also called a *trace*, with *input portion* $\pi_1(\bar{\sigma}) = x_1 \dots x_k$ and *output portion* $\pi_2(\bar{\sigma}) = y_1 \dots y_k$. Thus, for example, $(s_0, s_1, x_1/y_1)(s_1, s_2, x_1/y_1)$ is a path of $\mathcal{S}_1$ that has starting state s_0, ending state s_2, and label $x_1/y_1\ x_1/y_1$. Given PFSM M, $L_M(s)$ denotes the set of labels of paths of M that start at s; if s is the initial state of M then we sometimes use $L(M)$ instead. Naturally, $L_M(s)$ is prefix closed: if $\bar{\sigma} \in L(M)$ then all prefixes of $\bar{\sigma}$ are in $L(M)$.

We will need to reason about the possible behaviours of a PFSM after a trace $\bar{\sigma}$ has occurred. If s is a state of PFSM M, s-after-$\bar{\sigma}$ will be the set of states of M that can be reached from s by a path with trace $\bar{\sigma}$. Thus, $s' \in s$-after-$\bar{\sigma}$ if and only if M has a path that has starting state s, ending state s' and label $\bar{\sigma}$. In $\mathcal{S}_1$, for example, $(\mathcal{S}_1\text{-after-}x_1/y_1\ x_1/y_1) = \{s_0, s_2\}$. Naturally, if M is observable then s-after-$\bar{\sigma}$ is either empty or a singleton.

We extend h to input sequences: $h(s, \epsilon) = \{(s, \epsilon)\}$ and for $x \in X$ and $\bar{x} \in X^*$ we have that $h(s, x\bar{x}) = \{(s'', y\bar{y}) | \exists s' \in S.(s', y) \in h(s, x) \wedge (s'', \bar{y}) \in h(s', \bar{x})\}$.

3. Completely-specified FSMs

In this section we consider the problem of testing for equivalence to an FSM $\mathcal{S}$, showing that this can be mapped to the problem of testing for reduction to an FSM that can be constructed from $\mathcal{S}$. The practical benefit is that we can then use any test generation algorithm for reduction when testing for equivalence.

When testing for equivalence, we are testing to determine whether $L(\mathcal{S}) = L(\mathcal{I})$. Given set L of traces, we let $\bar{L}$ denote the set of traces not in L. Clearly, $L(\mathcal{S}) = L(\mathcal{I})$ if and only if $L(\mathcal{I}) \subseteq L(\mathcal{S})$ and $\bar{L}(\mathcal{I}) \subseteq \bar{L}(\mathcal{S})$, a fact that will be used in proofs.

The challenge is to show how the second of these two problems (testing that $\bar{L}(\mathcal{I}) \subseteq \bar{L}(\mathcal{S})$) can be expressed in terms of testing from an FSM; here we utilise absences.

As noted earlier, the behaviour of a PFSM is defined in terms of traces. We let $\Gamma = \{x/y | x \in X \wedge y \in Y\}$ denote the set of input/output pairs. Given an output y, we will use $a_y \notin Y$ to denote the observation of the *absence* of y in response to an input and we let $Y_a = \{a_y | y \in Y\}$. We therefore let $\Gamma_a = \{x/a_y | x \in X \wedge y \in Y\}$ denote the set of input/output pairs that denote absences. Given a sequence $\bar{\gamma} \in (\Gamma \cup \Gamma_a)^*$ (one that might include absences), at times we will want to be able to reason about the corresponding trace in Γ^*.

Definition 2. *Given $\bar{\gamma} \in (\Gamma \cup \Gamma_a)^*$, $to\_out(\bar{\gamma})$ will denote the trace in Γ^* formed by replacing every a_y by the corresponding output y. Thus, given $x \in X$ and $y \in Y$:*

$$to\_out(\bar{\gamma}) = \begin{cases} \epsilon & \text{if } \bar{\gamma} = \epsilon \\ x/y\ to\_out(\bar{\gamma_1}) & \text{if } \bar{\gamma} = x/y\ \bar{\gamma_1} \\ x/y\ to\_out(\bar{\gamma_1}) & \text{if } \bar{\gamma} = x/a_y\ \bar{\gamma_1} \end{cases}$$

Given FSM M, FSM $\mathcal{R}(M)$ will include absences to model traces that are not traces of M. The following is defined for partial FSMs since we will reuse the definition in the next section.

Definition 3. *Given observable PFSM $M = (S, s_0, X, Y, h)$, $\mathcal{R}(M)$ is the FSM $(S \cup \{s_R\}, s_0, X, Y \cup Y_a, h')$ in which $s_R \notin S$ and h' is defined by the following in which $s \in S$, $x \in X$, $y \in Y$, with $(s, x) \in dom\ h$.*

1. *If $y \in \pi_2(h(s, x))$ then $\{s'|(s', y) \in h(s, x)\} = \{s'|(s', y) \in h'(s, x)\}$ and $\{s'|(s', a_y) \in h'(s, x)\} = \emptyset$. Comment: this says that we retain the transitions of M.*
2. *If $y \notin \pi_2(h(s, x))$ then $\{s'|(s', a_y) \in h'(s, x)\} = \{s_R\}$ and $\{s'|(s', y) \in h'(s, x)\} = \emptyset$. Comment: this says that if M should not produce output y when it receives input x in state s then this is still the case in $\mathcal{R}(M)$ and, in addition, we add a transition from s with input x and absence a_y, with this going to the new sink state S_R.*
3. *We have that $h'(s_R, x) = \{(s_R, a_y)|y \in Y\}$. Comment: this says that in the new sink state we can only observe absences.*

Essentially, $\mathcal{R}(M)$ can be constructed from M by adding a new state s_R, in which all outputs are refused, and an input/output pair x/a_y from a state s to s_R if y cannot be produced in response to x from state s. The condition requiring that $(s, x) \in dom\ h$ is included for the case where the FSM M is partial: we do not want to add transitions corresponding to inputs that are not specified. Figure 2 gives an example of an FSM $\mathcal{S}_2$ and Figure 3 gives $\mathcal{R}(\mathcal{S}_2)$. In Figure 2, for example, an arc with label $x_1/y_2, x_2/y_1$ denotes two transitions: one with label x_1/y_2 and one with label x_2/y_1. Note that if M is a PFSM then $\mathcal{R}(M)$ is also partial (a case we consider in the next section).

We will now prove that in order to test for equivalence to $\mathcal{S}$ it is sufficient to test that $\mathcal{R}(\mathcal{I})$ is a reduction of $\mathcal{R}(\mathcal{S})$. The intuition is that given FSM M, $\mathcal{R}(M)$ captures both the traces that M can produce (the traces of $\mathcal{R}(M)$ that contain no absences) and also the traces that M cannot produce (the traces of $\mathcal{R}(M)$ that contain no absences). The following Lemmas give corresponding properties that are used in the proof of the main result; any proofs can be found in the appendix.

The first Lemma states that the traces in $L(\mathcal{R}(M))$ that are not traces in $L(M)$ end in absences; this is immediate from the definition of $\mathcal{R}(M)$.

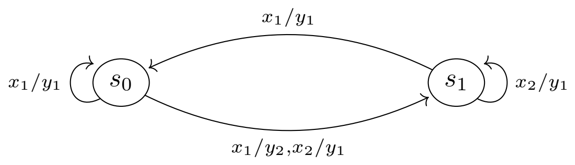


Figure 2: Completely-specified FSM $\mathcal{S}_2$

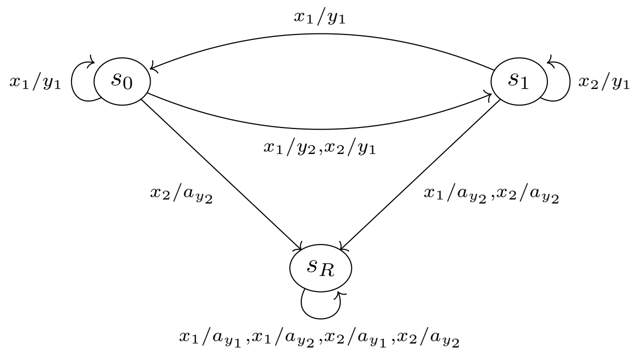

x_1/y_1
x_1/y_1
s_0
s_1
x_2/y_1
x_1/y_2,x_2/y_1
x_2/a_{y_2}
x_1/a_{y_2},x_2/a_{y_2}
s_R
x_1/a_{y_1},x_1/a_{y_2},x_2/a_{y_1},x_2/a_{y_2}

Figure 3: FSM $\mathcal{R}(\mathcal{S}_2)$

Lemma 1. *Given observable FSM M, all traces in $L(\mathcal{R}(M)) \setminus L(M)$ are in the form of $\bar{\sigma}_1\bar{\sigma}_2$ for $\bar{\sigma}_1 \in L(M)$ and non-empty $\bar{\sigma}_2 \in \Gamma_a^*$.*

The following is stated (and proved in the Appendix) for PFSMs since we also use the result in the next section. It tells us that the set of traces in $L(\mathcal{R}(M))$, that are not traces in $L(M)$, represents the set of traces of $\bar{L}(M)$.

Lemma 2. *Given observable PFSM M, $\bar{L}(M) = to\_out(L(\mathcal{R}(M)) \setminus L(M))$.*

The following is an immediate consequence of these two lemmas and the fact that $L(\mathcal{I}) = L(\mathcal{S})$ if and only if $L(\mathcal{I}) \subseteq L(\mathcal{S})$ and $\bar{L}(\mathcal{I}) \subseteq \bar{L}(\mathcal{S})$.

Lemma 3. *Given FSMs $\mathcal{S}$ and $\mathcal{I}$, $L(\mathcal{I}) = L(\mathcal{S})$ if and only if $L(\mathcal{R}(\mathcal{I})) = L(\mathcal{R}(\mathcal{S}))$.*

We can now give this section's main result, which follows from the above Lemma and the fact that, by construction, $L(\mathcal{R}(\mathcal{I})) = L(\mathcal{R}(\mathcal{S}))$ if and only if $L(\mathcal{R}(\mathcal{I})) \subseteq L(\mathcal{R}(\mathcal{S}))$.

Theorem 1. *Given observable FSMs $\mathcal{I}$ and $\mathcal{S}$ with the same input and output alphabets, $\mathcal{S}$ and $\mathcal{I}$ are equivalent if and only if $L(\mathcal{R}(\mathcal{I})) \subseteq L(\mathcal{R}(\mathcal{S}))$.*

This result tells us that if we want to test whether the SUT is equivalent to $\mathcal{S}$, it is sufficient to test whether the SUT (with absences observed in testing) is a reduction of $\mathcal{R}(\mathcal{S})$. Most likely, absences will be observed using the standard fairness assumption and, as noted earlier, it makes no sense to test for equivalence if one cannot observe absences; there is no observation that can be made in testing that leads to the SUT failing a test if the SUT is a reduction of $\mathcal{S}$ but is not equivalent to $\mathcal{S}$.

4. Partial FSMs

In this section we consider the problem of testing from an observable PFSM. We start by giving the definitions of the implementation relations (quasi-equivalence and quasi-reduction) we require for testing from PFSMs. We then define the set $U_{\mathcal{S}}$ of traces that 'do not matter' given specification $\mathcal{S}$; these are the traces that begin with a trace of $\mathcal{S}$ followed by an undefined input. Having defined $U_{\mathcal{S}}$, we show how quasi-equivalence and quasi-reduction can be characterised in terms of $L(\mathcal{S})$ and $L(\mathcal{I}) \setminus U_{\mathcal{S}}$. We then use this to prove that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $\mathcal{R}(\mathcal{I})$ is a quasi-reduction of $\mathcal{R}(\mathcal{S})$. This allows one to use any technique, for testing for quasi-reduction, when testing for quasi-equivalence. Further, it has previously been shown that, when testing for quasi-reduction, we can use any technique for testing for reduction [26]. Thus, the results in this section tell us that when testing whether the SUT is quasi-equivalent to the specification it is possible to use any one of the many techniques that have been devised for testing whether the SUT is a reduction of the specification [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

Recall that if input x is not specified in state s then all possible behaviours of an SUT are allowed if x is received when the specification is in state s (see, for example, [15, 16, 17, 24, 25]). The implementation relation quasi-equivalence has been defined to capture this requirement; it essentially relaxes equivalence to allow the SUT to have any behaviour in response to undefined inputs.

Definition 4. *Given states s_1 and s_2 of observable PFSM M, s_1 is quasi-equivalent to state s_2 if and only if, for all $\bar{\sigma} \in L_M(s_1) \cap L_M(s_2)$ and $x \in X$, if $x \in \Omega_M(s_2\text{-}after\text{-}\bar{\sigma})$ then:*

1. *$x \in \Omega_M(s_1\text{-}after\text{-}\bar{\sigma})$; and*
2. *$\{y' \in Y | \bar{\sigma}x/y' \in L_M(s_1)\} = \{y' \in Y | \bar{\sigma}x/y' \in L_M(s_2)\}$.*

Given observable PFSMs $\mathcal{I}$ and $\mathcal{S}$ with the same input and output alphabets, we say that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if the initial state of $\mathcal{I}$ is quasi-equivalent to the initial state of $\mathcal{S}$. We denote this $\mathcal{I} \approx_Q \mathcal{S}$.

It is clear that the quasi-equivalence relation is reflexive (every observable FSM is quasi-equivalent to itself) and it is not too hard to show that it is transitive. However, quasi-equivalence is not symmetric since, for example, every observable FSM is quasi-equivalent to the FSM that has no transitions but the converse does not hold.

We also define quasi-reduction since we will express quasi-equivalence in terms of quasi-reduction.

Definition 5. *Given states s_1 and s_2 of observable PFSM M, s_1 is a quasi-reduction of state s_2 if and only if, for all $\bar{\sigma} \in L_M(s_1) \cap L_M(s_2)$ and $x \in X$, if $x \in \Omega_M(s_2\text{-}after\text{-}\bar{\sigma})$ then:*

1. *$x \in \Omega_M(s_1\text{-}after\text{-}\bar{\sigma})$; and*
2. *$\{y' \in Y | \bar{\sigma}x/y' \in L_M(s_1)\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L_M(s_2)\}$.*

Similar to quasi-equivalence, quasi-reduction is reflexive and transitive but not symmetric.

We now explore how quasi-equivalence can be represented in terms of inclusion. First, we define the set of pairs $(\bar{\sigma}, x)$ such that input x is defined after trace $\bar{\sigma}$.

Definition 6. *Given observable PFSM M, D_M is the set of $(\bar{\sigma}, x)$ such that $\bar{\sigma} \in L(M)$ and input x is defined in the unique state s of M reached by $\bar{\sigma}$ ($s \in s_0\text{-}after\text{-}\bar{\sigma}$).*

We let U_M denote the set of traces associated with undefined inputs.

Definition 7. *Given observable PFSM M,*

$$U_M = \{\bar{\sigma}_1 x/y \bar{\sigma}_2 \in \Gamma^* | \bar{\sigma}_1 \in L(M) \wedge (\bar{\sigma}_1, x) \notin D_M\}$$

Clearly, U_M is a regular language and it is straightforward to construct a finite automaton that accepts U_M.

We can now formulate quasi-equivalence directly in terms of regular languages. The following essentially says that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if the two PFSMs 'agree' (have the same possible outputs) whenever we apply an input x after $\bar{\sigma}$ such that $(x, \bar{\sigma}) \in D_{\mathcal{S}}$. The essential idea is that $L(\mathcal{I}) \setminus U_{\mathcal{S}}$ captures all of the traces of $\mathcal{I}$ that are referred to in the definition of quasi-equivalence. Again, proofs of lemmas can be found in the appendix.

Lemma 4. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$ with the same input and output alphabets, $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $L(\mathcal{I}) \setminus U_{\mathcal{S}} = L(\mathcal{S})$.*

We can now express quasi-equivalence in terms of language inclusion, the next result following from properties of sets and Lemma 4.

Lemma 5. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$ with the same input and output alphabets, $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $(L(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq L(\mathcal{S})$ and $(\bar{L}(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq (\bar{L}(\mathcal{S}) \setminus U_{\mathcal{S}})$.*

Similar to Lemma 4, we can represent quasi-reduction in terms of language inclusion.

Lemma 6. *Let us suppose that $\mathcal{S}$ is an observable PFSM and $\mathcal{I}$ is an observable FSM with the same input and output alphabets. Then, $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$ if and only if $L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S})$.*

Given a PFSM M, at times we will reason about both U_M and $U_{\mathcal{R}(M)}$. In $\mathcal{R}(M)$, if we take a transition that includes the absence of an output then we go to the state in which all inputs are enabled. Thus, from the definition of U_M, if a trace $\bar{\sigma}$ is in $U_{\mathcal{R}(M)}$ then it must start with a trace $\bar{\sigma}_1$ from $L(M)$ followed by an 'undefined input' x ($(\bar{\sigma}_1, x) \notin D_M$) with either an output or the absence of an output. After this, all traces from $(\Gamma \cup \Gamma_a)^*$ are possible. We therefore have the following result, which is immediate.

Lemma 7. *Given observable PFSM M, we have that*

$$\begin{gathered} U_{\mathcal{R}(M)} = \\ \{\bar{\sigma}_1 x/z \bar{\sigma}_2 | \bar{\sigma}_1 \in L(M) \wedge \\ \bar{\sigma}_2 \in (\Gamma \cup \Gamma_a)^* \wedge (\bar{\sigma}_1, x) \notin D_M \wedge z \in Y \cup Y_a\} \end{gathered}$$

We now provide two more results that we require, before proving the main result in this section. These results will allow us to move between reasoning about traces in $(\Gamma \cup \Gamma_a)^*$ that contain absences (i.e. traces of $\mathcal{R}(\mathcal{S})$ or $\mathcal{R}(\mathcal{I})$) and corresponding traces in Γ^* that do not contain absences.

Lemma 8. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$, $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{S}} = to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})})$.*

Lemma 9. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$ with the same input and output alphabets, if $(L(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq L(\mathcal{S})$ then*

$$\begin{gathered} to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}) \subseteq \\ to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}) \iff \\ L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})} \end{gathered}$$

The following is the key result in this section.

Theorem 2. *Let us suppose that $\mathcal{S}$ is an observable PFSM and $\mathcal{I}$ is an observable FSM with the same input and output alphabets as $\mathcal{S}$. Then, $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $\mathcal{R}(\mathcal{I})$ is a quasi-reduction of $\mathcal{R}(\mathcal{S})$.*

Proof
First, from Lemma 5 we know that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $(L(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq L(\mathcal{S})$ and $(\bar{L}(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq (\bar{L}(\mathcal{S}) \setminus U_{\mathcal{S}})$. By Lemma 2 we know that $\bar{L}(\mathcal{I}) \setminus U_{\mathcal{S}} = to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{S}}$ and $\bar{L}(\mathcal{S}) \setminus U_{\mathcal{S}} = to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S})) \setminus U_{\mathcal{S}}$. Thus, we have that $(\bar{L}(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq (\bar{L}(\mathcal{S}) \setminus U_{\mathcal{S}})$ if and only if $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{S}} \subseteq to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S})) \setminus U_{\mathcal{S}}$. We therefore have that

$$\begin{gathered} \mathcal{I} \approx_Q \mathcal{S} \iff \\ L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S}) \wedge \\ to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{S}} \subseteq to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S})) \setminus U_{\mathcal{S}} \end{gathered}$$

By Lemma 8 we know that $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{S}} = to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})})$ and $to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S})) \setminus U_{\mathcal{S}} = to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})})$ and so

$$\begin{gathered} \mathcal{I} \approx_Q \mathcal{S} \iff \\ L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S}) \wedge \\ to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}) \subseteq to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}) \end{gathered}$$

We can now apply Lemma 9 to get the following.

$$\begin{gathered} \mathcal{I} \approx_Q \mathcal{S} \iff \\ L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S}) \wedge \\ L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})} \end{gathered}$$

All traces of $L(\mathcal{I})$ that are in $U_{\mathcal{S}}$ are also in $U_{\mathcal{R}(\mathcal{S})}$ and so $L(\mathcal{I}) \setminus U_{\mathcal{S}} = L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}$. Since the same holds for $L(\mathcal{S})$ and $L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S})$, we have that

$$\begin{gathered} \mathcal{I} \approx_Q \mathcal{S} \iff \\ L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})} \wedge \\ L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})} \end{gathered}$$

We can now observe that traces in $L(\mathcal{I})$ do not contain absences and all traces in $L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})$ contain absences (Lemma 1) and so

$$
\begin{gathered}
L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})} \wedge \\
(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq (L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S})) \setminus U_{\mathcal{R}(\mathcal{S})} \\
\Longleftrightarrow \\
(L(\mathcal{I}) \cup (L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}))) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq \\
(L(\mathcal{S}) \cup (L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}))) \setminus U_{\mathcal{R}(\mathcal{S})}
\end{gathered}
$$

We therefore have that

$$
\begin{gathered}
\mathcal{I} \approx_Q \mathcal{S} \iff \\
(L(\mathcal{I}) \cup (L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}))) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq \\
(L(\mathcal{S}) \cup (L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}))) \setminus U_{\mathcal{R}(\mathcal{S})}
\end{gathered}
$$

Since $L(\mathcal{I}) \subseteq L(\mathcal{R}(\mathcal{I}))$ and $L(\mathcal{S}) \subseteq L(\mathcal{R}(\mathcal{S}))$, the above term simplifies to

$$\mathcal{I} \approx_Q \mathcal{S} \iff L(\mathcal{R}(\mathcal{I})) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{R}(\mathcal{S})) \setminus U_{\mathcal{R}(\mathcal{S})}$$

The result therefore follows from the definition of quasi-reduction. □

Note that previous work showed how the problem of testing for quasi-reduction can be mapped to a problem of testing for reduction [26]. Combining these results, we have that the problem of testing for quasi-equivalence to an observable PFSM can be mapped to testing for reduction from a completely-specified FSM. As a result, we can use any one of the many automated test generation algorithms (for testing for reduction) when testing for quasi-equivalence [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

5. Making a partial FSM observable

The results in the previous sections were for specifications that are observable (P)FSMs. In this section we consider PFSMs that are not observable. In Section 5.1 we first explore the implementation relation, quasi-equivalence and show how it can be generalised to PFSMs that are not observable. In Section 5.2 we then show how a PFSM can be mapped to an observable PFSM from which we can test.

5.1. Implementation Relations

Recall that if input x is not specified in state s then all possible behaviours of an SUT are allowed if x is received when the specification is in state s (see, for example, [15, 16, 17, 24, 25]). The implementation relation quasi-equivalence has been defined to capture this requirement; it essentially relaxes equivalence to allow the SUT to have any behaviour in response to undefined inputs.

Recall that quasi-equivalence and quasi-reduction were recently generalised to allow PFSMs that do not have harmonised traces [26]. The new definitions considered an input x after trace $\bar{\sigma}$ if two conditions hold: a) $\bar{\sigma}$ is a trace of both the SUT and specification and also b) the input of x after $\bar{\sigma}$ is always defined in the specification. However, the definitions were slightly stronger than necessary if the specification is not observable. This is because they did not consider the

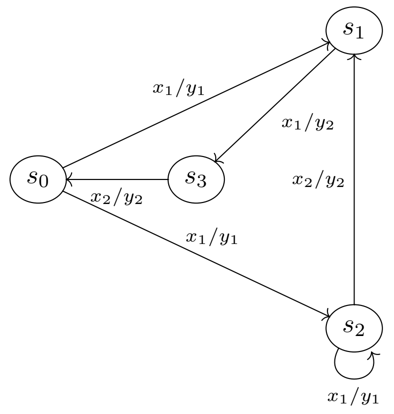


Figure 4: Partial FSM $\mathcal{S}_3$

possibility that $\bar{\sigma}$ has a prefix $\bar{\sigma}_1 x_1/y_1$ such that x_1 is defined in some states of the specification reached by $\bar{\sigma}_1$ but not all. In this section we weaken the definitions to resolve this issue. By weakening the implementation relations for PFSMs that are not observable we potentially allow a wider range of (valid) implementations of the specification. We start by introducing some notation.

Definition 8. *Given state s_1 of PFSM M, we let $L_M^F(s_1)$ denote the set of traces of M such that $\bar{\sigma} \in L_M^F(s_1)$ if and only if $\bar{\sigma} \in L(s_1)$ and for every prefix $\bar{\sigma}_1 x/y$ of $\bar{\sigma}$ we have that $x \in \Omega(s_2)$ for all $s_2 \in s_1$-after-$\bar{\sigma}_1$. If s is the initial state of M then we let $L^F(M)$ denote $L_M^F(s)$.*

Example 1. *Consider the PFSM $\mathcal{S}_3$ shown in Figure 4. In $\mathcal{S}_3$, for example, $x_1/y_1\ x_1/y_1$ and $x_1/y_1\ x_1/y_2$ are both in $L^F(\mathcal{S}_3)$ since x_1 is defined in all states reached by x_1/y_1. In contrast, $x_1/y_1\ x_2/y_2\ x_1/y_2$ is not in $L^F(\mathcal{S}_3)$ since x_2 is not defined in one of the states (s_1) reached by x_1/y_1 even though x_1 is defined in all states reached by $x_1/y_1\ x_2/y_2$.*

We can now define the new version of quasi-equivalence (it is straightforward to prove that this is equivalent to the recently defined version if the specification is observable).

Definition 9. *Given states s_1 and s_2 of PFSM M, s_1 is* quasi-equivalent *to state s_2 if for all $\bar{\sigma} \in L_M^F(s_2) \cap L_M(s_1)$ and $x \in X$, if $x \in \Omega_M(s)$ for all $s \in s_2$-after-$\bar{\sigma}$ then:*

1. *for all $s' \in s_1$-after-$\bar{\sigma}$ we have that $x \in \Omega_M(s')$; and*
2. *$\{y' \in Y | \bar{\sigma} x/y' \in L_M(s_1)\} = \{y' \in Y | \bar{\sigma} x/y' \in L_M(s_2)\}$.*

Given PFSMs $\mathcal{I}$ and $\mathcal{S}$ with the same input and output alphabets, we say that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if the initial state of $\mathcal{I}$ is quasi-equivalent to the initial state of $\mathcal{S}$. We again denote this $\mathcal{I} \approx_Q \mathcal{S}$.

For completeness, we give a corresponding more general definition for quasi-reduction.

Definition 10. *Given states s_1 and s_2 of PFSM M, s_1 is a* quasi-reduction *of state s_2 if for all $\bar{\sigma} \in L_M^F(s_2) \cap L_M(s_1)$ and $x \in X$, if $x \in \Omega_M(s)$ for all $s \in s_2$-after-$\bar{\sigma}$ then:*

1. *for all $s' \in s_1$-after-$\bar{\sigma}$ we have that $x \in \Omega_M(s')$; and*
2. $\{y' \in Y | \bar{\sigma}x/y' \in L_M(s_1)\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L_M(s_2)\}$.

Given PFSMs $\mathcal{I}$ and $\mathcal{S}$ with the same input and output alphabets, we say that $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$ if and only if the initial state of $\mathcal{I}$ is a quasi-reduction of the initial state of $\mathcal{S}$.

5.2. Making PFSMs observable

There are standard approaches that map a completely-specified FSM to an equivalent observable FSM. Essentially, this is because a completely-specified FSM is observable if and only if the corresponding finite automaton is deterministic; one can use standard techniques that transform a finite automaton into an equivalent deterministic finite automaton. However, the following shows that this approach does not work when testing against a PFSM.

Example 2. *Consider the partial FSM $\mathcal{S}_4$ in Figure 5 in which, for example, an arc with label $x_1/y_1, x_2/y_2$ denotes two transitions: one with label x_1/y_1 and one with label x_2/y_2. Further, let us suppose that we turn this into an observable PFSM by using the standard algorithm that takes a finite automaton and returns an equivalent deterministic finite automaton. The finite automaton corresponding to $\mathcal{S}_4$ is non-deterministic because it has two transitions from state s_0 with label x_1/y_1. The standard approach to generating a deterministic finite automaton creates a state corresponding to $\{s_1, s_2\}$, since we need a unique state reached by x_1/y_1, and results in the PFSM $\mathcal{S}_5$ shown in Figure 6. Observe that there is a transition with input x_2 leaving the state $\{s_1, s_2\}$ of $\mathcal{S}_5$. For the SUT to be quasi-equivalent to $\mathcal{S}_5$, or a quasi-reduction of $\mathcal{S}_5$, it must produce output y_2 in response to x_2 after x_1/y_1. However, there is a state of $\mathcal{S}_4$ reached by x_1/y_1 in which x_2 is not defined. As a result, if $\mathcal{S}_4$ is the specification then any output is allowed in response to x_2 after x_1/y_1 (since x_2 is not defined in the state s_1 reached by x_1/y_1). Thus, $\mathcal{S}_4$ and $\mathcal{S}_5$ do not allow the same set of SUTs.*

Motivated by the above, we develop a new technique that involves the addition of transitions that represent the refusal of *inputs*. The approach essentially operates as follows. We add a new output r that will be used to denote the refusal of an input and a new state s_O that denotes the state after a refusal of an input. Given PFSM M, we generate a completely-specified FSM $\mathcal{O}_1(M)$ as follows: whenever x is not defined in state s, we add the transition $(s, s_O, x/r)$. From the new state s_O, for each input x we include transition $(s_O, s_O, x/r)$. Thus, $\mathcal{O}_1(M)$ behaves like M unless input is received in a state where it is not defined; in such circumstances we observe output r. Clearly, $\mathcal{O}_1(M)$ is

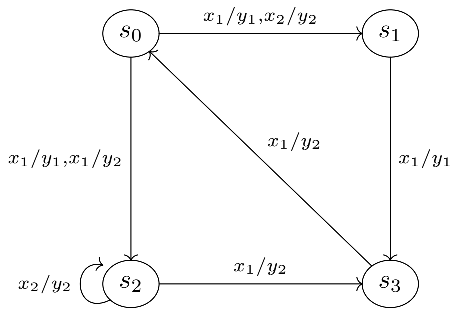


Figure 5: Partial FSM $\mathcal{S}_4$

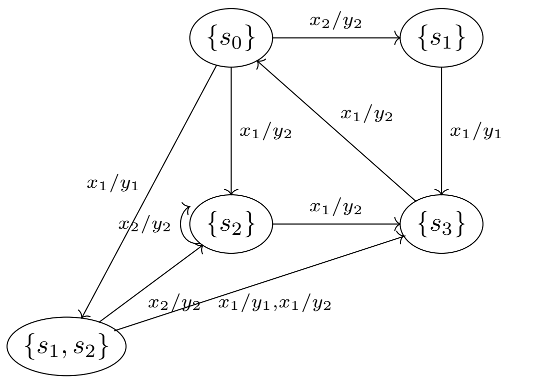


Figure 6: Partial FSM $\mathcal{S}_5$ formed by making $\mathcal{S}_4$ observable

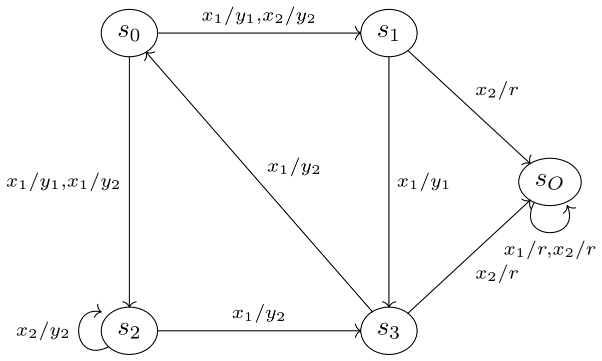


Figure 7: Completely-specified FSM $\mathcal{O}_1(\mathcal{S}_4)$

completely-specified and so one can then map $\mathcal{O}_1(M)$ to an equivalent observable FSM $\mathcal{O}_2(M)$. Finally, we remove some transitions from $\mathcal{O}_2(M)$ to produce the required PFSM $\mathcal{O}(M)$. We now provide the details of this approach.

We start by defining the FSM $\mathcal{O}_1(M)$ that essentially uses an output $r \notin Y$ to denote the refusal of an input. Note that this is similar to the notion of demonic completion used, for example, in testing for ioco [30] though it differs through the inclusion of a label r that denotes a refusal.

Definition 11. *Given PFSM $M = (S, s_0, X, Y, h)$ we let $\mathcal{O}_1(M)$ be the FSM $(S \cup \{s_O\}, s_0, X, Y \cup \{r\}, h')$ in which $s_O \notin S$ and h' is defined by the following for $s \in S \cup \{s_O\}$ and $x \in X$.*

1. *If $(s, x) \in dom\ h$ then $h'(s, x) = h(s, x)$.*
2. *If $(s, x) \notin dom\ h$ then $h'(s, x) = \{(s_O, r)\}$.*

Example 3. *Consider again the PFSM $\mathcal{S}_4$. Here input x_2 is not defined in states s_1 and s_3. The completely-specified FSM $\mathcal{O}_1(\mathcal{S}_4)$ is shown in Figure 7.*

Since $\mathcal{O}_1(M)$ is completely-specified, it is possible to convert it into an equivalent observable FSM $\mathcal{O}_2(M)$; we can use any technique that converts a non-deterministic finite automaton into an equivalent deterministic finite automaton (see, for example, [31]).

Example 4. *Consider $\mathcal{O}_1(\mathcal{S}_4)$. We can convert this into the equivalent observable FSM shown in Figure 8. Note that we have a transition with label x_2/r from the state with label $\{s_1, s_2\}$ since there is a transition with label x_2/r from the state s_1 of $\mathcal{O}_1(\mathcal{S}_4)$.*

We finally construct the PFSM $\mathcal{O}(M)$ from $\mathcal{O}_2(M)$ as follows, in which the essential idea is that if $\mathcal{O}_2(M)$ has a transition of the form $(s, s_O, x/r)$ then this

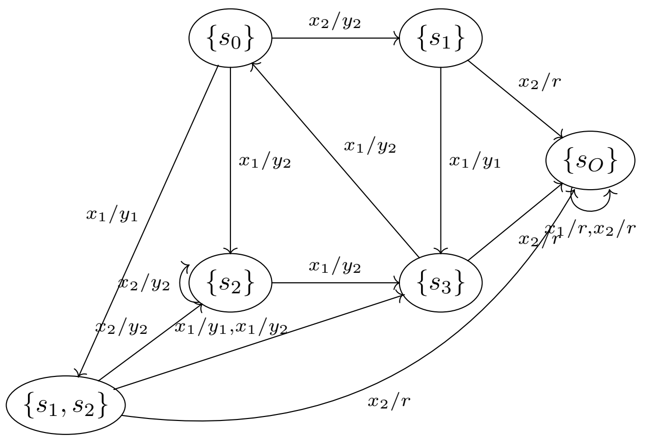


Figure 8: Completely-specified FSM $\mathcal{O}_2(\mathcal{S}_4)$ formed by making $\mathcal{O}_1(\mathcal{S}_4)$ observable

denotes the possibility that (if the state of $\mathcal{O}(M)$ is s) the current state of M is such that x is not defined and so we then remove all transitions from s that have input x. Again, a similar notion has been defined for ioco testing when applying a determinisation step to a model M; having applied a subset construction, an input is only enabled in a given state if it is enabled in all of the corresponding states of M [32].

Definition 12. *Given $\mathcal{O}_2(M) = (S', s'_0, X, Y \cup \{r\}, h_1)$, we construct $\mathcal{O}(M) = (S', s'_0, X, Y, h_2)$ in which h_2 is defined as follows for $s \in S'$ and $x \in X$:*

1. *If there exists state s' such that $(s', r) \in h_1(s, x)$ then $h_2(s, x) = \emptyset$;*
2. *If there does not exist state s' such that $(s', r) \in h_1(s, x)$ then $h_2(s, x) = h_1(s, x)$.*

We then remove any unreachable states and transitions that leave these states.

Example 5. *Consider $\mathcal{O}_2(\mathcal{S}_4)$. We need to remove all transitions with input x from a state s if $\mathcal{O}_2(\mathcal{S}_4)$ has at least one transition from s with label x/r. This involves removing a transition from state $\{s_1, s_2\}$, resulting in the partial FSM shown in Figure 9. This differs from the FSM $\mathcal{S}_5$ in Figure 6 through (as required) x_2 not being specified in state $\{s_1, s_2\}$.*

Note that since $\mathcal{O}_2(M)$ is observable, we must have that $\mathcal{O}(M)$ is observable. We now prove that we can test from $\mathcal{O}(M)$ instead of M.

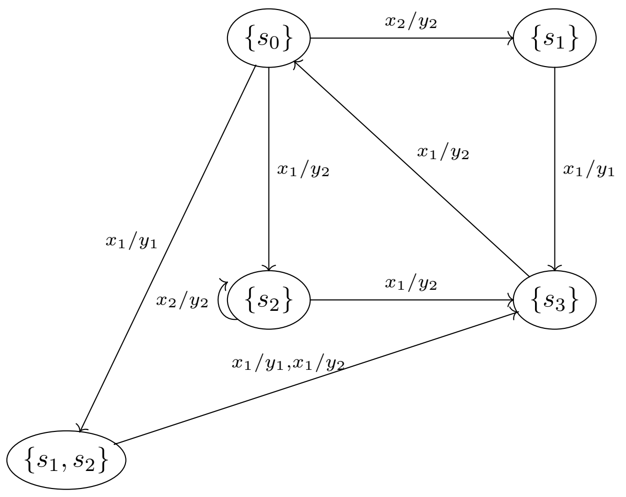


Figure 9: The result of removing transitions from $\mathcal{O}_2(\mathcal{S}_4)$

Theorem 3. *Given PFSM specification $\mathcal{S}$ and FSM $\mathcal{I}$, $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $\mathcal{I}$ is quasi-equivalent to $\mathcal{O}(\mathcal{S})$.*

Proof
First assume that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ and we are required to prove that $\mathcal{I}$ is quasi-equivalent to $\mathcal{O}(\mathcal{S})$. Consider some $\bar{\sigma} \in L^F(\mathcal{O}(\mathcal{S})) \cap L(\mathcal{I})$ and $x \in X$ such that $x \in \Omega_{\mathcal{O}(\mathcal{S})}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$. By construction, since $x \in \Omega_{\mathcal{O}(\mathcal{S})}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$ we have that $\bar{\sigma}x/r \notin L(\mathcal{O}_2(\mathcal{S}))$ and so $x \in \Omega_{\mathcal{S}}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$. Since $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$, by definition we have that for all $\bar{\sigma} \in L^F(\mathcal{S}) \cap L(\mathcal{I})$ and $x \in X$, if $x \in \Omega_{\mathcal{S}}(s)$ for all $s \in \mathcal{S}$-after-$\bar{\sigma}$ then:

1. for all $s' \in \mathcal{I}$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{I}}(s')$; and
2. $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$.

By construction, $\mathcal{O}(\mathcal{S})$ and $\mathcal{S}$ have the same set of traces of the form $\bar{\sigma}x/y$. We therefore have that

1. for all $s' \in \mathcal{I}$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{I}}(s')$; and
2. $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{O}(\mathcal{S}))\}$.

as required.

Now assume that $\mathcal{I}$ is quasi-equivalent to $\mathcal{O}(\mathcal{S})$ and we are required to prove that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$. Since $\mathcal{I}$ is quasi-equivalent to $\mathcal{O}(\mathcal{S})$, by definition we have that for all $\bar{\sigma} \in L^F(\mathcal{O}(\mathcal{S})) \cap L(\mathcal{I})$ and $x \in X$, if $x \in \Omega_{\mathcal{O}(\mathcal{S})}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$ then:

1. for all $s' \in \mathcal{I}$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{I}}(s')$; and

2. $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{O}(\mathcal{S}))\}$.

Now consider some $\bar{\sigma} \in L^F(\mathcal{S}) \cap L(\mathcal{I})$ and $x \in X$ such that $x \in \Omega_{\mathcal{S}}(s)$ for all $s \in \mathcal{S}$-after-$\bar{\sigma}$. We must have that x is defined in all of the states of $\mathcal{O}_1(\mathcal{S})$ reached by $\bar{\sigma}$ and the same set of outputs must be possible in response to x in these states: $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{O}_1(\mathcal{S}))\}$. Since $\mathcal{O}_1(\mathcal{S})$ and $\mathcal{O}_2(\mathcal{S})$ are equivalent, x must be defined in the unique state of $\mathcal{O}_2(\mathcal{S})$ reached by $\bar{\sigma}$ and $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{O}_2(\mathcal{S}))\}$. It is now sufficient to observe that in forming $\mathcal{O}(\mathcal{S})$ from $\mathcal{O}_2(\mathcal{S})$, we retain the unique path with label $\bar{\sigma}$ and also all transitions from state $\mathcal{O}_2(\mathcal{S})$-after-$\bar{\sigma}$ that have input x. The result therefore follows. $\square$

As a result of this, we know that if we are interested in testing for quasi-equivalence to a PFSM $\mathcal{S}$ that is not observable then we can first construct observable $\mathcal{O}(\mathcal{S})$ and we can test for quasi-equivalence to this. We can then use results from previous sections, which tell us that in order to test for quasi-equivalence it is sufficient to test for quasi-reduction to $\mathcal{R}(\mathcal{O}(\mathcal{S}))$.

For completeness, we now give the corresponding result for quasi-reduction.

Theorem 4. *Given PFSM specification $\mathcal{S}$ and FSM $\mathcal{I}$, $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$ if and only if $\mathcal{I}$ is a quasi-reduction of $\mathcal{O}(\mathcal{S})$.*

Proof
First assume that $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$ and we are required to prove that $\mathcal{I}$ is a quasi-reduction of $\mathcal{O}(\mathcal{S})$. Since $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$, by definition we have that for all $\bar{\sigma} \in L^F(\mathcal{S}) \cap L(\mathcal{I})$ and $x \in X$, if $x \in \Omega_{\mathcal{S}}(s)$ for all $s \in \mathcal{S}$-after-$\bar{\sigma}$ then:

1. for all $s' \in \mathcal{I}$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{I}}(s')$; and
2. $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$.

Now consider some $\bar{\sigma} \in L^F(\mathcal{O}(\mathcal{S})) \cap L(\mathcal{I})$ and $x \in X$ such that $x \in \Omega_{\mathcal{O}(\mathcal{S})}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$. By construction, since $x \in \Omega_{\mathcal{O}(\mathcal{S})}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{S}}(s)$ for all $s \in \mathcal{S}$-after-$\bar{\sigma}$. We therefore have that

1. for all $s' \in \mathcal{I}$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{I}}(s')$; and
2. $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$.

as required.

Now assume that $\mathcal{I}$ is a quasi-reduction of $\mathcal{O}(\mathcal{S})$ and we are required to prove that $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$. Since $\mathcal{I}$ is a quasi-reduction of $\mathcal{O}(\mathcal{S})$, by definition we have that for all $\bar{\sigma} \in L^F(\mathcal{O}(\mathcal{S})) \cap L(\mathcal{I})$ and $x \in X$, if $x \in \Omega_{\mathcal{O}(\mathcal{S})}(s)$ for all $s \in \mathcal{O}(\mathcal{S})$-after-$\bar{\sigma}$ then:

1. for all $s' \in \mathcal{I}$-after-$\bar{\sigma}$ we have that $x \in \Omega_{\mathcal{I}}(s')$; and
2. $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{O}(\mathcal{S}))\}$.

Now consider some $\bar{\sigma} \in L^F(\mathcal{S}) \cap L(\mathcal{I})$ and $x \in X$ such that $x \in \Omega_{\mathcal{S}}(s)$ for all $s \in \mathcal{S}$-after-$\bar{\sigma}$. We must have that x is defined in all of the states of $\mathcal{O}_1(\mathcal{S})$ reached by $\bar{\sigma}$ and the same set of outputs must be possible in response to x

in these states: $\{y' \in Y|\bar{\sigma}x/y' \in L(\mathcal{S})\} = \{y' \in Y|\bar{\sigma}x/y' \in L(\mathcal{O}_1(\mathcal{S}))\}$. Since $\mathcal{O}_1(\mathcal{S})$ and $\mathcal{O}_2(\mathcal{S})$ are equivalent, x must be defined in the unique state of $\mathcal{O}_1(\mathcal{S})$ reached by $\bar{\sigma}$ and $\{y' \in Y|\bar{\sigma}x/y' \in L(\mathcal{S})\} = \{y' \in Y|\bar{\sigma}x/y' \in L(\mathcal{O}_2(\mathcal{S}))\}$. It is now sufficient to observe that in forming $\mathcal{O}(\mathcal{S})$ from $\mathcal{O}_2(\mathcal{S})$, we retain the unique path with label $\bar{\sigma}$ and also all transitions from state $\mathcal{O}_2(\mathcal{S})$-after-$\bar{\sigma}$ that have input x. The result therefore follows. □

6. Practical Implications and Case Study

6.1. Practical Implications

This paper introduced a new approach that maps the problem of testing for quasi-equivalence to a partial FSM $\mathcal{S}$ to the problem of testing for quasi-reduction. Recent results have shown that one can map the problem of testing for quasi-reduction to the problem of testing whether the SUT is a reduction of a completely-specified FSM. When we combine these results, we have that we can use test techniques and tools for reduction when testing for quasi-equivalence. The main practical consequence of these results is that testers can utilise the many test generation algorithms, for testing for reduction, when testing for quasi-equivalence (see, for example, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]). The following sequence of steps can be used to achieve this, when testing for quasi-equivalence to PFSM $\mathcal{S}$.

1. Create the observable PFSM $\mathcal{S}_1 = \mathcal{O}(\mathcal{S})$.
2. Generate $\mathcal{S}_2 = \mathcal{R}(\mathcal{S}_1)$.
3. Apply one of the following approaches:
   (a) Test that the SUT is a quasi-reduction of $\mathcal{S}_2$; or
   (b) Use the results in [26] to generate a completely-specified FSM $\mathcal{S}_3$ and generate a test suite from $\mathcal{S}_3$ (for testing for reduction).

It is worth briefly explaining how one applies a test sequence that includes an absence. Let us suppose, for example, that $\bar{\sigma}x/a_y$ is a test sequence for some trace $\bar{\sigma}$. Then we start the process of apply this test sequence by providing the inputs in $\bar{\sigma}$ in the specified order, only continuing if the outputs given in $\bar{\sigma}$ are observed. For example, if $\bar{\sigma} = x_1/y_1x_2/y_2$ then we initially apply x_1 and terminate if the SUT does not output y_1 in response, even if this does not denote a failure. If the SUT produces y_1 in response to x_1 then we next apply x_2. If this process succeeds ($\bar{\sigma}$ is produced by the SUT) then we finally apply input x and observe the output produced; there is a failure if this final output is y (i.e. the absence is not observed). Naturally, a test case need not be a trace and might instead, for example, be defined by a tree/adaptive process. However, the same principles apply.

As previously noted, the main practical benefit is to make a much wider range of test generation techniques available to the tester (when testing from a partial FSM). Note that the results, along with those in [26], also unify several test generation problems, allowing these to be addressed or reasoned about together.

There are some limitations to this work. First, the process of making a partial FSM observable uses a subset construction and so can lead to exponentially many states. However, this issue also arises when considering completely-specified FSMs since the problem of making such an FSM observable is equivalent to the problem of deriving a deterministic finite automaton that is equivalent to a given non-deterministic finite automaton. Thus, this complexity problem is inherent in the area.

A second limitation is that the FSM $\mathcal{R}(\mathcal{S})$ constructed has a sink state and so one may not be able to apply test generation techniques that require the specification to be strongly-connected[4]. However, it should be possible to overcome such issues if the SUT has a reliable reset: a reset operation that is known to take the SUT back to its initial state.

6.2. Case Study

We now illustrate the ideas with a case study that represents a simple game. Essentially, the user chooses to start (input s, output message 1) and the system then randomly chooses one of three 'doors' (not telling the user). The user then chooses one of the three doors: input d_i represents the choice of door i ($1 \leq i \leq 3$). There are two winning conditions: if the user chooses d_1 and the game had chosen door 1, or if the user chooses d_2 and the game had chosen door 2. Note that the choice of d_3 always leads to the user losing (most likely they are not made aware of this!). Winning is represented by output 1; losing by 0. The partial FSM is shown in Figure 10 (left hand side) in which the initial state s_0 is shown twice in order to simplify the diagram. Here, it we want to test for quasi-equivalence and not quasi-reduction. For example, an implementation that did not allow d_1 to potentially win would not be a valid implementation. Note that $\mathcal{G}$ is not observable. Figure 10 (right hand side) shows an observable version, $\mathcal{G}_1 = \mathcal{O}(\mathcal{G})$.

We can now construct the partial FSM $\mathcal{G}_2 = \mathcal{R}(\mathcal{G}_1)$, which is shown in Figure 11; the self-loop transitions are not shown in state S_R. As demonstrated earlier, in order to test against $\mathcal{G}$ for quasi-equivalence it is sufficient to test for quasi-reduction to $\mathcal{G}_2$. Now let us suppose that we wish to test using the trace $s/1d_3/a_1$ that denotes that after $s/1$ the SUT should not be able to produce output 1 in response to d_3. As explained above, testing involve applying input s and observing the resultant output. If output 1 is produced by the SUT in response to s then the tester next applies input d_3 and returns verdict fail if output 1 is observed. As previously discussed, in order to obtain complete testing when testing from a non-deterministic finite state machine, it is normal to make a fairness assumption of the form: if an input sequence has been applied k times (some given k) then all possible resultant output sequences will have been observed. Naturally, the choice of k might be based on the criticality of the system but also on probabilistic arguments.

[4]An FSM M is strongly-connected if for every ordered pair of states (s, s') there is a path that takes M from s to s'.

$$\begin{tikzcd}
 & s_0 \arrow[dl, "s/1"'] \arrow[d, "s/1"'] \arrow[dr, "s/1"] & \\
D_1 \arrow[dr, "{d_1/1,d_2/0,d_3/0}"'] & D_2 \arrow[d, "{d_1/0,d_2/1,d_3/0}"] & D_3 \arrow[dl, "{d_1/0,d_2/0,d_3/0}"] \\
 & s_0 &
\end{tikzcd}
\qquad
\begin{tikzcd}
s_0 \arrow[d, "s/1"'] \\
D \arrow[d, "{d_1/0,d_1/1,d_2/1,d_2/0,d_3/0}"] \\
s_0
\end{tikzcd}$$

Figure 10: Partial FSM $\mathcal{G}$ representing a simple game and $\mathcal{G}_1 = \mathcal{O}(\mathcal{G})$

$$\begin{tikzcd}
s_0 \arrow[d, "s/1"'] & \\
D \arrow[r, "d_3/a_1"'] \arrow[d, "{d_1/0,d_1/1,d_2/1,d_2/0,d_3/0}"] & S_R \\
s_0 &
\end{tikzcd}$$

Figure 11: Partial FSM $\mathcal{G}_2 = \mathcal{R}(\mathcal{G}_1)$

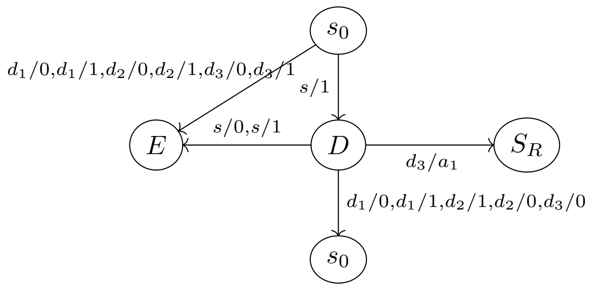

s_0
$d_1/0,d_1/1,d_2/0,d_2/1,d_3/0,d_3/1$
$s/1$
$s/0,s/1$
E
D
S_R
d_3/a_1
$d_1/0,d_1/1,d_2/1,d_2/0,d_3/0$
s_0

Figure 12: Completely-specified FSM $\mathcal{G}_3$

As a final step, we can create the FSM $\mathcal{G}_3$ shown in Figure 12 using the results from [26]. Similar to before, we do not include the self-loop transitions in the error state E. We know that the SUT is correct if and only if it is a reduction of $\mathcal{G}_3$ and so we can use any technique for testing from a completely-specified FSM.

7. Related Work

The work on testing from an FSM specification goes back to Moore's seminal paper in 1956 [33], with Hennie introducing the first test generation algorithm in 1964 [2]. Many FSM-based test generation algorithms have since been introduced [1, 3, 4, 5, 6, 7, 8, 9, 10, 11]. Many of these techniques produce tests that are complete in the sense that they are guaranteed to determine whether the SUT is faulty as long as the SUT is in a well-defined fault domain. Most of the FSM testing literature considers completely-specified FSMs that act as specifications. However, a specification might be partial and so there has been interest in testing from PFSMs [12, 13, 14, 15, 16, 17]. This line of work has treated the situation in which input x is not specified in state s as underspecification; all behaviours are allowed if x is received in state s.

When testing from a PFSM $\mathcal{S}$, one might have the situation in which it is possible to reach states s_1 and s_2 using input sequence $\bar{x}$ and input x is defined in s_1 and not s_2. The problem here is that one cannot safely apply x after $\bar{x}$ since one should not apply x in state s_2; this might preclude the possibility of applying x in state s_1. As a result, work on testing from PFSMs has traditionally required that this scenario cannot happen in the specification PFSM: the specification $\mathcal{S}$ has harmonised traces. Until recently, the two notions of correctness for testing from a PFSM (quasi-equivalence and quasi-reduction) were only defined when the specification has harmonised traces. In addition, algorithms for generating tests from PFSMs required the PFSMs to have harmonised traces.

Although there is practical motivation for requiring the specification to have harmonised traces, it is important to note that the restriction, that an input sequence $\bar{x}$ cannot take the specification to states s_1 and s_2 in which different inputs are defined, is applied even if the associated traces $\bar{\sigma}_1$ and $\bar{\sigma}_2$ are *different*. If $\bar{\sigma}_1$ and $\bar{\sigma}_2$ differ and testing is adaptive (the choice of next input can depend on the outputs that have been observed) then a test case can apply $\bar{x}$ and then choose the next input to apply based on the trace observed. Thus, if testing is adaptive then there is no need to restrict attention to specifications that have harmonised traces. Only recently have quasi-reduction and quasi-equivalence been generalised to allow the PFSM to not have harmonised traces [26] but this work still relies on the PFSM specification being observable. This paper has provided a full generalisation of quasi-reduction and quasi-equivalence to allow any PFSM specification.

Recent work showed that one can transform the problem of testing for quasi-reduction to an observable PFSM to one of testing for reduction to a completely-specified FSM [26]. This was achieved by suitably completing the specification PFSM. Earlier results showed how this could be done with PFSMs that have harmonised traces [15]. However, the completion does not work with quasi-equivalence [26], motivating the work described in this paper.

In this paper we noted that to test for equivalence or quasi-equivalence we must be able to determine, in testing, that the SUT cannot produce a particular output y in response to an input x after a trace $\bar{\sigma}$. This is similar to the notion of a refusal, which typically corresponds to the ability to observe that the SUT cannot engage in an action a. Refusals have been studied in the context of labelled transition systems and process algebras such as CSP (see, for example, [28, 34, 35, 36]).

The absence a_y is not a refusal in the classical sense: a refusal of y would normally be observed by an experiment in which the environment (tester) refuses to engage in events other than y and the composition of the SUT and environment deadlocks. In testing, it is normal to assume that the tester/environment cannot block output (see, for example, [30]) and so we cannot observe a refusal of an output in this way. Recent work on testing from CSP specifications has also noted that the tester cannot observe the refusal of a single output but it does allow the refusal of all outputs to be observed [37], with this corresponding to the observation of quiescence in the *ioco* testing theory [30]. The observation of quiescence is not required when testing from finite state machines since a finite state machine is always quiescent after the execution of a transition.

In order to observe trace $\bar{\sigma}x/a_y$ in testing, it is sufficient to utilise the standard fairness assumption used in testing from a non-deterministic (P)FSM. This involves applying x after $\bar{\sigma}$ at least k times and concluding that y cannot be output, and so a_y has been observed, if y is not produced. Much of the work on testing from non-deterministic finite state machines makes such a fairness assumption (see, for example, [7, 8, 9, 10, 11]).

8. Conclusions

There has been significant interest in automating testing from a finite state machine specification $\mathcal{S}$. Previous work has defined two implementation relations (notions of correctness) for completely-specified FSMs: reduction and equivalence. There are also corresponding implementation relations for testing from a PFSM, with these being quasi-reduction and quasi-equivalence. Recent work showed that we can convert the problem of testing for quasi-reduction to a PFSM specification $\mathcal{S}$ into the problem of testing for reduction to a completely-specified FSM constructed from $\mathcal{S}$. This allows standard test generation techniques (for testing for reduction) to be utilised when testing for quasi-reduction. The main motivation for the work in this paper was to extend this to testing for quasi-equivalence.

We started by discussing how one tests for equivalence, noting that testing must be able to show that the SUT cannot produce output y in response to input x after trace $\bar{\sigma}$. In practice, this would be achieved by having a test case that inputs x after $\bar{\sigma}$ and applying this a sufficient number of times (having made a fairness assumption). We noted that the inability to produce y in response to x after $\bar{\sigma}$ can be represented by trace $\bar{\sigma}x/a_y$ and we called a_y an absence. We showed how FSM M can be enriched to form a new FSM $\mathcal{R}(M)$ that includes such absences and proved that in order to test for equivalence to completely-specified FSM $\mathcal{S}$ it is sufficient to test that the SUT is a reduction of $\mathcal{R}(\mathcal{S})$.

We then considered the problem of testing against PFSM $\mathcal{S}$, exploring the quasi-equivalence implementation relation. We proved that in order to test for quasi-equivalence to PFSM $\mathcal{S}$, one can test for quasi-reduction to $\mathcal{R}(\mathcal{S})$. Recent results have shown that one can map the problem of testing for quasi-reduction to the problem of testing whether the SUT is a reduction of a completely-specified FSM. When we combine these results, we have that we can use test techniques and tools for reduction when testing for quasi-equivalence. The main practical consequence of the results in this paper is that testers can utilise the many test generation algorithms, for testing for reduction, when testing for quasi-equivalence [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]. However, there are additional potential benefits of unifying several test generation problems (e.g. results regarding one problem transfer to the others).

The results were proved for the case where the specification $\mathcal{S}$ is observable. If $\mathcal{S}$ is completely-specified then this is not a significant restriction since we can convert $\mathcal{S}$ into an equivalent observable FSM and test from this. However, this approach does not work when testing from PFSMs. In order to investigate the problem of converting a PFSM specification into an observable PFSM from which we can test, we generalised quasi-equivalence to deal with PFSMs that are not observable. The resulting generalisation allows greater freedom in design and removes the potential to (incorrectly) state that a correct SUT is faulty. We then showed that given PFSM specification $\mathcal{S}$, we can produce an observable PFSM $\mathcal{O}(\mathcal{S})$ from which we can test: the SUT is quasi-equivalent to $\mathcal{S}$ if and only if it is quasi-equivalent to $\mathcal{O}(\mathcal{S})$, with the corresponding result also holding for quasi-reduction. When combined with previous results, this shows that the problems of testing for quasi-reduction, equivalence, and quasi-equivalence can all be mapped to a corresponding problem of testing for reduction.

Regarding future work, it would be interesting to explore whether corresponding results hold for other FSM problems such as constructing synchronising sequences or distinguishing states.

[1] T. S. Chow, Testing software design modelled by finite state machines, IEEE Transactions on Software Engineering 4 (1978) 178–187.

[2] F. C. Hennie, Fault-detecting experiments for sequential circuits, in: Proceedings of Fifth Annual Symposium on Switching Circuit Theory and Logical Design, Princeton, New Jersey, 1964, pp. 95–110.

[3] H. Ural, X. Wu, F. Zhang, On minimizing the lengths of checking sequences, IEEE Transactions on Computers 46 (1) (1997) 93–99.

[4] A. Simão, A. Petrenko, N. Yevtushenko, On reducing test length for FSMs with extra states, Software Testing, Verification and Reliability 22 (6) (2012) 435–454.

[5] R. M. Hierons, U. C. Türker, Incomplete distinguishing sequences for finite state machines, The Computer Journal 58 (11) (2015) 3089–3113.

[6] R. M. Hierons, U. C. Türker, Parallel algorithms for testing finite state machines: Generating UIO sequences, IEEE Transactions on Software Engineering 42 (11) (2016) 1077–1091.

[7] H. AboElFotoh, O. Abou-Rabia, H. Ural, A test generation algorithm for protocols modeled as non-deterministic FSMs, The Software Engineering Journal 8 (4) (1993) 184–188.

[8] R. M. Hierons, Generating candidates when testing a deterministic implementation against a non–deterministic finite state machine, The Computer Journal 46 (3) (2003) 307–318.

[9] I. Hwang, T. Kim, S. Hong, J. Lee, Test selection for a nondeterministic FSM, Computer Communications 24 (12) (2001) 1213–1223.

[10] A. Petrenko, A. Simão, N. Yevtushenko, Generating checking sequences for nondeterministic finite state machines, in: Fifth IEEE International Conference on Software Testing, Verification and Validation (ICST 2012), 2012, pp. 310–319.

[11] F. Zhang, T.-Y. Cheung, Optimal transfer trees and distinguishing trees for testing observable nondeterministic finite-state machines, IEEE Transactions on Software Engineering 29 (1) (2003) 1–14.

[12] A. L. Bonifácio, A. V. Moura, Test suite completeness and partial models, in: 12th International Conference on Software Engineering and Formal Methods (SEFM 2014), Vol. 8702 of LNCS, Springer, 2014, pp. 96–110.

[13] A. L. Bonifácio, A. V. Moura, Partial models and weak equivalence, in: 11th International Colloquium on Theoretical Aspects of Computing (ICTAC 2014), Vol. 8687 of LNCS, Springer, 2014, pp. 80–96.

[14] A. L. Bonifácio, A. V. Moura, On the completeness of test suites, in: Symposium on Applied Computing (SAC 2014), ACM, 2014, pp. 1287–1292.

[15] A. Petrenko, N. Yevtushenko, G. v. Bochmann, Testing deterministic implementations from nondeterministic FSM specifications, in: IFIP TC6 9th International Workshop on Testing of Communicating Systems, Chapman and Hall, Darmstadt, Germany, 1996, pp. 125–141.

[16] A. Petrenko, N. Yevtushenko, Testing from partial deterministic FSM specifications, IEEE Transactions on Computers 54 (9) (2005) 1154–1165.

[17] A. Petrenko, N. Yevtushenko, Conformance tests as checking experiments for partial nondeterministic FSM, in: 5th International Workshop on Formal Approaches to Software Testing (FATES 2005), Vol. 3997 of Lecture Notes in Computer Science, Springer, 2006, pp. 118–133.

[18] A. Bertolino, E. Marchetti, H. Muccini, Introducing a reasonably complete and coherent approach for model-based testing, Electr. Notes Theor. Comput. Sci. 116 (2005) 85–97.

[19] M. Kalinowski, M. Felderer, T. Conte, R. O. Spínola, R. Prikladnicki, D. Winkler, D. M. Fernández, S. Wagner, Preventing incomplete/hidden requirements: Reflections on survey data from austria and brazil, in: Software Quality. The Future of Systems- and Software Development, 8th International Conference (SWQD 2016), Vol. 238 of Lecture Notes in Business Information Processing, Springer, 2016, pp. 63–78.

[20] N. Kushik, N. Yevtushenko, A. R. Cavalli, On testing against partial non-observable specifications, in: 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014, IEEE Computer Society, 2014, pp. 230–233.

[21] F. Brglez, ACM/SIGMOD benchmark dataset, available online at `http://cbl.ncsu.edu:16080/benchmarks/Benchmarks-upto-1996.html`.

[22] R. M. Hierons, U. C. Türker, Distinguishing sequences for partially specified FSMs, in: NASA Formal Methods - 6th International Symposium, NFM 2014,, Vol. 8430 of Lecture Notes in Computer Science, Springer, 2014, pp. 62–76.

[23] M.-C. Gaudel, Testing can be formal too, in: 6th International Joint Conference CAAP/FASE Theory and Practice of Software Development (TAPSOFT'95), Vol. 915 of Lecture Notes in Computer Science, Springer, 1995, pp. 82–96.

[24] G. Luo, A. Petrenko, G. v. Bochmann, Selecting test sequences for partially-specified nondeterministic finite state machines, in: The 7th IFIP Workshop on Protocol Test Systems, Chapman and Hall, Tokyo, Japan, 1994, pp. 95–110.

[25] A. Simão, A. Petrenko, Generating checking sequences for partial reduced finite state machines, in: 20th IFIP TC 6/WG 6.1 International Conference Testing of Software and Communicating Systems, 8th International Workshop on Formal Approaches to Testing of Software (TestCom/FATES 2008), Vol. 5047 of Lecture Notes in Computer Science, Springer, 2008, pp. 153–168.

[26] R. M. Hierons, Testing from partial finite state machines without harmonised traces, IEEE Transactions on Software Engineering 43 (11) (2017) 1033–1043.

[27] G. L. Luo, G. v. Bochmann, A. Petrenko, Test selection based on communicating nondeterministic finite-state machines using a generalized Wp-method, IEEE Transactions on Software Engineering 20 (2) (1994) 149–161.

[28] R. v. Glabbeek, The linear time-branching time spectrum I. The semantics of concrete, sequential processes, in: J. Bergstra, A. Ponse, S. Smolka (Eds.), Handbook of process algebra, North Holland, 2001, Ch. 1.

[29] A. Cavalcanti, R. M. Hierons, Testing with inputs and outputs in CSP, in: 16th International Conference on Fundamental Approaches to Software Engineering (FASE 2013), Vol. 7793 of Lecture Notes in Computer Science, Springer, 2013, pp. 359–374.

[30] J. Tretmans, Model based testing with labelled transition systems, in: Formal Methods and Testing, Vol. 4949 of Lecture Notes in Computer Science, Springer, 2008, pp. 1–38.

[31] J. E. Hopcroft, An n log n algorithm for minimizing the states in a finite automaton, in: Z. Kohavi (Ed.), The theory of Machines and Computation, Academic Press, 1971, pp. 189–196.

[32] B. K. Aichernig, H. Brandl, E. Jöbstl, W. Krenn, Model-based mutation testing of hybrid systems, in: 8th International Symposium on Formal Methods for Components and ObjectsFMCO 2009, Vol. 6286 of Lecture Notes in Computer Science, 2009, pp. 228–249.

[33] E. F. Moore, Gedanken-experiments, in: C. Shannon, J. McCarthy (Eds.), Automata Studies, Princeton University Press, 1956.

[34] L. Heerink, J. Tretmans, Refusal testing for classes of transition systems with inputs and outputs, in: Formal Description Techniques and Protocol Specification, Testing and Verification (FORTE X/PSTV XVII), Vol. 107 of IFIP Conference Proceedings, Chapman & Hall, 1997, pp. 23–38.

[35] I. Phillips, Refusal testing, Theoretical Computer Science 50 (3) (1987) 241–284.

[36] A. Roscoe, The Theory and Practice of Concurrency, Prentice Hall, 1998.

[37] A. Cavalcanti, R. M. Hierons, S. Nogueira, A. Sampaio, A suspension-trace semantics for CSP, in: 10th International Symposium on Theoretical Aspects of Software Engineering (TASE 2016), IEEE Computer Society, 2016, pp. 3–13.

Lemma 2. *Given observable PFSM M, $\bar{L}(M) = to\_out(L(\mathcal{R}(M)) \setminus L(M))$.*

Proof
First we will assume that $\bar{\sigma} \in to\_out(L(\mathcal{R}(M)) \setminus L(M))$ and we are required to prove that $\bar{\sigma} \in \bar{L}(M)$. Let $\bar{\gamma}$ be a trace in $L(\mathcal{R}(M)) \setminus L(M)$ such that $to\_out(\bar{\gamma}) = \bar{\sigma}$. By the definition of $\mathcal{R}(M)$, we must have that $\bar{\gamma}$ has prefix $\bar{\gamma}_1 = \bar{\sigma}_1 x/a_y$ for some $\bar{\sigma}_1 \in \Sigma^*$, $x \in X$, and $y \in Y$. Since M is observable, there is only one state of M that can be reached by a path with label $\bar{\sigma}_1$ and so we must have that $\bar{\sigma}_1 x/y \notin L(M)$. However, $\bar{\sigma}_1 x/y$ is a prefix of $to\_out(\gamma) = \bar{\sigma}$. Since $L(M)$ is prefix closed, we must have that $\bar{\sigma} \notin L(M)$ and so $\bar{\sigma} \in \bar{L}(M)$ as required.

Now we will assume that $\bar{\sigma} \in \bar{L}(M)$ and we are required to prove that $\bar{\sigma} \in to\_out(L(\mathcal{R}(M)) \setminus L(M))$. We must have that $\bar{\sigma} = \bar{\sigma}_1 x/y\bar{\sigma}_2$ where $\bar{\sigma}_1$ is the longest prefix of $\bar{\sigma}$ that is in $L(M)$. Thus, $\bar{\sigma}_1$ labels a unique path of M and that path ends in a state s from which there is no transition with input x and output y. By the construction of $\mathcal{R}(M)$ we therefore have that $\bar{\sigma}$ is the image, under $to\_out()$, of some $\bar{\gamma} \in L(\mathcal{R}(M))$. Further, $\bar{\gamma}$ must contain at least one absence and so we have that $\bar{\gamma} \notin L(M)$. We therefore have that $\bar{\gamma} \in L(\mathcal{R}(M)) \setminus L(M)$ and so $\bar{\sigma} \in to\_out(L(\mathcal{R}(M)) \setminus L(M))$ as required. □

Lemma 4. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$ with the same input and output alphabets, $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ if and only if $L(\mathcal{I}) \setminus U_{\mathcal{S}} = L(\mathcal{S})$.*

Proof
First, let us suppose that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$ and let $\bar{\sigma}$ be a trace that is not in $U_{\mathcal{S}}$; we require to prove that $\bar{\sigma} \in L(\mathcal{I})$ if and only if $\bar{\sigma} \in L(\mathcal{S})$. Proof by contradiction: assume that this does not hold and so $\bar{\sigma} \notin L(\mathcal{I}) \cap L(\mathcal{S})$ and $\bar{\sigma} \in L(\mathcal{I}) \cup L(\mathcal{S})$. Thus, $\bar{\sigma} = \bar{\sigma}_1 x/y\bar{\sigma}_2$ for some maximal $\bar{\sigma}_1 \in L(\mathcal{S}) \cap L(\mathcal{I})$. Since $\bar{\sigma} \notin U_{\mathcal{S}}$, we have that x is defined in the unique state of $\mathcal{S}$ reached by $\bar{\sigma}_1$. Since $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$, we must have that x is also defined in the unique state of $\mathcal{I}$ reached by $\bar{\sigma}_1$. But this implies that x is defined after $\bar{\sigma}_1$ in $\mathcal{I}$ and $\mathcal{S}$ but also that one of $\mathcal{I}$ and $\mathcal{S}$ can produce output y in response to x after $\bar{\sigma}_1$ and the other cannot. This contradicts the second condition of the definition of $\mathcal{I}$ being quasi-equivalent to $\mathcal{S}$, as required.

Now suppose that $L(\mathcal{I}) \setminus U_{\mathcal{S}} = L(\mathcal{S})$ and we are required to prove that $\mathcal{I}$ is quasi-equivalent to $\mathcal{S}$. Let us suppose that $\bar{\sigma} \in L(\mathcal{I}) \cap L(\mathcal{S})$, $x \in X$, and $(\bar{\sigma}, x) \in D_{\mathcal{S}}$. By definition, we require to prove that $(\bar{\sigma}, x) \in D_{\mathcal{I}}$ and $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$. Since $(\bar{\sigma}, x) \in D_{\mathcal{S}}$, $U_{\mathcal{S}}$ does not contain any trace of the form $\bar{\sigma}x/y$. Thus, since $L(\mathcal{I}) \setminus U_{\mathcal{S}} = L(\mathcal{S}) \setminus U_{\mathcal{S}}$, we must have that $L(\mathcal{I})$ has at least one trace of the form $\bar{\sigma}x/y$ and so we can conclude that $(\bar{\sigma}, x) \in D_{\mathcal{I}}$. Further, since $L(\mathcal{I}) \setminus U_{\mathcal{S}} = L(\mathcal{S})$, we can conclude that $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} = \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$ as required. The result therefore follows. □

The following proof uses the fact that $\mathcal{I}$ is completely-specified.

Lemma 6. *Let us suppose that $\mathcal{S}$ is an observable PFSM and $\mathcal{I}$ is an observable FSM with the same input and output alphabets. Then, $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$ if and only if $L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S})$.*

Proof
First, let us suppose that $\mathcal{I}$ is a quasi-reduction $\mathcal{S}$ and let $\bar{\sigma}$ be a trace that is not in $U_{\mathcal{S}}$; we require to prove that $\bar{\sigma} \in L(\mathcal{I})$ implies that $\bar{\sigma} \in L(\mathcal{S})$. Proof by contradiction: assume that this does not hold and so $\bar{\sigma} \in L(\mathcal{I})$ and $\bar{\sigma} \notin L(\mathcal{S})$. Thus, $\bar{\sigma} = \bar{\sigma}_1 x/y \bar{\sigma}_2$ for some maximal $\bar{\sigma}_1 \in L(\mathcal{S}) \cap L(\mathcal{I})$. Since $\bar{\sigma} \notin U_{\mathcal{S}}$, we have that x is defined in the unique state of $\mathcal{S}$ reached by $\bar{\sigma}_1$. Since $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$, we must also have that x is also defined in the unique state of $\mathcal{I}$ reached by $\bar{\sigma}_1$. But this implies that x is defined after $\bar{\sigma}_1$ in $\mathcal{I}$ and $\mathcal{S}$ but also that $\mathcal{I}$ can produce output y in response to x after $\bar{\sigma}_1$ and $\mathcal{S}$ cannot. This contradicts the definition of $\mathcal{I}$ being a quasi-reduction of $\mathcal{S}$, as required.

Now suppose that $L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S}) \setminus U_{\mathcal{S}}$ and we are required to prove that $\mathcal{I}$ is a quasi-reduction of $\mathcal{S}$. Let us suppose that $\bar{\sigma} \in L(\mathcal{I}) \cap L(\mathcal{S})$, $x \in X$, and $(\bar{\sigma}, x) \in D_{\mathcal{S}}$. By definition, we require to prove that $(\bar{\sigma}, x) \in D_{\mathcal{I}}$ and $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$. The first follows from $\mathcal{I}$ being completely-specified. Since $(\bar{\sigma}, x) \in D_{\mathcal{S}}$, for all $y \in Y$ we have that $\bar{\sigma}x/y \notin U_{\mathcal{S}}$. Thus, since $L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S}) \setminus U_{\mathcal{S}}$, we can conclude that $\{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{I})\} \subseteq \{y' \in Y | \bar{\sigma}x/y' \in L(\mathcal{S})\}$ as required. The result therefore follows. □

Lemma 8. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$, $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I})) \setminus U_{\mathcal{S}} = to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})})$.*

Proof
It is sufficient to prove that if $\bar{\gamma} \in L(\mathcal{R}(N)) \setminus L(N)$ then $\bar{\gamma} \in U_{\mathcal{R}(M)}$ if and only if $to\_out(\bar{\gamma}) \in U_M$. Let $\bar{\sigma} = to\_out(\bar{\gamma})$. Observe that $\bar{\gamma}$ contains at least one absence.

By the definition of $\mathcal{R}(M)$, $\bar{\gamma} \in U_{\mathcal{R}(M)}$ if and only if $\bar{\gamma} = \bar{\sigma}_1 \bar{\gamma}_2$ for some $\bar{\sigma}_1 \in L(M)$ and non-empty $\bar{\gamma}_2 \in (\Gamma \cup \Gamma_a)^*$ where the first input x in $\bar{\gamma}_2$ is such that $(\bar{\sigma}_1, x) \notin D_M$. However, we also have that $\bar{\sigma} \in U_M$ if and only if we can write $\bar{\sigma}$ as $\bar{\sigma}_1 \bar{\sigma}_2$ for some $\bar{\sigma}_1 \in L(M)$ and non-empty $\bar{\sigma}_2 \in \Gamma^*$ where the first input x in $\bar{\sigma}_2$ is such that $(\bar{\sigma}_1, x) \notin D_M$. Since $\bar{\sigma} = to\_out(\bar{\gamma})$, these conditions are equivalent and so the result follows. □

Lemma 9. *Given observable PFSMs $\mathcal{S}$ and $\mathcal{I}$ with the same input and output alphabets, if $(L(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq L(\mathcal{S})$ then*

$$\begin{gathered} to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}) \subseteq \\ to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}) \iff \\ L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})} \end{gathered}$$

Proof
The right-to-left result is immediate and so we assume that $L(\mathcal{I}) \setminus U_{\mathcal{S}} \subseteq L(\mathcal{S}) \setminus U_{\mathcal{S}}$

and $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}) \subseteq to\_out(L(\mathcal{R}(\mathcal{S}))) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})})$ and we are required to prove that $L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})} \subseteq L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}$.

We use proof by contradiction and suppose that $\bar{\gamma}$ is a minimal element of $L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}$ such that $\bar{\gamma} \notin L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}$. Let $\bar{\sigma} = to\_out(\bar{\gamma})$. Then $\bar{\sigma} = \bar{\sigma}_1\bar{\sigma}_2$ for some maximal prefix $\bar{\sigma}_1 \in L(\mathcal{I})$. Since $\bar{\gamma} \notin U_{\mathcal{R}(\mathcal{S})}$, we also have that $\bar{\sigma}_1 \notin U_{\mathcal{S}}$ (if $\bar{\sigma}_1 \in U_{\mathcal{S}}$ then, by definition, all extensions of $\bar{\sigma}_1$ are in $U_{\mathcal{S}}$ and so also in $U_{\mathcal{R}(\mathcal{S})}$). Further, by construction, we have that $\bar{\gamma} = \bar{\sigma}_1\bar{\gamma}_2$ for some $\bar{\gamma}_2 \in \Gamma_a^*$ such that $\bar{\sigma}_2 = to\_out(\bar{\gamma}_2)$.

Since $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}) \subseteq to\_out(L(\mathcal{R}(\mathcal{S}))) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})})$ we have that $\bar{\sigma} \in to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})})$. Let $\bar{\gamma}'$ be the trace in $L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}$ such that $to\_out(\bar{\gamma}') = \bar{\sigma}$; since $\mathcal{R}(\mathcal{S})$ is observable, $\bar{\gamma}'$ is uniquely defined. Since $\bar{\sigma}_1 \in L(\mathcal{I})$ and $\bar{\sigma}_1 \notin U_{\mathcal{S}}$, we have that $\bar{\sigma}_1 \in L(\mathcal{I}) \setminus U_{\mathcal{S}}$. Since $(L(\mathcal{I}) \setminus U_{\mathcal{S}}) \subseteq (L(\mathcal{S}) \setminus U_{\mathcal{S}})$, we must have that $\bar{\sigma}_1 \in L(\mathcal{S}) \setminus U_{\mathcal{S}}$. Thus, $\bar{\sigma}_1$ is a prefix of $\bar{\gamma}'$ and so $\bar{\gamma}' = \bar{\sigma}_1\bar{\gamma}'_2$ for some $\bar{\gamma}'_2 \in \Gamma^*\Gamma_a^*$.

Since $\bar{\gamma} \notin L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})}$ we must have that $\bar{\gamma}'_2 \notin \Gamma_a^*$ (since then we would have $\bar{\gamma}_2 = \bar{\gamma}'_2$). Thus, $\bar{\gamma}' = \bar{\sigma}_1 x/y\bar{\gamma}_3$ for some $x/y \in \Gamma$. But this implies that $\bar{\sigma}_1 x/y \in L(\mathcal{S})$ and so, since $\mathcal{S}$ is observable, $\bar{\sigma}_1 x/a_y \notin L(\mathcal{R}(\mathcal{S}))$. But this tells us that $\bar{\sigma}_1 x/y \notin to\_out(L(\mathcal{R}(\mathcal{S})) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})})$. However, $\bar{\sigma}_1 x/y \in to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})})$ and this contradicts the assumption that $to\_out(L(\mathcal{R}(\mathcal{I})) \setminus L(\mathcal{I}) \setminus U_{\mathcal{R}(\mathcal{S})}) \subseteq to\_out(L(\mathcal{R}(\mathcal{S}))) \setminus L(\mathcal{S}) \setminus U_{\mathcal{R}(\mathcal{S})})$ as required. $\square$