Checking Finite State Machine Conformance when there are Distributed Observations

Robert M. Hierons

Abstract

This paper concerns state-based systems that interact with their environment at physically distributed interfaces, called ports. When such a system is used a projection of the global trace, called a local trace, is observed at each port. This leads to the environment having reduced observational power: the set of local traces observed need not uniquely define the global trace that occurred. We consider the previously defined implementation relation $\sqsubseteq_s$ and start by investigating the problem of defining a language $\tilde{\mathcal{L}}(M)$ for a multi-port finite state machine (FSM) M such that $N \sqsubseteq_s M$ if and only if every global trace of N is in $\tilde{\mathcal{L}}(M)$. The motivation is that if we can produce such a language $\tilde{\mathcal{L}}(M)$ then this can potentially be used to inform development and testing. We show that $\tilde{\mathcal{L}}(M)$ can be uniquely defined but need not be regular. We then prove that it is generally undecidable whether $N \sqsubseteq_s M$, a consequence of this result being that it is undecidable whether there is a test case that is capable of distinguishing two states or two multi-port FSM in distributed testing. This result complements a previous result that it is undecidable whether there is a test case that is guaranteed to distinguish two states or multi-port FSMs. We also give some conditions under which $N \sqsubseteq_s M$ is decidable. We then consider the implementation relation $\sqsubseteq_s^k$ that only concerns input sequences of length k or less. Naturally, given FSMs N and M it is decidable whether $N \sqsubseteq_s^k M$ since only a finite set of traces is relevant. We prove that if we place bounds on k and the number of ports then we can decide $N \sqsubseteq_s^k M$ in polynomial time but otherwise this problem is NP-hard.

1 Introduction

Many systems interact with their environment at multiple physically distributed interfaces, called ports, with web-services, cloud systems and wireless sensor networks being important classes of such systems. When we test a system that has multiple ports we place a local tester at each port and the local tester at port p only observes the events at p. This has led to the (ISO standardised) definition of the distributed test architecture in which we have a set of distributed testers, the testers do not communicate with one another during testing, and there is no global clock [23]. While it is sometimes possible to make testing more effective by allowing the testers to exchange coordination messages during testing [4, 31], this is not always feasible and the distributed test architecture is typically simpler and cheaper to implement. Importantly, the situation in which separate agents (users or testers) interact with the system at its ports can correspond to the expected use of the system.

Distributed systems often have a persistent internal state and such systems are thus modeled or specified using state-based languages. In the context of testing the focus has largely been on finite state machines (FSMs) and input output transition systems (IOTSs). This is both because such approaches are suitable and because most tools and techniques for model-based testing[1] transform the models, written in a high-level notation, to an FSM or IOTS [5, 12, 13, 11, 34]. Model-based testing has received much recent attention since it facilitates test automation, the results of a recent major industrial project showing the potential for significant reductions in the cost of testing [13].

[1] In model-based testing, test automation is based on a model of the expected behaviour of the system or some aspect of this expected behaviour.

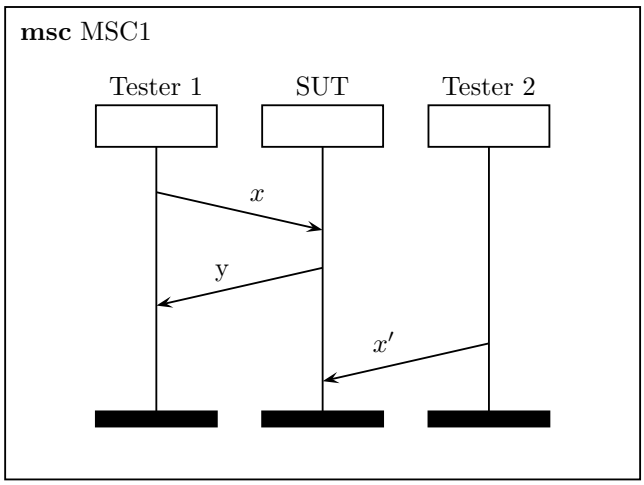

Figure 1. A controllability problem caused by input x'

This paper concerns problems related to developing a multi-port system based on a (multi-port) FSM model/specification. Much of the work in the area of distributed testing has focussed on FSM models [7, 9, 10, 24, 32, 36], although there has also been work that considers more general models such as IOTSs and variants of IOTSs [14, 37, 20, 19, 21]. While IOTSs are more expressive, this paper explores decidability and complexity issues in distributed testing and so we restrict attention to finite state models and, in particular, to multi-port FSMs. Naturally, the negative decidability and complexity results proved in this paper extend immediately to IOTSs.

When a state-based system interacts with its environment there is a resultant sequence of inputs and outputs called a *global trace*. When there are physically distributed ports the user or tester at a port p only observes the sequence of events that occur at p, the projection at p of the global trace, and this is called a *local trace*. It is known that the local testers only observing local traces introduces additional issues into testing [7, 9, 10, 14, 37, 24, 32, 36].

Previous work has shown that distributed testing introduces additional controllability and observability problems. A controllability problem occurs when a tester does not know when to supply an input due to it not observing the events at the other ports [32, 9]. Consider, for example, the global trace shown in Figure 1. We use diagrams (message sequence charts) such as this to represent scenarios. In such diagrams vertical lines represent processes and time progresses as we go down a line. In this case the system under test (SUT) has two ports, 1 and 2, we have one vertical line representing the SUT, one representing the local tester at port 1, and one representing the local tester at port 2. There is a controllability problem because the tester at port 2 should send input x' after y has been sent by the SUT but cannot know when this has happened since it does not observe the events at port 1

Observability problems refer to the fact that the observational ability of a set of distributed testers is less than that of a global tester since the set of local traces need not uniquely define the global trace that occurred [10]. Consider, for example, the global traces σ and σ' shown in Figures 2 and 3 respectively. These global traces are different but the local testers observe the same local traces: in each case the tester at port 1 observes $xyxy$ and the tester at port 2 observes y'. Recent work has defined new notions of conformance (implementation relations) that recognise this reduced observational power of the environment [16, 20]. These implementation relations essentially say that the SUT conforms to the specification if the environment cannot observe a failure. When using such implementation relations, we do not have to consider observability problems: if a global trace σ of the SUT is observationally equivalent to one in the specification then σ is considered to be an allowed behaviour since a set of distributed testers or users would not observe a failure.

Given multi-port FSMs N and M, there are two notions of conformance for situations in which distributed observations are made: weak conformance ($\sqsubseteq_w$) and strong conformance ($\sqsubseteq_s$). Under $\sqsubseteq_w$, it is sufficient that for every global trace σ of N and port p there is some global trace σ_p of M such that σ and σ_p are indistinguishable at port p; they have the same local traces at p. In contrast, under $\sqsubseteq_s$ we require that for every global trace σ of N there is some global trace σ' of M such that σ and σ' are indistinguishable at all of the ports. To see the difference, let us suppose that there are two allowed responses to input x_1 at port 1: either y_1 at port 1 and y_2 at port 2

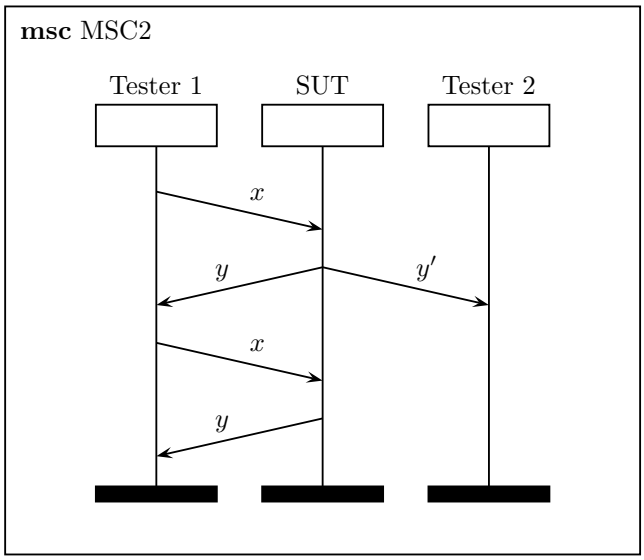

Figure 2. Global trace σ.

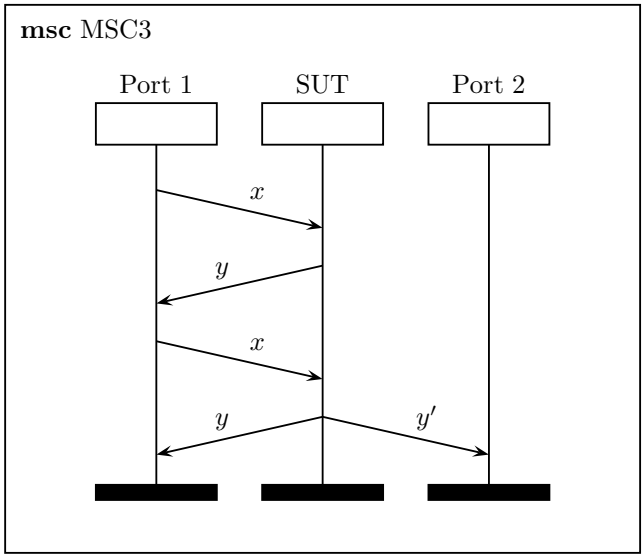

Figure 3. Global trace σ'.

(forming global trace σ) or y'_1 at port 1 and y'_2 at port 2 (forming global trace σ') . Under $\sqsubseteq_w$ it is acceptable for the SUT to respond to x_1 with y_1 at port 1 and y'_2 at port 2 since the local trace at port 1 is x_1y_1, which is a projection of σ, and the local trace at port 2 is y'_2, which is a projection of σ'. However, this is not acceptable under $\sqsubseteq_s$ since there is no global trace of the specification that has projection x_1y_1 at port 1 and projection y'_2 at port 2.

One of the benefits of using an FSM to model the required behaviour of a system that interacts with its environment at only one port is that there are standard algorithms for many problems that are relevant to test generation. For example, we can decide whether there are strategies (test cases) that reach or distinguish states [2] and such strategies are used by many test generation algorithms [1, 6, 15, 26, 28, 35]. In addition, if we have an FSM specification M and an FSM design N then we can decide whether N conforms to M. Thus, if we wish to adapt standard FSM test techniques to the situation where we have distributed testers then we need to investigate corresponding problems for multi-port FSMs. Recent work has shown that it is undecidable whether there is a strategy that is guaranteed to reach a state or distinguishes two states of an FSM in distributed testing [18]. However, this left open the question of whether one can decide whether one FSM conforms to another. It also left open the related question of whether it is decidable whether there is a strategy that is capable of distinguishing two FSMs[2].

This paper concerns the problem of deciding, for multi-port FSMs M and N, whether N conforms to M. Clearly, this can be decided in low order polynomial time for $\sqsubseteq_w$: for each port p we simply compare the projections of N and M at p. However, $\sqsubseteq_w$ will often be too weak since it assumes that we can never have the situation in which an agent is aware of observations made at two or more ports. We therefore focus on the implementation relation $\sqsubseteq_s$.

We start by investigating the question of whether, given a multi-port FSM M, we can define a language $\tilde{\mathcal{L}}(M)$ such that for every multi-port FSM N we have that $N \sqsubseteq_s M$ if and only if all global traces of N are in $\tilde{\mathcal{L}}(M)$. If we can define such an $\tilde{\mathcal{L}}(M)$ then there is the potential to explore properties of this in order to find algorithms for deciding $N \sqsubseteq_s M$ for classes of N and M. There is also the potential to base testing and development on $\tilde{\mathcal{L}}(M)$. It has already been shown that we can produce such a language $\tilde{\mathcal{L}}(M)$ for the special case where we restrict testing to controllable input sequences and are testing from deterministic FSMs [17]. We prove that $\tilde{\mathcal{L}}(M)$ is uniquely defined but need not be regular.

We then consider the problem of determining whether $N \sqsubseteq_s M$ for multi-port FSMs N an M and prove that this is generally undecidable. We also give some conditions under which $N \sqsubseteq_s M$ is decidable. Clearly, this problem is important when we are checking an FSM design against an FSM specification. In addition, $N \sqsubseteq_s M$ if no *possible* behaviour of N can be distinguished from the behaviours of M. Thus, since it is undecidable whether $N \sqsubseteq_s M$ it is also undecidable whether there is a test case that is *capable* of distinguishing two states or FSMs. This complements the result that it is undecidable whether there is a test case that is guaranteed to distinguish two states or FSMs [18]. However, the proofs use very different approaches: the proof of the previous result [18] used results from multi-player games while in this paper we develop and then use results regarding multi-tape automata. Note that many traditional methods for testing from an FSM use sequences that distinguish between states, in order to check that a (prefix of a) test case takes the SUT to a correct state [1, 6, 15, 26, 28, 35]. The results in this paper and in [18] suggest that it will be difficult to adapt such techniques for distributed testing. In addition, we can represent a possible fault in the SUT by an FSM N formed by introducing the fault into the specification FSM M: the results in this paper show that it is undecidable whether there is a test case that can detect such a 'fault', and thus also whether it represents an incorrect implementation.

Since it is undecidable whether $N \sqsubseteq_s M$, we define a weaker implementation relation $\sqsubseteq_s^k$ that only considers sequences of length k or less. This is relevant when we know a bound on the length of sequences in use or we know that the system will be reset after at most k inputs have been received. For example, a protocol might have a bound on the number of steps that can occur before a 'disconnect' happens. Naturally, it is decidable whether $N \sqsubseteq_s^k M$ since we only have to reason about finite sets of global traces. We prove that if we place a bound on k and the number of ports then we can decide whether $N \sqsubseteq_s^k M$ in polynomial time but the problem is NP-hard if we do not have such bounds.

This paper is structured as follows. Section 2 provides preliminary definitions. Section 3 then investigates the problem of defining a language $\tilde{\mathcal{L}}(M)$ such that for every multi-port FSM N we have that $N \sqsubseteq_s M$ if and only if all global traces of N are in $\tilde{\mathcal{L}}(M)$. In Section 4 we prove results regarding multi-tape automata that we use in

[2]It is decidable whether there is a strategy that is capable of reaching a given state of an FSM.

Section 5 to prove that it is generally undecidable whether $N \sqsubseteq_s M$. Section 5 also gives conditions under which $N \sqsubseteq_s M$ is decidable. In Section 6 we then explore $\sqsubseteq_s^k$. Finally, in Section 7 we draw conclusions and discuss possible lines of future work.

2 Preliminaries

This paper concerns the testing of state-based systems whose behaviour is characterised by the input/output sequences (global traces) that they can produce. Given a set A we let A^* denote the set of sequences formed from elements of A and we let ϵ denote the empty sequence. In addition, A^+ denotes the set of non-empty sequences in A^*. Given sequence $\sigma \in A^*$ we let $pref(\sigma)$ denote the set of prefixes of σ. We are interested in finite state machines, which define global traces (input/output sequences). Given a global trace $\sigma = x_1/y_1 \dots x_k/y_k$, in which $x_1, \dots, x_k$ are inputs and $y_1, \dots, y_k$ are outputs, the prefixes of σ are the global traces of the form $x_1/y_1 \dots x_j/y_j$ with $j \leq k$.

In this paper we investigate the situation in which a system interacts with its environment at n physically distributed interfaces, called ports. We let $\mathcal{P} = \{1, \dots, n\}$ denote the names of these ports. Then a multi-port FSM M is defined by a tuple (S, s_0, I, O, h) in which S is the finite set of states, $s_0 \in S$ is the initial state, I is the finite input alphabet, O is the finite output alphabet, and h is the transition relation. The set of inputs is partitioned into subsets $I_1, \dots, I_n$ such that for $p \in \mathcal{P}$ we have that I_p is the set of inputs that can be received at port p. Similarly, for port p we let O_p denote the set of output that can be observed at p. As is usual [9, 10, 32, 36] we allow an input to lead to outputs at several ports and so we let $O = ((O_1 \cup \{-\}) \times \dots \times (O_n \cup \{-\}))$ in which $-$ denotes null output. We can ensure that the I_p and also the O_p are pairwise disjoint by labelling input and output with the port name, where necessary. We let $\mathcal{A}ct = I \cup O$ denote the set of possible observations and for $p \in \mathcal{P}$ we let $\mathcal{A}ct_p = I_p \cup O_p$ denote the set of possible observations at port p.

The transition relation h is of type $S \times I \leftrightarrow S \times O$ and should be interpreted in the following way: if $(s', y) \in h(s, x)$, $y = (z_1, \dots, z_n)$, and M receives input x when in state s then it can move to state s' and send output z_p to port p (all $p \in \mathcal{P}$). This defines the *transition* $(s, s', x/y)$, which is a *self-loop transition* if $s = s'$. Since we only consider multi-port FSMs in this paper, we simply call them FSMs. The FSM M is said to be a *deterministic FSM (DFSM)* if $|h(s, x)| \leq 1$ for all $s \in S$ and $x \in I$.

An FSM M is completely-specified if for every state s and input x, we have that $h(s, x) \neq \emptyset$. A sequence $(s_1, s_2, x_1/y_1)(s_2, s_3, x_2/y_2) \dots (s_k, s_{k+1}k, x_k/y_k)$ of consecutive transitions is said to be a *path*, which has *starting state* s_1 and *ending state* s_{k+1}. This path has *label* $x_1/y_1 \dots x_k/y_k$, which is called a (global) *trace*. Further, $x_1 \dots x_k$ and $y_1 \dots y_k$ are said to be the *input portion* and the *output portion* respectively of $x_1/y_1 \dots x_k/y_k$. A path is a *cycle* if its starting and ending states are the same. The FSM M defines the regular language $L(M)$ of the labels of paths of M that have starting state s_0. Given state $s \in S$ of M we let $L_M(s)$ denote the set of global traces that are labels of paths of M with starting state s, and so $L(M) = L_M(s_0)$. We say that M is *initially connected* if for every state s of M there is a path that has starting state s_0 and ending state s. Throughout this paper we assume that any FSM considered is completely-specified and initially connected. Where this condition does not hold we can remove the states that cannot be reached and we can complete the FSM by, for example, either adding self-loop transitions with null output or transitions to an error state.

At times we will use results regarding finite automata (FA) and so we briefly define FA here. A FA M is defined by a tuple (S, s_0, X, h, F) in which S is the finite set of states, $s_0 \in S$ is the initial state, X is the finite alphabet, h is the transition relation, and $F \subseteq S$ is the set of final states. The transition relation has type $S \times (X \cup \{\tau\}) \times S$ where τ represents a silent transition that is not observed. The notions of a path and its label, which does not include instances of τ, correspond to those defined for FSMs and so are not defined here. The FA M defines the language $L(M)$ of labels of paths that have starting state s_0 and an ending state in F.

For a global trace σ and port $p \in \mathcal{P}$ let $\pi_p(\sigma)$ denote the *local trace* formed by removing from σ all elements that do not occur at p. This is defined by the following rules in which σ is a global trace (see, for example, [22]).

Figure 4. Finite State Machines M_1 and N_1

$$\begin{aligned}
\pi_p(\epsilon) &= \epsilon \\
\pi_p((x/(z_1,\ldots,z_n))\sigma) &= \pi_p(\sigma) \text{ if } x \notin I_p \wedge z_p = - \\
\pi_p((x/(z_1,\ldots,z_n))\sigma) &= x\pi_p(\sigma) \text{ if } x \in I_p \wedge z_p = - \\
\pi_p((x/(z_1,\ldots,z_n))\sigma) &= z_p\pi_p(\sigma) \text{ if } x \notin X_p \wedge z_p \neq - \\
\pi_p((x/(z_1,\ldots,z_n))\sigma) &= xz_p\pi_p(\sigma) \text{ if } x \in X_p \wedge z_p \neq -
\end{aligned}$$

Given a set A of global traces and port p we let $\pi_p(A)$ denote the set of projections of sequences in A. Thus, $\pi_p(A) = \{\pi_p(\sigma)|\sigma \in A\}$.

In the distributed test architecture, a local tester at port $p \in \mathcal{P}$ only observes events from $\mathcal{A}ct_p$. Thus, two global traces σ and σ' are indistinguishable if they have the same projections at every port and we denote this $\sigma \sim \sigma'$. More formally, we say that $\sigma \sim \sigma'$ if for all $p \in \mathcal{P}$ we have that $\pi_p(\sigma) = \pi_p(\sigma')$.

Given an FSM M, we let $\mathcal{L}(M)$ denote the set of global sequences that are equivalent to elements of $L(M)$ under $\sim$. These are the sequences that are indistinguishable from sequences in $L(M)$ when distributed observations are made. Previous work has defined two conformance relations for testing from an FSM that reflect the observational power of distributed testing [16]. In some situations the agents at the separate ports of the SUT will never interact with one another or share information with other agents that can interact. In such cases it is sufficient for the local trace observed at a port p to be a local trace of M. This situation is captured by the following conformance relation.

Definition 1 *Given FSMs N and M with the same input and output alphabets and the same set of ports, $N \sqsubseteq_w M$ if for every global trace $\sigma \in L(N)$ and port p there exists some $\sigma_p \in L(M)$ such that $\pi_p(\sigma_p) = \pi_p(\sigma)$. N is then said to* weakly conform *to M.*

However, sometimes there is the potential for information from separate testers to be received by an external agent. For example, there may be a central controller that receives the observations made by each tester and thus knows the projection of the global trace at each port. This leads to the following stronger conformance relation.

Definition 2 *Given FSMs N and M with the same input and output alphabets and the same set of ports, $N \sqsubseteq_s M$ if for every global trace $\sigma \in L(N)$ there exists some $\sigma' \in L(M)$ such that $\sigma' \sim \sigma$. N is then said to* strongly conform *to M.*

It is straightforward to see that given FSMs N and M we have that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq \mathcal{L}(M)$. It is also clear that $N \sqsubseteq_s M$ implies that $N \sqsubseteq_w M$. In order to see that $\sqsubseteq_s$ is strictly stronger than $\sqsubseteq_w$ it is sufficient to consider the FSMs M_1 and N_1 shown in Figure 4. Clearly we do not have that $N_1 \sqsubseteq_s M_1$ since M_1 has no global trace equivalent to $x_1/(y_1, y'_2)$ under $\sim$. However, for every global trace σ of N_1 and port p there is a global trace σ' of M_1 such that $\pi_p(\sigma) = \pi_p(\sigma')$. Thus, we have that $N_1 \sqsubseteq_w M_1$.

3 Test models for conformance

In this section we investigate the problem of defining a language $\tilde{\mathcal{L}}(M)$ for an FSM M such that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq \tilde{\mathcal{L}}(M)$. The motivation is that if we are developing the SUT from M and we do have some

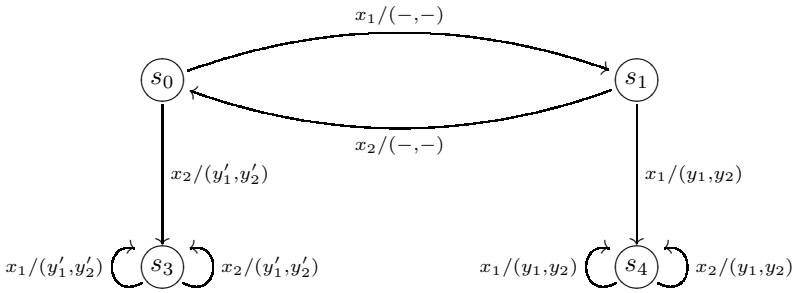

Figure 5. DFSM M_4

such $\tilde{\mathcal{L}}(M)$ then we can use standard approaches to refine $\tilde{\mathcal{L}}(M)$. In addition, if in testing we wish to test for $\sqsubseteq_s$ but we can connect the local testers to form a global tester then we should compare the global traces of the SUT with $\tilde{\mathcal{L}}(M)$: if we compare the global traces from the SUT with $L(M)$ then we could lead to the SUT N being declared faulty even if $N \sqsubseteq_s M$. It would be particularly useful if we could find an FSM or IOTS M' such that $L(M') = \tilde{\mathcal{L}}(M)$; we could then test N against M' using normal test methods and the usual conformance relation (trace inclusion).

We start by considering the language $\mathcal{L}(M)$. Clearly, we have that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq \tilde{\mathcal{L}}(M)$. However, if we can represent $\mathcal{L}(M)$ using an FSM or IOTS then for every $\sigma \in \mathcal{L}(M)$ and $\sigma' \in pre(\sigma)$ we must have that $\sigma' \in \mathcal{L}(M)$: $\mathcal{L}(M)$ must be prefix closed.

Proposition 1 *The language $\mathcal{L}(M)$ need not be prefix closed.*

Proof
Consider a DFSM M such that $x_1/(y_1, y_2)x_1/(y_1, -) \in L(M)$. Then $\mathcal{L}(M)$ contains $x_1/(y_1, -)x_1/(y_1, y_2)$ but $x_1/(y_1, -) \notin \mathcal{L}(M)$ and so $\mathcal{L}(M)$ is not prefix closed. □

The languages defined by FSMs and IOTSs are prefix closed and so we know that $\mathcal{L}(M)$ cannot always be represented by such a model. However, the languages defined by finite automata need not be prefix closed. Thus, Proposition 1 does not preclude the possibility of representing the $\mathcal{L}(M)$ using finite automata, however, the following does. It is already known from Mazurkiewicz trace theory that, where some elements of an alphabet commute (i.e. $ab = ba$) the set of sequences equivalent to those defined by a FA need not be regular (see, for example, [8]). It is straightforward to show that this also holds for FSMs.

Proposition 2 *Given FSM M, the language $\mathcal{L}(M)$ need not be regular.*

Proof
Consider the FSM M_4 shown in Figure 5 and let $L = \mathcal{L}(M_4)$. Proof by contradiction: assume that L is a regular language.

Let L' be the set of global traces in which all outputs are $(-, -)$. Clearly L' is a regular language. Thus, since L is regular we must have that $L'' = L \cap L'$ is regular. The language L'' is the set of global traces from $\mathcal{L}(M_4)$ that have null output. Thus, L'' is the set of all global traces with inputs drawn from $\{x_1, x_2\}$ and that have null output and in which the number of instances of x_2 is either equal to the number of instances of x_1 or is one less than this. However, this language is not regular, providing a contradiction as required. □

We have that $\mathcal{L}(M)$ has the property we want ($N \sqsubseteq_s M$ if and only if $L(N) \subseteq \mathcal{L}(M)$) but that $\mathcal{L}(M)$ need not be regular. The observation that $\mathcal{L}(M)$ is not prefix closed tells us that it can contain strings that are in no $L(N)$ for an FSM or IOTS N such that $N \sqsubseteq_s M$. It seems natural to remove these global traces.

Definition 3 *Given FSM M let $\mathcal{L}_{pc}(M)$ denote the set of global traces from $\mathcal{L}(M)$ whose prefixes are also in $\mathcal{L}(M)$. More formally, $\mathcal{L}_{pc}(M) = \{\sigma \in \mathcal{L}(M) | pre(\sigma) \subseteq \mathcal{L}(M)\}$.*

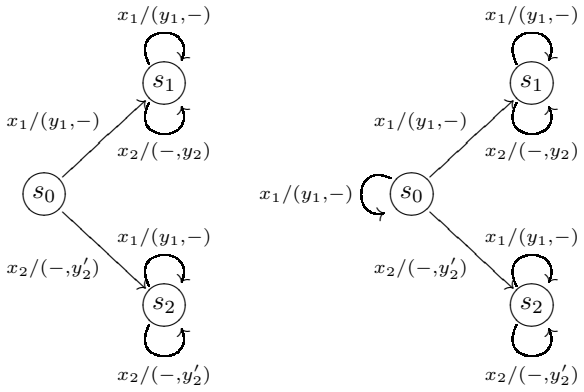

Figure 6. DFSMs M and M'

Proposition 3 *Given FSMs M and N we have that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq \mathcal{L}_{pc}(M)$.*

Proof
In forming $\mathcal{L}_{pc}(M)$ we only remove from $\mathcal{L}(M)$ a global trace σ if some of its prefixes are not in $\mathcal{L}(M)$, and so σ cannot be a global trace of an FSM N such that $N \sqsubseteq_s M$. The result therefore follows from the fact that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq \mathcal{L}(M)$. □

Clearly, $\mathcal{L}_{pc}(M) \subseteq \mathcal{L}(M)$ and so it is natural to ask whether there remain any global traces in $\mathcal{L}_{pc}(M)$ that cannot appear in FSMs that conform to M under $\sqsubseteq_s$.

Proposition 4 *Given an FSM M and global trace $\sigma \in \mathcal{L}_{pc}(M)$ there exists an FSM N such that $N \sqsubseteq_s M$ and $\sigma \in L(N)$. Further, this is true even if we restrict N to being deterministic.*

Proof
Let σ be a global trace of length k and so $\sigma = x_1/y_1 \dots x_k/y_k$ for some $x_1, \dots, x_k \in I$ and $y_1, \dots, y_k \in O$. For all $1 \leq i \leq k$ let σ_i denote the prefix of σ with length i. We will construct an FSM N' in the following way. First, define an initial state s'_0 and for every $1 \leq i < k$ we define a state s'_i and add the transition $(s'_{i-1}, s'_i, x_i/y_i)$.

Since $\sigma \in \mathcal{L}_{pc}(M)$, for each state s'_i, $1 \leq i \leq k$, we choose a (not necessarily unique) state of M, which we call s_i, that is reached from the initial state of M using a global trace $\sigma'_i \sim \sigma_i$. We add a copy of M to the structure already defined and will use transitions to the states of this copy of M in order to complete N'.

First, we add the transition $(s'_{k-1}, s_k, x_k/y_k)$ so if we follow σ by further input in N' then we will obtain σ followed by a global trace σ' from s_k in M. For all $1 \leq i < k$, $x \in I \setminus \{x_i\}$, and $(s', y) \in h(s_{i-1}, x)$, we add the transition $(s'_{i-1}, s', x/y)$. Every global trace in N' is either a global trace of M or is $\sigma_i\sigma'$ for a σ_i (which is in $\mathcal{L}(M)$) and a global trace σ' such that $\sigma' \in L_M(s_i)$ and so $\sigma \in \mathcal{L}(M)$. Further, it is clear that there is some DFSM N such that $L(N) \subseteq L(N')$ and $\sigma \in L(N)$. Thus, N is a DFSM with $L(N) \subseteq L(N') \subseteq \mathcal{L}(M)$ and so we have that $N' \sqsubseteq_s M$ as required. □

Thus, $\mathcal{L}_{pc}(M)$ is the smallest language such that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq \mathcal{L}_{pc}(M)$ and so it appears to be the language we want.

Figure 6 shows two FSMs M and M' such that $M' = \mathcal{L}_{pc}(M)$. Since $M' = \mathcal{L}_{pc}(M)$, for an FSM N we have that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq L(M')$. This works because the only global traces that are in $\mathcal{L}(M)$ but not $L(M)$ are those in the form $(x_1/(y_1,-))^* x_2/(-,y'_2)((x_1/(y_1,-)) + x_2/(-,y'_2))^*$ and these are included in $L(M')$.

We have shown that we are required to keep all of the sequences in $\mathcal{L}_{pc}(M)$. Now we show that if $L = L(M')$ for some FSM M' and we have that $\mathcal{L}_{pc}(M) \subset L(M')$ then the language $L(M')$ is too large. We do this by proving that there can be a reduction N of M' that does not conform to M under $\sqsubseteq_s$ and this is the case even if we restrict N to be deterministic.

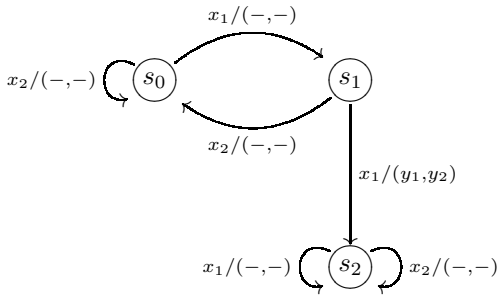

Figure 7. DFSM M_5

Proposition 5 *Given an FSM M, if $L' = L(M')$ for some FSM M' and $\mathcal{L}_{pc}(M) \subset L'$ then there is an FSM N such that $L(N) \subseteq L'$ but we do not have that $N \sqsubseteq_s M$. Further, this result holds even if we restrict N to being deterministic.*

Proof
Since $\mathcal{L}_{pc}(M) \subset L'$ there is some global trace $\sigma \in L(M') \setminus \mathcal{L}_{pc}(M)$. Since $L' = L(M')$ for an FSM M', it is clear that there exists a DFSM N such that $L(N) \subseteq L'$ and $\sigma \in L(N)$.

It is now sufficient to observe that σ and all of its prefixes are in $L(N)$ and since $\sigma \notin \mathcal{L}_{pc}(M)$ we must have that at least one of these sequences is not in $\mathcal{L}(M)$. □

Thus, the language we are looking for must contain $\mathcal{L}_{pc}(M)$ and if we restrict attention to languages defined by FSMs, then the language cannot contain any additional global traces[3]. Unfortunately, however, $\mathcal{L}_{pc}(M)$ need not be regular.

Proposition 6 *The language $\mathcal{L}_{pc}(M)$ need not be regular.*

Proof
We will use the FSM M_5 shown in Figure 7. Proof by contradiction: assume that $\mathcal{L}_{pc}(M_5)$ is regular.

Let L' denote the regular language $(x_2/(-,-))^*(x_1/(-,-))^*$. Consider the language $\mathcal{L}(M_5) \cap L'$, which is the set of sequences of the form $(x_2/(-,-))^*(x_1/(-,-))^*$ where the number of instances of x_2 can be at worst one less than the number of instances of x_1. This is prefix closed since each element of $\mathcal{L}(M_5) \cap L'$ starts with all of its instances of $x_2/(-,-)$. Clearly $\mathcal{L}(M_5) \cap L'$ is not regular.

Since $\mathcal{L}(M_5) \cap L'$ is prefix closed, $\mathcal{L}_{pc}(M_5) \cap L' = \mathcal{L}(M_5) \cap L'$. As a result, we know that $\mathcal{L}_{pc}(M_5) \cap L'$ is not regular. But L' is regular and so if $\mathcal{L}_{pc}(M_5)$ was regular then $\mathcal{L}_{pc}(M_5) \cap L'$ would also be regular. This provides a contradiction as required. □

We therefore obtain the following result.

Theorem 1 *Given an FSM M there need not exist an FSM M' with the property that for all FSMs N we have that $N \sqsubseteq_s M$ if and only if $L(N) \subseteq L(M')$. In addition, this result holds even if we restrict attention to deterministic N.*

Proof
By Propositions 4 and 5 we know that we must have that $L(M') = \mathcal{L}_{pc}(M)$. The result thus follows from Proposition 6. □

If $\mathcal{L}_{pc}(M)$ is regular then there is a corresponding FSM. It would thus be interesting to explore conditions under which $\mathcal{L}_{pc}(M)$ is guaranteed to be regular and also properties of $\mathcal{L}_{pc}(M)$ when this is not regular. There may also be scope to represent $\mathcal{L}_{pc}(M)$ using an IOTS.

[3]We can easily extend the proofs to more general formalisms such as IOTSs.

4 Conformance and multi-tape automata

While we can decide (in polynomial time) whether $N \sqsubseteq_w M$, this is quite a weak conformance relation since it does not allow us to bring together local traces observed at the separate ports. It seems likely that normally the implementation relation $\sqsubseteq_s$ will be more suitable and so we consider the problem of deciding whether $N \sqsubseteq_s M$. In this section we study the problem of deciding language inclusion for multi-tape automata; in Section 5 we use the results proved here regarding multi-tape automata to show that it is generally undecidable whether $N \sqsubseteq_s M$ for FSMs M and N. We first define multi-tape FA [30].

Definition 4 *An r-tape FA with disjoint alphabets Σ_i, $1 \leq i \leq r$ is a tuple (S, s_0, Σ, h, F) in which S is a finite set of states, $s_0 \in S$ is the initial state $F \subseteq S$ is the set of final states and $h : S \times \bigcup_{i=1}^{r} \Sigma_i \times S$ is the transition relation. Then A accepts a tuple $(w_1, \ldots, w_r) \in \Sigma_1^* \times \ldots \times \Sigma_r^*$ if and only if there is some sequence $\sigma \in (\bigcup_{i=1}^{r} \Sigma_i)^*$ that takes A to a final state such that $\pi_i(\sigma) = w_i$ for all $1 \leq i \leq r$. We let $\mathcal{L}(N)$ denote the set of tuples accepted by N.*

Given a multi-tape FA N with r tapes, we will let $L(N)$ denote the language in $(\Sigma_1 \cup \ldots \cup \Sigma_r)^*$ of the corresponding FA. We obtain $L(N)$ by treating N as a FA with alphabet $\Sigma = \Sigma_1 \cup \ldots \cup \Sigma_r$.

It might seem that deciding whether $N \sqsubseteq_s M$ is similar to deciding whether, for two multi-tape FA N' and M', the language defined by N' is a subset of that defined by M'. This problem, regarding multi-tape FA, is known to be undecidable [30]. However, the proof of the result regarding multi-tape FA uses FA in which not all states are final and in FSMs there is no concept of a state not being a final state. Thus, the results of [30] are not directly applicable to the problem of deciding whether $N \sqsubseteq_s M$ for FSMs N and M and it appears that the corresponding problem, for multi-tape FA in which all states are final, has not previously been solved. In this section we prove that language inclusion is generally undecidable for multi-tape FA in which all states are final states. Before we consider decidability issues, we investigate the corresponding languages and closure properties.

Proposition 7 *Let us suppose that $N_1 = (S, s_0, h_1, S)$ and $N_2 = (Q, q_0, h_2, Q)$ are multi-tape FA with the same number of tapes and the same alphabets and also that all of the states of N_1 and N_2 are final states. Then we have the following.*

1. *There exists a multi-tape FA M such that $\mathcal{L}(M) = \mathcal{L}(N_1) \cup \mathcal{L}(N_2)$ and all of the states of M are final states.*
2. *There exists a multi-tape FA M such that $\mathcal{L}(M) = \mathcal{L}(N_1)\mathcal{L}(N_2)$ and all of the states of M are final states.*
3. *There may not exists a multi-tape FA M such that $\mathcal{L}(M) = \mathcal{L}(N_1) \setminus \mathcal{L}(N_2)$ and all of the states of M are final states.*
4. *There may not exists a multi-tape FA M such that $\mathcal{L}(M) = \mathcal{L}(N_1) \cap \mathcal{L}(N_2)$ and all of the states of M are final states.*

Proof
We will use $A \oplus B$, for sets A and B, to denote the disjoint union of A and B. For the first result it is sufficient to define the FA $(S \oplus Q \oplus \{r_0\}, r_0, h', S \oplus Q \oplus \{r_0\})$ for $r_0 \notin S \oplus Q$, in which h is the union of h_1 and h_2 plus the following transitions: for every $(s_0, a, s) \in h_1$ we include in h' the tuple (r_0, a, s); and for every $(q_0, a, q) \in h_2$ we include in h' the tuple (r_0, a, q).

For the second part, it is sufficient to take the disjoint union of N_1 and N_2 and for every transition (s, s', a) of N_1 add a transition (s, q_0, a).

For the third part, it is sufficient to observe that for any choice of N_1 and N_2 we have that the empty sequence is in $\mathcal{L}(N_1)$ and $\mathcal{L}(N_2)$ and so not in $\mathcal{L}(N_1) \setminus \mathcal{L}(N_2)$. Thus, it is sufficient to choose any N_1 and N_2 such that $\mathcal{L}(N_1) \setminus \mathcal{L}(N_2)$ is non-empty.

For the last part, let us suppose that we have two tapes with alphabets $\{a_1\}$ and $\{a_2\}$, let $L(N_1) = \{\epsilon, a_1, a_1a_2\}$ and let $L(N_2) = \{\epsilon, a_2, a_2a_1\}$ and so $\mathcal{L}(N_1) \cap \mathcal{L}(N_2) = \{\epsilon, a_1a_2, a_2a_1\}$. □

We will use Post's Correspondence Problem to prove that language inclusion is undecidable.

Definition 5 *Given sequences $\alpha_1, \ldots, \alpha_m$ and $\beta_1, \ldots, \beta_m$ over an alphabet Σ, Post's Correspondence Problem (PCP) is to decide whether there is a sequence $i_1, \ldots, i_k$ of indices from $[1, m]$ such that $\alpha_{i_1} \ldots \alpha_{i_k} = \beta_{i_1} \ldots \beta_{i_k}$.*

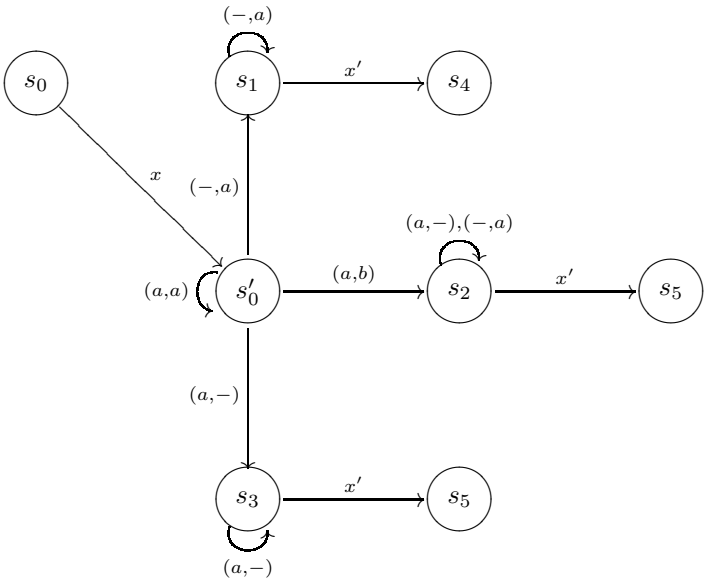

Figure 8. Finite Automaton M

It is known that Post's Correspondence Problem is undecidable [29].

Theorem 2 *Post's Correspondence Problem is undecidable.*

We now prove the main result from this section.

Theorem 3 *Let us suppose that N and M are multi-tape FA in which all states are final states. The following problem is undecidable, even when there are only two tapes: do we have that $\mathcal{L}(N) \subseteq \mathcal{L}(M)$?*

Proof
We will show that if we can solve this problem then we can also solve Post's Correspondence Problem. We therefore assume that we have been given an instance of the PCP with sequences $\alpha_1, \ldots, \alpha_m$ and $\beta_1, \ldots, \beta_m$ with alphabet Σ. To allow elements of Σ to be on both tapes we use two disjoint copies, $\Sigma_1 = \{f_1(a)|a \in \Sigma\}$ and $\Sigma_2 = \{f_2(a)|a \in \Sigma\}$, of Σ. Given a sequence $x_1 \ldots x_i$ and $j \in \{1, 2\}$ we let $f_j(x_1 \ldots x_i)$ denote $f_j(x_1) \ldots f_j(x_i)$.

We consider multi-tape automata with two tapes with alphabets $\Sigma_1 \cup \{x, x'\}$ and Σ_2 respectively, where x, x' are chosen so that they are not in $\Sigma_1 \cup \Sigma_2$. We let N denote an FA such that $L(N)$ is the language that contains all sequences in the regular language $x((f_1(\alpha_{i_1})f_2(\beta_{i_1}) + \ldots + (f_1(\alpha_{i_k})f_2(\beta_{i_k}))^* x'$ and all prefixes of such sequences. Clearly, such an FA N exists. Consider all sequences in $\mathcal{L}(N)$ that contain an x and an x' and let σ_a and σ_b be the sequences in the two tapes with x and x' removed. Then we must have that σ_a is of the form $f_1(\alpha_{i_1}) \ldots f_1(\alpha_{i_k})$ and σ_b is of the form $f_2(\beta_{i_1}) \ldots f_2(\beta_{i_k})$ with $i_1, \ldots, i_k \in [1, m]$. In addition, all such combinations correspond to sequences in $L(N)$. Thus, there is a solution to this instance of the PCP if and only if $\mathcal{L}(N)$ contains a tuple $(x\sigma_a x', \sigma_b)$ in which $\sigma_a = \sigma_b$.

The FA M that defines language $L(M)$ is shown in Figure 8 in which $(a, -)$ denotes elements all of the form $f_1(a)$ and $(-, a)$ denotes all elements of the form $f_2(a)$, $a \in \Sigma$. Further, (a, b) denotes sequences of the form $f_1(a)f_2(b)$ with $a, b \in \Sigma$ and $a \neq b$. We use (a, a) to denote sequences of the form $f_1(a)f_2(a)$ with $a \in \Sigma$. In all cases where we have sequences with length 2, this represents a cycle of length 2.

Now consider the problem of deciding whether $\mathcal{L}(N) \subseteq \mathcal{L}(M)$. Specifically, we will focus on the problem of deciding whether $\mathcal{L}(N)$ is not a subset of $\mathcal{L}(M)$ and so $\mathcal{L}(N) \setminus \mathcal{L}(M)$ is non-empty. First, note that in $\mathcal{L}(M)$ we have:

1. The language defined by paths that pass through s_1 contains all tuples of the form $(xf_1(w_1), f_2(w_2))$ or $(xf_1(w_1)x', f_2(w_2))$ such that $w_1, w_2 \in \Sigma^*$ and w_1 is a strict prefix of w_2.

2. The language defined by paths that pass through s_3 contains all tuples of the form $(xf_1(w_1), f_2(w_2))$ or $(xf_1(w_1)x', f_2(w_2))$ such that $w_1, w_2 \in \Sigma^*$ and w_2 is a strict prefix of w_1.

3. The language defined by paths that pass through s_2 contains all tuples of the form $(xf_1(w_1w_3), f_2(w_2w_4))$ or $(xf_1(w_1w_3)x', f_2(w_2w_4))$ in which $w_1 \in \Sigma^*$ and $w_2 \in \Sigma^*$ have the same length, $w_1 \neq w_2$, and $w_3, w_4 \in \Sigma^*$.

4. The language defined by paths that do not leave s_0 contains all tuples of the form $(xf_1(w), f_2(w))$.

Consider tuples in $\mathcal{L}(M)$ that do not contain x' and thus tuples of the form $(xf_1(w_1), f_2(w_2))$ in which $w_1, w_2 \in \Sigma^*$. Then paths that pass through s_1 define all such tuples in which w_1 is a strict prefix of w_2 and paths that pass through s_3 define all such tuples in which w_2 is a strict prefix of w_1. In addition, paths that do not leave s_0 define all such tuples in which $w_1 = w_2$ and paths that pass through s_2 define all such tuples in which w_1 and w_2 differ after a (possibly empty) common prefix. Thus, $\mathcal{L}(M)$ defines all tuples of the form $(xf_1(w_1), f_2(w_2))$ in which $w_1, w_2 \in \Sigma^*$ and so all tuples in $\mathcal{L}(N)$ that do not contain x' are also in $\mathcal{L}(M)$.

Now, consider the tuples in $\mathcal{L}(M)$ that contain x'. These are of the form $(xf_1(w_1)x', f_2(w_2))$ and are defined by paths that pass through s_1, s_2 and s_3. By examining the languages defined by paths that pass through these states we find that $\mathcal{L}(M)$ contains the set of tuples of this form in which $w_1 \neq w_2$. Thus, $\mathcal{L}(N) \setminus \mathcal{L}(M)$ is non-empty if and only if $\mathcal{L}(N)$ contains a tuple of the form $(xf_1(w)x', f_2(w))$. But we know that this is the case if and only if there is a solution to this instance of the PCP and so the result follows from Theorem 2. □

For the sake of completeness, we now prove some additional decidability results regarding multi-tape automata in which all states are final.

Theorem 4 *Let us suppose that N and M are multi-tape FA in which all states are final states. The following problem is undecidable, even when there are only three tapes: do we have that $\mathcal{L}(N) \cap \mathcal{L}(M)$ contains a non-empty sequence.*

Proof
We assume that we have been given an instance of the PCP with sequences $\alpha_1, \ldots, \alpha_m$ and $\beta_1, \ldots, \beta_m$ with alphabet Σ and follow an approach similar to that used in the proof of Theorem 3. To allow elements of Σ on two tapes we use two copies of elements of Σ and let $\Sigma_1 = \{f_1(a)|a \in \Sigma\} \cup \{x\}$, $\Sigma_2 = \{f_2(a)|a \in \Sigma\}$, and $\Sigma_3 = \{x'\}$. Given a sequence $a_1, \ldots, a_i \in \Sigma^*$ and $j \in \{1, 2\}$ we let $f_j(a_1 \ldots a_i)$ denote $f_j(a_1) \ldots f_j(a_i)$.

We consider multi-tape automata with three tapes with alphabets Σ_1, Σ_2, and Σ_3. We let N denote such an FA such that $L(N)$ contains all tuples formed by sequences in the regular language $x((f_1(\alpha_{i_1})f_2(\beta_{i_1}) + \ldots + (f_1(\alpha_{i_k})f_2(\beta_{i_k}))^+x'$ and all prefixes of such sequences. Note that xx' is not contained in $L(N)$. In addition, we let M denote the multi-tape automaton defined by the following:

1. There is a transition from s_0 to state s_1 and this has label x';

2. For all $a \in \Sigma$ there is a cycle starting and ending at s_1 with label $f_1(a)f_2(a)$;

3. There is a transition from s_1 to s_2, not involved in the cycles, with label x.

4. There are no transition from s_2.

Thus, all elements of $\mathcal{L}(M)$ are either $(\epsilon, \epsilon, \epsilon)$ or contain x'. As a result, if there is a non-empty element of $\mathcal{L}(N) \cap \mathcal{L}(M)$ then this must contain both x and x'. It is now sufficient to observe that this is the case if and only if there is a solution to this instance of the PCP. □

Finally, we prove that equivalence is undecidable for multi-tape FA in which all states are final states.

Theorem 5 *Let us suppose that N and M are multi-tape FA in which all states are final states. The following problem is undecidable, even when there are only two tapes: do we have that $\mathcal{L}(N) = \mathcal{L}(M)$?*

Proof
First observe that given sets A and B we have that $A \subseteq B$ if and only if $A \cup B = B$. Let us suppose that we have two multi-tape automata N_1 and N_2 with the same numbers of tapes and the same alphabets and assume that all states of N_1 and N_2 are final states. By Proposition 7 we know that we can construct a multi-tape automaton N_3 such that $\mathcal{L}(N_3) = \mathcal{L}(N_1) \cup \mathcal{L}(N_2)$ and all states of N_3 are final states. Thus, if we can decide whether $\mathcal{L}(N) = \mathcal{L}(M)$ for two multi-tape FA that have only final states then we can also decide whether $\mathcal{L}(N_1) \cup \mathcal{L}(N_2) = \mathcal{L}(N_2)$ for two multi-tape FA that only have final states. However, this holds if and only if $\mathcal{L}(N_1) \subseteq \mathcal{L}(N_2)$. The result thus follows from Theorem 3. □

We now show how we can represent the problem of deciding language inclusion for multi-tape FA in terms of deciding whether $N \sqsubseteq_s M$ for suitable FSMs N and M.

5 Deciding Strong Conformance

We have proved some decidability results for multi-tape FA in which all states are final states. However, we are interested in FSMs and here a transition has an input/output pair as a label. We now show how a multi-tape FA can be represented using an FSM (with multiple ports) before using this result to prove that $N \sqsubseteq_s M$ is generally undecidable for FSMs N and M.

In order to extend Theorem 3 to FSMs we define a function that takes a multi-port finite automaton and returns an FSM.

Definition 6 *Let us suppose that $N = (S, s_0, \Sigma, h, S)$ is a FA with r tapes with alphabets $\Sigma_1, \ldots, \Sigma_r$. We define the FSM $\mathcal{F}(N)$ with $r+1$ ports as defined below in which for all $1 \leq p \leq r$ we have that the input alphabet of N at p is Σ_p and the output alphabet is empty and further we have that the input alphabet at port $r+1$ is empty and the output alphabet at $r+1$ is $\{0,1\}$. In the following for $a \in \{0,1\}$ we use a_k to denote the k-tuple whose first $k-1$ elements are empty and whose kth element is a.*

$\mathcal{F}(N) = (S \cup \{s_e\}, s_0, \Sigma, \{0_{n+1}, 1_{n+1}\}, h')$ in which $s_e \notin S$, for all $z \in \Sigma$ we have that $h'(s_e, z) = \{(s_e, 0_{r+1})\}$ and for all $s \in S$ and $z \in \Sigma$ we have that $h'(s,z)$ is defined by the following:

1. *If $h(s,z) = S' \neq \emptyset$ then $h'(s,z) = \{(s', 1_{r+1}), (s', 0_{r+1}) | s' \in S'\}$;*
2. *If $h(s,z) = \emptyset$ then $h'(s,z) = \{(s_e, 0_{r+1})\}$.*

The idea is that while following a path of N the FSM $\mathcal{F}(N)$ can produce either 0 or 1 at port $r+1$ in response to each input but once we diverge from such a path the FSM can then only produce 0 (at $r+1$) in response to an input.

Lemma 1 *Let us suppose that N and M are r-tape FA with alphabets $\Sigma_1, \ldots, \Sigma_r$. Then $\mathcal{L}(N) \subseteq \mathcal{L}(M)$ if and only if $\mathcal{F}(N) \sqsubseteq_s \mathcal{F}(M)$.*

Proof
First assume that $\mathcal{F}(N) \sqsubseteq_s \mathcal{F}(M)$ and we are required to prove that $\mathcal{L}(N) \subseteq \mathcal{L}(M)$. Assume that $\sigma \in \mathcal{L}(N)$ and so there exists some $\sigma' \sim \sigma$ such that $\sigma' \in L(N)$. Since $\sigma' \in L(N)$ we have that $L(\mathcal{F}(N))$ contains the global trace ρ' in which the input portion is σ' and each output is 1_{r+1}. Since $\mathcal{F}(N) \sqsubseteq_s \mathcal{F}(M)$ we must have that there is some $\rho'' \in L(\mathcal{F}(M))$ such that $\rho'' \sim \rho'$. However, since the outputs are all at port $r+1$ and the inputs are at ports $1, \ldots, r$ we must have that ρ'' has output portion that contains only 1_{r+1} and input portion σ'' for some $\sigma'' \sim \sigma'$. Thus, we must have that $\sigma'' \in L(M)$. Since $\sigma \sim \sigma'$ and $\sigma' \sim \sigma''$ we must have that $\sigma \in \mathcal{L}(M)$ as required.

Now assume that $\mathcal{L}(N) \subseteq \mathcal{L}(M)$ and we are required to prove that $\mathcal{F}(N) \sqsubseteq_s \mathcal{F}(M)$. Let ρ be some element of $L(\mathcal{F}(N))$ and it is sufficient to prove that $\rho \in \mathcal{L}(\mathcal{F}(M))$. Then $\rho = \rho_1\rho_2$ for some maximal ρ_2 such that all outputs in ρ_2 are 0_{r+1}. Let the input portions of ρ_1 and ρ_2 be σ_1 and σ_2 respectively. By the maximality of ρ_2 we must have that ρ_1 is either empty or ends in output 1_{r+1}. Thus, $\sigma_1 \in L(N)$ and so, since $\mathcal{L}(N) \subseteq \mathcal{L}(M)$, there exists $\sigma_1' \sim \sigma_1$ with $\sigma_1' \in L(M)$. But this means that M can produce the output portion of ρ_1 in response to σ_1' and so there exists $\rho_1' \in L(\mathcal{F}(M))$ with $\rho_1' \sim \rho_1$. By the definition of $\mathcal{F}(M)$, since all outputs in ρ_2 are 0_{r+1} we have that $\rho' = \rho_1'\rho_2 \in L(\mathcal{F}(M))$. The result therefore follows from observing that $\rho' = \rho_1'\rho_2 \sim \rho_1\rho_2 = \rho$. □

Theorem 6 *The following problem is undecidable: given two multi-port FSMs N and M with the same alphabets, do we have that $N \sqsubseteq_s M$?*

Proof
This follows from Lemma 1 and Theorem 3. □

When considering FSM M with only one port, we represent the problem of deciding whether two states s and s' of M are equivalent by comparing the FSMs M_s and $M_{s'}$, formed by starting M in s and s' respectively. However, we also have that the general problem is undecidable.

Theorem 7 *The following problem is undecidable: given a multi-port FSM M and two states s and s' of M, are s and s' equivalent.*

Proof
We will prove that we can express the problem of deciding whether multi-port FSMs are equivalent in terms of state equivalence. We therefore assume that we have multi-port FSMs M_1 and M_2 with the same input and output alphabets and we wish to decide whether M_1 an M_2 are equivalent. Let s_{01} and s_{02} denote the initial states of M_1 and M_2 respectively. We will construct an FSM M in the following way. We add a new port p and input x_p at p. The input of x_p in the initial state s_0 of M moves M non-deterministically to either s_{01} or s_{02} and produces no output. All other input in state s_0 moves M to a state $s'_0 \neq s_0$, that is not a state of M_1 or M_2, from which all transitions are self-loops. The input of x_p in a state of M_1 or M_2 leads to no output and no change of state. Now we can observe that a sequence in the language defined by starting M in state s_{0i}, $i \in \{0, 1\}$, is equivalent under $\sim$ to a sequence from $\mathcal{L}(M_i)$ followed by a sequence of zero or more instances of x_p. Thus, s_{01} and s_{02} are equivalent if and only if M_1 and M_2 are equivalent. The result thus follows from Theorem 6 and the fact that if we can decide equivalence then we can also decide inclusion. □

We now consider problems relating to distinguishing FSMs and states in testing. We can only distinguish between FSMs and states on the basis of observations and each observation, in distributed testing, defines an equivalence class of $\sim$.

Definition 7 *It is possible to distinguish FSM N from FSM M in distributed testing if and only if $\mathcal{L}(N) \not\subseteq \mathcal{L}(M)$. Further, it is possible to distinguish between FSMs N and M in distributed testing if and only if $\mathcal{L}(N) \not\subseteq \mathcal{L}(M)$ and $\mathcal{L}(M) \not\subseteq \mathcal{L}(N)$.*

The first part of the definition says that we can only distinguish N from M in distributed testing if there is some global trace of N that is not observationally equivalent to a global trace of M. The second part strengthens this by requiring that we can distinguish N from M and also M from N. The following is an immediate consequence of Theorem 6.

Theorem 8 *The following problems are generally undecidable in distributed testing.*

- *Is it possible to distinguish FSM N from FSM M?*
- *Is it possible to distinguish between FSMs N and M?*

Similar to the proof of Theorem 7, we can express the problem of distinguishing between two FSMs as that of distinguishing two states s and s' of an FSM M. Thus, the above shows that it is undecidable whether there is some test case that is capable of distinguishing two states of an FSM or two FSMs. This complements a previous result [18], that it is undecidable whether there is some test case that is *guaranteed* to distinguish two states or FSMs.

Finally, we give conditions under which equivalence and inclusion are decidable. The first uses the notion of a Parikh Image of a sequence [27]. Given a sequence $\sigma \in \Sigma^*$, where we have ordered the elements of Σ as $a_1, \ldots, a_m$, the Parikh Image of σ is the tuple $(x_1, \ldots, x_m)$ in which for all $1 \leq i \leq m$ we have that σ contains x_i instances of a_i. Given a set A of sequences, the Parikh Image of A is the set of tuples formed by taking the Parikh Image of each sequence in A.

There are classes of languages where the Parikh Image of the language is guaranteed to be a semi-linear set. A linear set is defined by a set of vectors $v_0, \ldots, v_k$ that have the same dimension. Specifically, the linear set defined by $v_0, \ldots, v_k$ and is the set of $v_0 + n_1 v_1 + \ldots + n_k v_k$ where $n_1, \ldots, n_k$ are all non-negative integers. A semi-linear set is a finite union of linear sets.

Proposition 8 *Let us suppose that multi-port FSMs N and M have the same input and output alphabets and that for each port $p \in \mathcal{P}$ we have that $|\mathcal{A}ct_p| \leq 1$. Then it is decidable whether $N \sqsubseteq_s M$.*

Proof
Since for all $p \in \mathcal{P}$ we have that $|\mathcal{A}ct_p| \leq 1$, for each $\sigma \in \mathcal{A}ct^*$ we have that σ is equivalent under $\sim$ to all its permutations. Thus, the Parikh Image of a sequence in $L(N)$ or $L(M)$ uniquely defines the corresponding equivalence class. Thus, $N \sqsubseteq_s M$ if and only if the Parik Image of $L(N)$ is a subset of the Parikh Image of $L(M)$. However, these Parikh Images are semi-linear sets and it is decidable whether one semi-linear set is a subset of another (see, for example, [25]). The result thus follows. □

We now consider the case where each transition produces output at all ports.

Proposition 9 *Let us suppose that M is an FSM in which all transitions produce output at all ports. Then $N \sqsubseteq_s M$ if and only if $L(N) \subseteq L(M)$.*

Proof
First observe that if $N \sqsubseteq_s M$ then each transition of N must also produce output at every port. Consider a sequence $\sigma \in L(M) \cup L(N)$ that contains k inputs. Since every transition produces output at all ports, for a port p we have that $\pi_p(\sigma)$ contains k outputs with each input x_i at p being between the output produced at p by the previous input and the output produced at p in response to x_i. Thus, given sequences $\sigma, \sigma' \in L(N) \cup L(M)$ we must have that $\sigma' \sim \sigma$ if and only if $\sigma' = \sigma$. The result therefore holds. □

6 Bounded conformance

We have seen that it is undecidable whether two FSMs are related under $\sqsubseteq_s$. However, we might use a weaker notion of conformance where we only consider sequences of length at most k for some k. This would be relevant when the expected usage does not involve sequences of length greater than k since, for example, the system will be reset after at most k inputs. In this section we define such a weaker implementation relation and explore the problem of deciding whether two FSMs are related under this.

First, we introduce some notation. We let IO_k denote the set of global traces that have at most k inputs. In addition, for an FSM N we let $L_k(N) = L(N) \cap IO_k$ denote the set of global traces of N that have at most k inputs. We can now define our implementation relation.

Definition 8 *Given FSMs N and M with the same input and output alphabets, we say that N strongly k-conforms to M if for all $\sigma \in L_k(N)$ there exists some $\sigma' \in L(M)$ such that $\sigma' \sim \sigma$. If this is the case then we write $N \sqsubseteq_s^k M$.*

Clearly, given N and M it is decidable whether $N \sqsubseteq_s^k M$: we can simply generate every element of $L_k(N)$ and for each $\sigma \in L_k(N)$ we determine whether $\sigma \in \mathcal{L}(M)$. The following shows that this can be achieved in polynomial time if we have a bound on the number of ports. We use a result from Mazurkiewicz trace theory. In Mazurkiewicz trace theory an independence graph is a directed graph where each vertex of the graph represents an element of $\mathcal{A}ct$ and there is an edge between the vertex representing $a \in \mathcal{A}ct$ and the vertex representing $b \in \mathcal{A}ct$ if and only if ab and ba are equivalent; a and b are said to be independent [8]. Thus, for FSMs we have that a and b are independent if and only if they are at different ports.

Lemma 2 *Given a sequence $\sigma \in IO_k$ and FSM M with n ports, we can decide whether $\sigma \in \mathcal{L}(M)$ in time of $O(|\sigma|^n)$.*

Proof
The membership problem for a sequence σ and rational trace language with alphabet Σ and independence relation $\mathcal{I}$ can be solved in time of $O(|\sigma|^\alpha)$ where α is the size of the largest clique in the independence graph [3]. Since each observation is made at exactly one port and two observations are independent if and only if they are at different ports, we have that the maximal cliques of the independence graph all have size n and so $\alpha = n$. The result therefore follows. □

Theorem 9 *If there are bounds on the value of k and the number n of ports then the following problem can be solved in polynomial time: given FSMs N and M with at most n ports, do we have that $N \sqsubseteq_s^k M$?*

Proof
First observe that the number of elements in $L_k(N)$ is of $O(q^k)$, where q denotes the maximum number of transitions leaving a state of N. Thus, since k is bounded, the elements in $L_k(N)$ can be produced in polynomial time. It only remains to consider each σ in $L_k(N)$ and decide whether it is in $\mathcal{L}(M)$. However, by Lemma 2, this can be decided in polynomial time. The result therefore follows. □

Thus, if we have bounds on the number of ports of the system and the length of sequences we are considering then we can decide whether $N \sqsubseteq_s^k M$ in polynomial time. However, the proof of Theorem 9 introduced terms that are exponential in n and k and so it is natural to ask what happens if we do not place bounds on these values. It transpires that the problem is then NP-hard even for DFSMs, the proof using the following.

Definition 9 *Given boolean variables $z_1, \ldots, z_r$ let $C_1, \ldots, C_k$ denote sets of three literals, where each literal is either a variable z_i or its negation. The three-in-one SAT problem is: does there exist an assignment to the boolean variables such that each C_j contains exactly one true literal.*

The three-in-one SAT problem is known to be NP-complete [33].

Theorem 10 *The following problem is NP-hard: given k and completely specified DFSMs N and M, do we have that $N \sqsubseteq_s^k M$?*

Proof
We will show that we can reduce the three-in-one SAT problem to this and suppose that we have variables $z_1, \ldots, z_r$ and clauses $C_1, \ldots, C_q$. We will define a DFSM M_1 with $r + q + 2$ ports, inputs $z_{-1}, z_0, z_1, \ldots, z_r$ at ports $-1, 0, 1, \ldots, r$ and outputs $y_1, \ldots, y_{r+q}$ at ports $1, \ldots, r + q$. We count ports from -1 rather than 1 since the roles of inputs at -1 and 0 will be rather different from the roles of the other inputs.

DFSM M_1 has four states s_0, s_1, s_2, s_3 in which s_0 is the initial state. The states effectively represent different 'modes' and we now describe the roles of s_1 and s_2. In state s_1 an input at port p, $1 \leq p \leq r$, will lead to output at all of the ports corresponding to clauses with literal z_p. In state s_2 an input at port p, $1 \leq p \leq r$, will lead to output at all of the ports corresponding to clauses with literal $\neg z_p$. The input z_0 moves M_1 from s_1 to s_2. The special input z_{-1} takes M_1 from state s_0 to state s_1.

Overall, input z_0 does not produce output and only changes the state of M_1 if it is in state s_1, in which case it takes M to state s_2. Input z_{-1} does not produce output and only changes the state of M_1 if it is in state s_0, in which case it takes M_1 to state s_1.

For an input z_p with $1 \leq p \leq r$ there are four transitions:

1. From state s_1 there is a transition that, for all $1 \leq j \leq k$, sends output y_{r+j} to port $r + j$ if C_j contains literal z_p and otherwise sends no output to port $r + j$. The transition sends no output to ports $-1, \ldots, r$ and does not change state.

2. From state s_2 there is a transition that, for all $1 \leq j \leq k$, sends output y_{r+j} to port $r + j$ if C_j contains literal $\neg z_p$ and otherwise sends no output to port $r + j$. The transition sends no output to ports $-1, \ldots, r$ and does not change state.

3. From state s_0 there is a transition to state s_3 that produces no output.

4. From state s_3 there is a transition to state s_3 that produces no output.

Now consider the global trace σ that starts with input sequence $z_{-1} z_0 z_1 \ldots z_{r-1}$ and then has input z_r producing the outputs $y_{r+1} \ldots y_{r+q}$; all outputs are produced in response to the last input. Clearly we do not have $\sigma \in L(M_1)$. We now prove that $\sigma \in \mathcal{L}(M_1)$ if and only if there is a solution to the instance of the three-in-one SAT problem. Consider the problem of deciding whether there exists $\sigma' \in L(M_1)$ such that $\sigma' \sim \sigma$. Clearly the first input in σ' must be z_{-1}. Each input z_p is received once by the DFSM and these can be received in any order after z_{-1}. Thus, for all $1 \leq p \leq r$ we do not know whether z_p will be received before or after z_0 in σ'. If z_p is received before z_0 then an output is sent to all ports that correspond to clauses that contain literal z_p. If z_p is received after z_0 then an output is sent to all ports that correspond to clauses that contain literal $\neg z_p$. Thus there exists $\sigma' \in L(M_1)$ such that $\sigma' \sim \sigma$ if and only if there exists an assignment to the boolean variables $z_1, \ldots, z_r$ such that each C_j contains exactly one true literal.

We now construct DFSMs N and M such that $N \sqsubseteq_s^k M$ if and only if $\sigma \in \mathcal{L}(M_1)$. In the following we assume that $r > 1$ and let σ_1 be the global trace formed from σ by replacing the prefix $z_{-1}z_0z_1$ by $z_1z_{-1}z_0$. Thus, $\sigma_1 \sim \sigma$. We form N from M_1 by adding a new path that has label σ_1. We add state s'_3 such that the input of z_1 in state s_0 leads to state s'_3 (instead of s_3) and no output. From s'_3 we add a transition with label z_0 to another new state s'_4. We repeat this process, adding new states, until we have a path from s_0 with label $z_1z_0z_{-1}z_2z_3 \dots z_{r-1}$ ending in state s'_{r+3}. We then add a transition from s'_{r+3} to s'_{r+4} with input z_r and the outputs $y_{r+1}, \dots, y_{r+q}$. Finally, we complete N by adding a transition to s_3 with input z_p and null output from a state s'_j if there is no transition from s'_j with input z_p. Clearly, $L(N) = L(M_1) \cup pref(\sigma_1)I^*$. Let σ'_1 be defined such that $\sigma_1 = z_1\sigma'_1$. We can similarly form an FSM M from M_1 such that $L(M) = L(M_1) \cup pref(\{z_1\}I\{\sigma'_1\})I^*$. Since each I_p contains only one input we have that $\{z_1\}I\{\sigma'_1\}I^*$ and $\{\sigma_1\}I^+$ define the same sets of equivalence classes under $\sim$. Thus, the equivalence classes of $pref(\sigma_1)I^*$ and $pref(\{z_1\}I\{\sigma'_1\})I^*$ under $\sim$ differ only in the one that contains σ_1 and we know that $\sigma_1 \sim \sigma$. We therefore have that $N \sqsubseteq_s^k M$, for $k > r + 1$, if and only if $\sigma \in \mathcal{L}(M_1)$ and we know that this is the case if and only if the instance of the three-in-one SAT problem has a solution. The result follows from the three-in-one SAT problem being NP-hard. □

Naturally, the results in this section are also relevant when we are looking for tests of length no longer than k that distinguish states or FSMs.

7 Conclusions

There are important classes of systems such as cloud systems, web services and wireless sensor networks, that interact with their environment at physically distributed ports. In testing such a system we place a local tester at each port and the local tester (or user) at port p only observes the events that occur at p. It is known that this reduced observational power, under which a set of local traces is observed, can introduce additional controllability and observability problems.

This paper has considered the situation in which there is a finite state machine (FSM) model M that acts as the specification for a system that interacts with its environment at physically distributed ports. We considered the implementation relation $\sqsubseteq_s$ that requires the set of local traces observed to be consistent with some global trace of the specification. We investigated the problem of defining a language $\tilde{\mathcal{L}}(M)$ such that we know that $N \sqsubseteq_s M$ if and only if all of the global traces of N are contained in $\tilde{\mathcal{L}}(M)$. We showed that $\tilde{\mathcal{L}}(M)$ can be uniquely defined but need not be regular.

We proved that it is generally undecidable whether $N \sqsubseteq_s M$ even if there are only two ports, although we also gave conditions under which this is decidable. An additional consequence of this result is that it is undecidable whether there is a test case (a strategy for each local tester) that is *capable* of distinguishing two states of an FSM or two FSMs. This complements earlier results that show that it is undecidable whether there is a test case that is *guaranteed* to distinguish between two states of an FSM or two FSMs. While these results appear to be related the proofs relied on very different approaches: the earlier result looked at the problem in terms of multi-player games while this paper developed and used results regarding multi-tape automata.

Since it is generally undecidable whether $N \sqsubseteq_s M$ we defined a weaker implementation relation $\sqsubseteq_s^k$ under which we only consider input sequences of length k or less. This is particularly relevant in situations in which it is known that input sequences of length greater than k need not be considered since, for example, the system must be reset before this limit has been reached. We proved that if we place a bound on k and the number of ports then we can decide whether $N \sqsubseteq_s^k M$ in polynomial time but otherwise this problem is NP-hard.

There are several avenues for future work. First, there is the problem of finding weaker conditions under which we can decide whether $N \sqsubseteq_s M$. In addition, it would be interesting to find conditions under which $\tilde{\mathcal{L}}(M)$ can be constructed. There is also the problem of extending the results to situation in which we can make additional observations; for example, we might consider languages such as CSP in which we can also observe refusals. Finally, one of the motivations for this work was the problem of deciding whether there is a test case that is capable of distinguishing two states of an FSM and, despite this being undecidable, it would be interesting to develop heuristics for this problem.

References

[1] A. V. Aho, A. T. Dahbura, D. Lee, and M. U. Uyar. An optimization technique for protocol conformance test generation based on UIO sequences and rural chinese postman tours. *IEEE Transactions on Communications*, 39(11):1604–1615, 1991.

[2] R. Alur, C. Courcoubetis, and M. Yannakakis. Distinguishing tests for nondeterministic and probabilistic machines. In *27th ACM Symposium on Theory of Computing*, pages 363–372, 1995.

[3] Alberto Bertoni, Giancarlo Mauri, and Nicoletta Sabadini. Equivalence and membership problems for regular trace languages. In *9th Colloquium on Automata, Languages and Programming (ICALP)*, volume 140 of *Lecture Notes in Computer Science*, pages 61–71. Springer, 1982.

[4] L. Cacciari and O. Rafiq. Controllability and observability in distributed testing. *Information and Software Technology*, 41(11–12):767–780, 1999.

[5] Emanuela G. Cartaxo, Patrícia D. L. Machado, and Francisco G. Oliveira Neto. On the use of a similarity function for test case selection in the context of model-based testing. *Software Testing, Verification and Reliability*, 21(2):75–100, 2011.

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

[7] Eduardo Cunha de Almeida, João Eugenio Marynowski, Gerson Sunyé, Yves Le Traon, and Patrick Valduriez. Efficient distributed test architectures for large-scale systems. In *22nd IFIP WG 6.1 International Conference on Testing Software and Systems (ICTSS 2010)*, volume 6435 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 2010.

[8] Volker Diekert and Yves Mtivier. Partial commutation and traces. In *Handbook of formal languages, vol. 3: beyond words*, pages 457–533. 1997.

[9] R. Dssouli and G. von Bochmann. Error detection with multiple observers. In *Protocol Specification, Testing and Verification V*, pages 483–494. Elsevier Science (North Holland), 1985.

[10] R. Dssouli and G. von Bochmann. Conformance testing with multiple observers. In *Protocol Specification, Testing and Verification VI*, pages 217–229. Elsevier Science (North Holland), 1986.

[11] E. Farchi, A. Hartman, and S. Pinter. Using a model-based test generator to test for standard conformance. *IBM systems journal*, 41(1):89–110, 2002.

[12] Wolfgang Grieskamp. Multi-paradigmatic model-based testing. In *Formal Approaches to Software Testing and Runtime Verification (FATES/RV 2006)*, volume 4262 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2006.

[13] Wolfgang Grieskamp, Nicolas Kicillof, Keith Stobie, and Victor Braberman. Model-based quality assurance of protocol documentation: tools and methodology. *The Journal of Software Testing, Verification and Reliability*, 21(1):55–71, 2011.

[14] Stefan Haar, Claude Jard, and Guy-Vincent Jourdan. Testing input/output partial order automata. In *(TestCom/FATES 2007)*, volume 4581 of *Lecture Notes in Computer Science*, pages 171–185. Springer, 2007.

[15] F. C. Hennie. Fault-detecting experiments for sequential circuits. In *Proceedings of Fifth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 95–110, Princeton, New Jersey, November 1964.

[16] R. M. Hierons. Oracles for distributed testing. *IEEE Transactions on Software Engineering*, to appear.

[17] Robert M. Hierons. Canonical finite state machines for distributed systems. *Theoretical Computer Science*, 411(2):566–580, 2010.

[18] Robert M. Hierons. Reaching and distinguishing states of distributed systems. *SIAM Journal of Computing*, 39(8):3480–3500, 2010.

[19] Robert M. Hierons, Mercedes G. Merayo, and Manuel Núñez. Controllable test cases for the distributed test architecture. In *6th International Symposium on Automated Technology for Verification and Analysis (ATVA 2008)*, volume 5311 of *Lecture Notes in Computer Science*, pages 201–215. Springer, 2008.

[20] Robert M. Hierons, Mercedes G. Merayo, and Manuel Núñez. Implementation relations for the distributed test architecture. In *20th IFIP TC 6/WG 6.1 International Conference on the Testing of Software and Communicating Systems (TestCom/FATES 2008)*, volume 5047 of *Lecture Notes in Computer Science*, pages 200–215. Springer, 2008.

[21] Robert M. Hierons and Manuel Núñez. Testing probabilistic distributed systems. In *Joint 12th IFIP WG 6.1 International Conference, FMOODS 2010 and 30th IFIP WG 6.1 International Conference, FORTE 2010*, volume 6117 of *Lecture Notes in Computer Science*, pages 63–77. Springer, 2010.

[22] Robert M. Hierons and Hasan Ural. The effect of the distributed test architecture on the power of testing. *The Computer Journal*, 51(4):497–510, 2008.

[23] Joint Technical Committee ISO/IEC JTC 1. *International Standard ISO/IEC 9646-1. Information Technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts.* ISO/IEC, 1994.

[24] Ahmed Khoumsi. A temporal approach for testing distributed systems. *IEEE Transactions on Software Engineering*, 28(11):1085–1103, 2002.

[25] Eryk Kopczynski and Anthony Widjaja To. Parikh images of grammars: Complexity and applications. In *The 25th Annual IEEE Symposium on Logic in Computer Science (LICS 2010)*, pages 80–89. IEEE Computer Society, 2010.

[26] G. Luo, A. Petrenko, and G. v. Bochmann. Selecting test sequences for partially-specified nondeterministic finite state machines. In *The 7th IFIP Workshop on Protocol Test Systems*, pages 95–110, Tokyo, Japan, November 8–10 1994. Chapman and Hall.

[27] Rohit Parikh. On context-free languages. *Journal of the ACM*, 13(4):570–581, 1966.

[28] Alexandre Petrenko and Nina Yevtushenko. Testing from partial deterministic FSM specifications. *IEEE Transactions on Computers*, 54(9):1154–1165, 2005.

[29] E. L. Post. A variant of a recursively unsolvable problem. *Bulletin of the American Mathematical Society*, 52:264–268, 1946.

[30] M. O. Rabin and D. Scott. Finite automata and their decision problems. *IBM Journal of Research and Development*, 3(2):114–125, 1959.

[31] Omar Rafiq and Leo Cacciari. Coordination algorithm for distributed testing. *The Journal of Supercomputing*, 24(2):203–211, 2003.

[32] B. Sarikaya and G. v. Bochmann. Synchronization and specification issues in protocol testing. *IEEE Transactions on Communications*, 32:389–395, April 1984.

[33] Thomas J. Schaefer. The complexity of satisfiability problems. In *Tenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–226, 1978.

[34] Jan Tretmans. Model based testing with labelled transition systems. In *Formal Methods and Testing*, volume 4949 of *Lecture Notes in Computer Science*, pages 1–38. Springer, 2008.

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

[36] Hasan Ural and Craig Williams. Constructing checking sequences for distributed testing. *Formal Aspects of Computing*, 18(1):84–101, 2006.

[37] Gregor von Bochmann, Stefan Haar, Claude Jard, and Guy-Vincent Jourdan. Testing systems specified as partial order input/output automata. In *20th IFIP TC 6/WG 6.1 International Conference on Testing of Software and Communicating Systems (TestCom/FATES)*, volume 5047 of *Lecture Notes in Computer Science*, pages 169–183. Springer, 2008.