**Recent Advances in the Theory of Checking Experiments with State-oriented Models**

Alexandre PETRENKO  
CRIM, Centre de recherche informatique de Montréal, 405 Ogilvy Avenue, Suite 101, Montréal (Québec) H3N 1M3, Canada

**Keywords:** Formal methods, Finite state machines, Input output transition systems, Checking experiments, Model based testing, Conformance testing, Software testing.

**Abstract.** We consider the problem of model-based testing of reactive systems with state-oriented models, such as finite state machines and transition systems, both distinguishing inputs and outputs. Inputs to an Implementation under Test (IUT) are generated by a tester, while outputs are analyzed by the latter to conclude whether the IUT is a conforming implementation of a given specification. This problem has its roots in the theory of checking experiments for finite state machines. It continues to attract a lot of attention of research community. In this paper, we provide a short overview of directions in which this theory has evolved. We report on our recent contributions and point to several unsolved tasks.

**Introduction**

Formal methods are indispensable in high assurance systems engineering, as they provide repeatable results in verification and testing: one can always use models to try to reproduce a detected flaw to make sure it is not a false alarm or that it is fixed. Model-based testing is a very active research area, where several academic and commercial tools are already being applied for safely-critical systems.

The problem of model-based testing of reactive systems has traditionally been addressed with state-oriented models, such as finite state machines and transition systems, both distinguishing inputs and outputs. Inputs to an Implementation under Test (IUT) are generated by a tester, while outputs are analyzed by the latter to conclude whether the IUT is a conforming implementation of a given specification.

Various conformance relations can be used to generate test cases; such relations are usually model-specific. Nevertheless, given a particular relation, there seems to be a universally accepted notion of “dream-test suite”, complete test suite which is sound and exhaustive. To be sound, it must not fail on any conforming implementation, and to be exhaustive it must fail on each nonconforming implementation. Constructing such a test suite is a challenging problem which has been first addressed in so called “gedanken” checking experiments with Finite State Machines ( FSM) about sixty years ago, see [1, 2].

The theory of checking experiments for FSMs still continues to attract a lot of attention of research community. In this paper, we aim at providing a short overview of directions in which this theory has evolved. We report on our recent contributions and formulate several unsolved tasks.

**State-oriented Models**

We consider two main types of state-oriented models, finite state machines and input output transition systems. The main difference between the two models lies in state transition labelling. In FSMs, each transition is labelled by pair of input and output, their execution is an atomic action. In input output transition systems, each transition is labelled by input, output or internal action.
FSM and Their Variations

A **Finite State Machine** (FSM or simple machine) $M$ is a 5-tuple $(S, s_0, I, O, T)$, where $S$ is a finite set of states with initial state $s_0$; $I$ and $O$ are finite non-empty disjoint sets of inputs and outputs, respectively; $T$ is a transition relation $T \subseteq S \times I \times O \times S$. An FSM is often called a Mealy machine.

$M$ is **completely specified** (complete FSM) if for each tuple $(s, x) \in S \times I$ there exists transition $(s, x, o, s') \in T$, otherwise it is partially defined (partial FSM). It is **deterministic** (DFSM) if for each $(s, x) \in S \times I$ there exists at most one transition $(s, x, o, s') \in T$, otherwise it is **nondeterministic** (NFSM); $M$ is **observable** if for each tuple $(s, x, o) \in S \times I \times O$ there exists at most one transition; otherwise, it is **non-observable**.

By combinations, we have several classes of FSMs:
- Complete deterministic FSM
- Partial deterministic FSM
- Complete observable nondeterministic FSM
- Complete non-observable nondeterministic FSM
- Partial observable nondeterministic FSM
- Partial non-observable nondeterministic FSM

Each of them has specific features which require special handling in considering checking experiments, as we discuss later in the paper.

A more complex types of FSMs are commonly known as Extended FSMs, see, e.g., [3, 4]. In these models sets of input and output as well as sets of states are not restricted to be finite sets. This feature creates additional research challenges.

Several relations have been defined for states and FSMs.

Equivalence relation which requires that the sets of traces coincide is applicable to complete deterministic and nondeterministic machines. A machine is reduced if it has no equivalent states, otherwise it is unreduced.

Quasi-equivalence of a complete machine to a partial FSM requires trace equivalence only for input sequences defined in the partial machine. Reduction relation defined for complete nondeterministic machines requires that one machine is trace included into another FSM. Quasi-reduction relation is a special kind of trace inclusion, when a partial FSM has so-called harmonized traces.

**Input Output Transition Systems and Their Variations**

We use **input/output transition systems** (IOTS, a.k.a. input/output automata) for modelling systems. Formally, an IOTS $S$ is a quintuple $(S, s_0, I, O, T)$, where $S$ is a finite set of states, $s_0 \in S$ is initial state, $I$ and $O$ are disjoint sets of input and output actions, respectively, and $T \subseteq S \times (I \cup O \cup \tau) \times S$ is the transition relation, where $\tau$ is internal action.

$S$ is **deterministic** if it has no internal actions and $h_S$ is a function on a subset of $S \times (I \cup O)$, otherwise it is **nondeterministic**. IOTS $S$ is **input-enabled** or **completely specified** if any state with an input transition has transitions for all inputs, otherwise it is **partially specified**. $S$ is **output-deterministic** if each state has at most one output transition, otherwise, it is **output-nondeterministic**. States with both input and output transitions represents input/output **conflicts**; IOTS without conflicts is often called Mealy IOTS. States that have only input transitions are called stable or **quiescent**. By combinations, we have several classes of IOTSs, mostly used IOTSs are input-enabled and output-nondeterministic with or without input/output conflicts. It is usually assumed that they are **progressive**, this means that IOTS has no sink state and each cycle contains a transition labeled with input, i.e., there is no output divergence.

One of the mostly used relations between two IOTSs is a so-called ioco relation, IOTS $T$ is ioco to IOTS $S$ if an IOTS $T$ after any trace defined in $S$ has outputs that are defined in it. Quiescence is considered as an output in this relation.
Checking Experiments for FSM and IOTS

We provide a short overview of the main results obtained for the two state-oriented models.

Checking Experiments for FSM and Their Extensions

The main idea of constructing a checking experiment is to detect faults of predefined types in any IUT, so it is fault model based testing. In [5], we proposed to define a fault model as a tuple of a specification, fault domain, and conformance relation. In the context of testing from finite state machines, the specification is a certain type of an FSM. Conformance relation is specific to the type of an FSM and for complete deterministic machines it is equivalence relation. Fault domain is a set of implementation machines, aka mutants, each of which models some faults, such as output, transfer or transition faults. In the traditional checking experiment theory the specification is a complete deterministic FSM and the fault domain is a universe of all such machines with a given number of states and input and output sets of the specification, see, e.g., [5-11]. It is possible to consider a smaller fault domain, as suggested in [12, 13]; this is achieved by defining a fault domain as a set of all possible submachines of a given nondeterministic FSM, called a mutation machine. The mutation machine contains a specification machine and extends it with a number of mutated transitions, modelling potential faults.

Checking experiments are also defined for deterministic FSMs, which are not required to be reduced and completely specified [14]. Next step in the development of the checking experiment theory was to allow machines to be nondeterministic, see, e.g, [15-20].

In spite of a number of obtained results, the problem of checking experiment generation for an arbitrary fault model and any type of specification FSMs remains open.

Towards Checking Experiments for IOTS

The theory of checking experiments with FSM is fully applicable to Mealy IOTSs. However, there is not much work done to extend this theory to more general types of IOTSs.

Traditional approaches to testing systems modeled by IOTS are based on the so-called ioco testing framework [21]. In this framework, a tester interacting with an implementation under test (IUT) either applies input to the IUT or accept any output from the IUT, but cannot do both. The composition used to formalize this interaction is synchronous and does not distinguish between inputs and outputs. By the semantics of the composition, the tester has to preempt outputs of the IUT any time it decides to send inputs to the IUT, which overrides the principle that “output actions can never be blocked by the environment” [21]. Considering systems that generate output actions at their own will, the assumption implied by the synchronous composition that testers can block outputs from IUT is unrealistic.

The input/output conflict problem creates a serious challenge for attempts to generalize checking experiment to IOTS. Previous work considered transition coverage for asynchronous testing via queues, see [22] and test generation driven by test purposes [23]. In a recent work [24], it was demonstrated that checking experiments (complete test suite) that tolerate input/output conflicts can be constructed for a certain class of IOTSs. The key assumption about the implementation IOTSs in the fault domain is that each implementation in a state with input/output conflict does not produce any output if its input queue contains an input. Such implementations are called input-eager.

More work is needed to remove a number of constraints imposed by this approach on a specification IOTS before checking experiments can be generated for any type of IOTS. Also it is interesting to see whether alternative assumptions about the implementations can also be made.

Summary

We considered the problem of model-based testing of reactive systems with state-oriented models, namely, finite state machines and transition systems, both distinguishing inputs and outputs. Inputs to an Implementation under Test (IUT) are generated by a tester, while outputs are analyzed by the latter to conclude whether the IUT is a conforming implementation of a given specification. This problem
has its roots in the theory of checking experiments for finite state machines. It continues to attract a lot of attention of research community. In this paper, we aimed at providing a short overview of directions in which this theory has evolved. The reader is referred to a recent survey [25] to have a broader view on software testing using models. We reported on our recent contributions to the area and formulated several unsolved tasks for two models.

Acknowledgement

This research is supported in part by the NSERC of Canada and by the MEIE of Gouvernement du Québec.

References


