Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 52 (2008) 432–460
Computer Networks
www.elsevier.com/locate/comnet

Formal testing from timed finite state machines ☆

Mercedes G. Merayo, Manuel Núñez *, Ismael Rodríguez

Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid, E-28040 Madrid, Spain

Received 1 September 2006; received in revised form 6 September 2007; accepted 4 October 2007
Available online 13 October 2007

Responsible Editor: Dr. R. Gotzhein

Abstract

In this paper we present a formal methodology to test both the functional and temporal behaviors in systems where temporal aspects are critical. We extend the classical finite state machines model with features to represent timed systems. Our formalism allows three different ways to express the timing requirements of systems. Specifically, we consider that time requirements can be expressed either by means of fix time values, by using random variables, or by considering time intervals. Different implementation relations, depending on both the interpretation of time and on the non-determinism appearing in systems, are presented and related. We also study how test cases are defined and applied to implementations. Test derivation algorithms, producing sound and complete test suites, are also presented. That is, by deriving these test suites we relate the different notions of passing tests and the different implementation relations. In other words, for a given correctness criterion, a system represents an appropriate implementation of a given model if and only if the system successfully passes all the test belonging to the derived test suite.

Keywords: Conformance testing; Formal methods; Timed systems

1. Introduction

The scale and heterogeneity of current systems make impossible for developers to have an overall view of them. Thus, it is difficult to foresee those errors that are either critical or more probable. In this context, *formal testing techniques* provide systematic procedures to check implementations in such a way that the coverage of critical parts/aspects of the system under test depends less on the intuition of the tester. In this line, they allow to test the correctness of a system with respect to a specification. Formal testing originally targeted the functional behavior of systems, such as determining whether the tested system can, on the one hand, perform certain actions and, on the other hand, does not perform some unexpected ones. While the relevant aspects of some systems only concern *what* they do, in some other systems it is equally relevant *how* they do what they do. Thus, after the initial consolidation stage, formal testing techniques

☆ Research partially supported by the Spanish MEC project WEST/FAST (TIN2006-15578-C02-01) and the Marie Curie project TAROT (MRTN-CT-2003-505121).
* Corresponding author.
E-mail addresses: mgmerayo@fdi.ucm.es (M.G. Merayo), mn@sip.ucm.es (M. Núñez), isrodrig@sip.ucm.es (I. Rodríguez).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2007.10.002

started also to deal with *non-functional* properties such as the probability of an event to happen, the time that it takes to perform a certain action, or the time when a certain action happens. The work on formal testing applied to timed systems has attracted a lot of attention during the last years. In fact, there are already several proposals for timed testing (e.g. [22,7,17,31,28,10,24,12,9,20,19,3]). In these papers, time is considered to be *deterministic*, that is, time requirements follow the form "after/before t time units..." In fact, in most of the cases time is introduced by means of clocks following [1]. Even though the inclusion of time allows to give a more precise description of the system to be implemented, there are frequent situations that cannot be accurately described by using this notion of deterministic time. For example, in order to express that a message will arrive at any point of time belonging to the interval [0,1] we will need, in general, infinite transitions, one for each possible value belonging to the interval. In this case, it would be more appropriate to use time intervals to describe the system. Let us consider now that we have to simulate the performance of a petrol station. Since cars arrive in such stations by following a Poisson distribution, we would need again to use an infinite number of transitions. Moreover, if we have to use a time interval we would be very imprecise since all that we could say is that the next car will arrive in the interval $[0,\infty)$. Thus, it would be very useful to have a mechanism allowing to express that a time constraint is given by using a random variable that follows a precise probability distribution function.

In this paper we study formal testing methodologies where the temporal behavior of systems is taken into account. In order to present our contribution, we will use a simple extension of the classical concept of *Finite State Machine*. Intuitively, transitions in finite state machines indicate that if the machine is in a state s and receives an input i then it will produce an output o and it will change its state to s'. An appropriate notation for such a transition could be $s \xrightarrow{i/o} s'$. If we consider a timed extension of finite state machines, transitions as $s\xrightarrow{i/o}_d s'$ indicate that the time between receiving the input i and returning the output o is given by d, where d belongs to a certain time domain. Even though we have chosen finite state machines, because they are widely used in the formal testing community, our results can be straightforwardly adapted to deal with (input-output) labelled transition systems; the extension of our results to deal with (timed) automata is more cumbersome, but not difficult taking as basis [28].

We consider three different domains to express temporal requirements: Time given by *fix values*, by *random variables*, and by *time intervals*. A transition such as $s\xrightarrow{i/o}_t s'$ indicates that if the machine is in state s and receives the input i, it will perform the output o and reach the state s' after t time units. A transition as $s\xrightarrow{i/o}_\xi s'$ indicates that if the machine is in state s and receives the input i, it will perform the output o and reach the state s' after a certain time t with probability $F_\xi(t)$, where F_ξ is the probability distribution function associated with ξ. Finally, $s\xrightarrow{i/o}_{[t_1,t_2]} s'$ means that if the machine is in state s and receives the input i, it will perform the output o and reach the state s', and it will take a time greater than or equal to t_1 but smaller than or equal to t_2. Even though our methodology allows three very different ways for representing time requirements, random variables and time intervals present a more complex situation than fix time values. Thus, we need to treat them separately, although following a common line. Specifically, due to the fact that we work under the assumption of a black-box testing framework, testers cannot compare in a direct way timed requirements of the *real* implementation with those established in the model (either random variables or time intervals). The idea is that we can *see* the random variable (or the time interval) defining a given transition in the model, but we cannot do the same with the corresponding transition of the implementation, since we do not have access to it. Thus, in contrast with approaches considering fix time values, to perform a transition of the implementation once does not allow us to obtain all the information about its temporal behavior. In order to overcome this problem, we have to perform the same transition to collect different time values. So, we consider a set of observations collected by means of the interaction with the implementation and establish different levels of temporal agreement with respect to the (accessible) values appearing in the formal model. We think that this additional complication is the main reason why there is almost no work on testing timed systems where time is not given by means of fix time values. In fact, as far as we know [25] represents the only proposal presenting a formal testing methodology to test stochastic time systems that can be described by means of finite state machines. Also, [2,21] present testing frameworks for stochastic systems but their approaches are not