Available online at www.sciencedirect.com
SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science

ELSEVIER Electronic Notes in Theoretical Computer Science 113 (2005) 145–162
www.elsevier.com/locate/entcs

Monitoring Algorithms for Metric Temporal Logic Specifications

Prasanna Thati and Grigore Roşu

Department of Computer Science
University of Illinois at Urbana Champaign, USA
{thati,grosu}@cs.uiuc.edu

September 11, 2004

Abstract

Program execution traces can be so large in practical testing and monitoring applications that it would be very expensive, if not impossible, to store them for detailed analysis. Monitoring execution traces *without storing* them, can be a nontrivial matter for many specification formalisms, because complex formulae may require a considerable amount of information about the past. *Metric temporal logic* (MTL) is an extension of propositional linear temporal logic with discrete-time-bounded temporal operators. In MTL, one can specify time limits within which certain temporal properties must hold, thus making it very suitable to express real-time monitoring requirements. In this paper, we present monitoring algorithms for checking timestamped execution traces against formulae in MTL or certain important sublogics of it. We also present lower bounds for the monitoring problem, showing that the presented algorithms are asymptotically optimal.

Keywords: Runtime verification, execution trace, specification, metric temporal logic.

1 Introduction

Runtime verification and *monitoring* have been proposed as lightweight formal verification methods [13] with the explicit goal of checking systems against their formal requirements while they execute. In most monitoring applications, execution traces are available only *incrementally* and they are *much larger* than the formulae against which they are checked. Storing an entire execution trace and then performing the formal analysis by having random access to the trace is very expensive and sometimes even impossible. For example, the monitor may lack resources, e.g., if it runs within an embedded system, or the monitor

1571-0661/$ – see front matter
doi:10.1016/j.entcs.2004.01.029

may be expected to react promptly when its requirements are violated, in order for the system to safely take a recovery or a shutdown action.

In this paper, we adopt the position that a *monitoring algorithm* does not store execution traces, but rather *consumes* the events as they are received from the monitored program. The problem of checking execution traces against temporal specifications is known to have very simple and efficient algorithms for several temporal logics, as shown for example in [19], but most of these algorithms assume that the entire execution trace is available beforehand, so they violate the assumptions for a monitoring algorithm.

In this paper, we investigate monitoring algorithms for the *metric temporal logic (MTL)* [1,15] and its sublogics. MTL is an extension of propositional linear temporal logic (LTL) that can refer to discrete-timed properties, and its models are timestamped state-sequences, thus making it an appealing formalism for expressing monitoring requirements in real-time systems. Besides the propositional operators, MTL allows future and past time linear temporal operators which are bounded by *discrete-time intervals*. For example, $\phi \mathcal{U}_{[3,7]} \psi$ states that ψ should hold between 3 and 7 time units from now, and until then ϕ should hold. One or both of the ends of an interval can be 0 or ∞. LTL can be seen as a special case of MTL where every interval is $[0, \infty)$. As introduced in [1], MTL also provides *congruences* that allow one to state that a formula should hold periodically with respect to an absolute time. We call these *absolute congruences* and support them in our MTL specifications as well, but in addition we introduce a novel variant that we call *relative congruence*. Relative congruences allow one to refer to moments that occur periodically starting with the *current* time.

We first present a general MTL monitoring algorithm based on the idea of transforming the MTL formula as each time-stamped observation (or event, for short) is received from the monitored program. The underlying principle of the algorithm is "resolve the past and derive the future". By "resolving the past" we mean that the MTL formula is transformed into an equivalent formula with the property that it has no past time operator rooted subformulae which are not guarded by other temporal operators. By "deriving the future" we mean that the MTL formula is transformed into a new MTL formula with the property that the current formula holds before processing the newly received event if and only if the derived formula holds after processing the event. We show that this MTL monitoring algorithm runs in space $O(m2^m)$ and takes time $O(m^3 2^{3m})$ for processing each event, where m equals $|\underline{\phi}|$ plus the sum of all the numeric constants occurring in ϕ, and $\underline{\phi}$ is ϕ with all the timing subscripts droped. The reader may note that although exponential, these bounds are *independent* of the size of execution trace which is