Model Checking Timed Automata with Priorities Using DBM Subtraction

Alexandre David[1], John Håkansson[2], Kim G. Larsen[1], and Paul Pettersson[2]

[1] Department of Computer Science, Aalborg University, Denmark
{adavid, kgl}@cs.aau.dk
[2] Department of Information Technology, Uppsala University, Sweden
{johnh, paupet}@it.uu.se

Abstract. In this paper we describe an extension of timed automata with priorities, and efficient algorithms to compute subtraction on DBMs (difference bounded matrices), needed in symbolic model-checking of timed automata with priorities. The subtraction is one of the few operations on DBMs that result in a non-convex set needing *sets* of DBMs for representation. Our subtraction algorithms are efficient in the sense that the number of generated DBMs is significantly reduced compared to a naive algorithm. The overhead in time is compensated by the gain from reducing the number of resulting DBMs since this number affects the performance of symbolic model-checking. The uses of the DBM subtraction operation extend beyond timed automata with priorities. It is also useful for allowing guards on transitions with urgent actions, deadlock checking, and timed games.

1 Introduction

Since the introduction of timed automata [2] in 1990, the theory has proven its capability of specifying and analysing timed systems in many case studies, e.g., [4,23]. To support such studies, tools as Kronos [7], Uppaal [18], and RED [24] have been developed to offer means for modelling, simulation, model-checking, and also testing, of real-time systems specified as timed automata.

In the implementation of real-time systems, the concept of *priorities* is often used as a way to structure and control the usage of shared resources. Priorities are often associated with processes (or tasks) to control their usage of shared resources such as CPU or shared memory areas. As a consequence, programming languages such as Ada [3,12], and scheduling policies used in real-time operating system, such as *rate-monotonic scheduling* [9], are often based on a notion of priorities on tasks. In lower levels, closer to the hardware, priorities are often associated with interrupts to hardware devices and access to e.g., shared communication buses.

Priorities have been studied in process algebras, e.g., [11,8], and can be modelled using timed automata [12,14]. However, it can be cumbersome and error-prone to do so. Consider the simple example shown in Figure 1 and assume that the location l can be reached with any time assignment satisfying the constraint

E. Asarin and P. Bouyer (Eds.): FORMATS 2006, LNCS 4202, pp. 128–142, 2006.

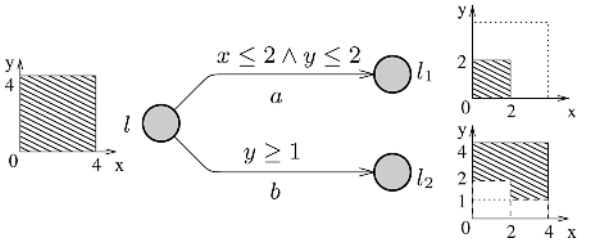

Fig. 1. A timed automaton with priorities on actions

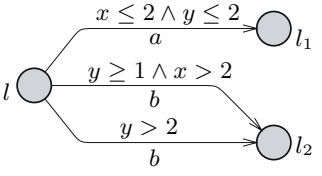

Fig. 2. Encoding of the priorities in Fig. 1

$x \leq 4 \wedge y \leq 4$. Further assume that the edge labelled with a has priority over the edge labelled b. We see that l_1 can be reached with any time assignment satisfying the constraint $x \leq 2 \wedge y \leq 2$. The location l_2 is reachable under the constraint $(y \geq 1 \wedge x \leq 4 \wedge y \leq 4) \wedge \neg(x \leq 2 \wedge y \leq 2)$, which is a non-convex set of clock valuations and thus not representable as a conjunction of simple constraints. This fact will make (symbolic) state-space exploration potentially costly, since the set of clock valuations reachable in one step over a low priority transition, such as transitions derived from the b-edge, generally will have to be represented by *a set* of convex constraint systems. In Figure 2 we show a timed automaton in which the priorities of the automaton in Figure 1 have been encoded. Note that the b-edge has been split to two edges to encode the disjunctive constraints on the clock valuations reaching l_2.

Model-checking tools for timed automata typically uses DBMs (difference bound matrices) [13,22] to represent convex constraints on clock variables. However, as illustrated above, analysis of timed automata with priorities will require the model-checking engine to efficiently handle disjunctive constraints. As a second contribution of this paper, we present a variety of techniques for performing *subtractions* on DBMs. That is, how to compute $D - D'$ defined as $D \wedge \neg D'$, for two DBMs D and D'. Guided by the goal to minimise the set of DBMs resulting from subtraction, and to keep them disjoint, we give a heuristic algorithm with good performance. To back up this statement, we present experimental evidence from applying a version of the UPPAAL tool extended with priorities, on a set of examples. We note that DBM subtraction is already needed for backward model-checking of full TCTL or scheduler synthesis [23], controller synthesis [10], and to support urgent guards.

The rest of this paper is organised as follow: Timed automata with priorities are described in Section 3, and the required DBM subtraction operation in