UPPAAL-Tiga: Time for Playing Games!
(Tool Paper)

Gerd Behrmann[1], Agnès Cougnard[1], Alexandre David[1], Emmanuel Fleury[2], Kim G. Larsen[1], and Didier Lime[3]

[1] CISS, Aalborg University, Aalborg, Denmark
{behrmann,adavid,kgl}@cs.aau.dk
[2] LaBRI, Bordeaux-1 University, CNRS (UMR 5800), Talence, France
fleury@labri.fr
[3] IRCCyN, École Centrale de Nantes, CNRS (UMR 6597), Nantes, France
Didier.Lime@irccyn.ec-nantes.fr

Abstract. In 2005 we proposed the first efficient on-the-fly algorithm for solving games based on timed game automata with respect to reachability and safety properties. The first prototype presented at that time has now matured to a fully integrated tool with dramatic improvements both in terms of performance and the availability of the extended input language of UPPAAL-4.0. The new tool can output strategies or let the user play against them both from the command line and from the graphical simulator that was completely re-designed.

1 Introduction

For more than a decade timed, priced and hybrid games have been proposed and studied by various researchers [AMPS98, DAHM01, MPS95, BCFL04]. Though several decidability results and algorithms have been presented, so far only prototype tools have been developed [AT02]. UPPAAL-TIGA[1] is the first efficient tool supporting the analysis of timed games allowing synthesis of controllers for control problems modelled as timed game automata and with safety or liveness control objectives.

2 What Can Be Done with Uppaal-Tiga?

Control Problems. The modeling formalism of UPPAAL-TIGA consists of a network of timed game automata [MPS95] (NTGA). A timed game automaton is a timed automaton [AD94] in which the set of actions is partitioned into *controllable* actions and *uncontrollable* actions. The former are actions that can be triggered by the controller, the latter by the environment/opponent. The opponent has

---

[1] http://www.cs.aau.dk/~adavid/tiga/

priority over the controller. Given a NTGA, we are mainly interested in two types of control objectives:

The *reachability* objective: Is it possible to find a strategy for the triggering of controllable actions guaranteeing that a given set of (goal) states of the system is reached regardless of what and when uncontrollable actions are taken?

The *safety* objective: Is it possible to find a strategy for the triggering of controllable actions guaranteeing that a given set of (bad) states of the system are never reached regardless of what and when uncontrollable actions are taken?

Formally, control objectives are formulated as "`control: P`", where P is TCTL formula specifying either a safety property (`A[]` ϕ or `A[`ϕ_1 `W` ϕ_2`]`) or a liveness property (`A<>` ϕ or `A[`ϕ_1 `U` ϕ_2`]`). Given a control objective "`control: P`" the search engine of Uppaal-Tiga will provide a strategy (if any such exists) under which the behaviour will satisfy P. Here a (winning) strategy is simply a function describing for each state of the system what the controller should do either in terms of "performing a particular controllable action" or to "delay".
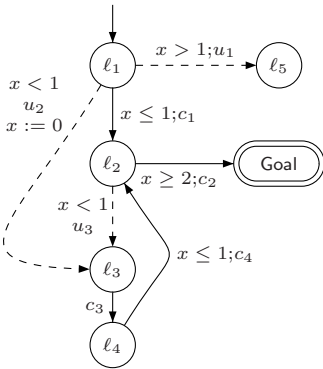

Fig. 1. Timed game automaton example

Example. To illustrate the use of Uppaal-Tiga, consider the example in Fig. 1, consisting of a timed game automaton A with one clock x. It has two types of edges: controllable (c_i) and uncontrollable (u_i). A defines the rules of the game. In our current example (Fig. 1), we define the simple reachability objective "`control: A<> A.Goal`". Uppaal-Tiga searches for a winning strategy on this game using the algorithm proposed in [CDF+05]. Each winning condition, entered in Uppaal-Tiga as a regular query, is marked as "*satisfied*" if there exists a winning strategy and "*not satisfied*" if none exists. Hence, in our example, the query "`control: A<> A.Goal`" will be marked "*satisfied*". Moreover, the tool can output the corresponding strategy from the command line or let the user play against it in the

simulator as shown in Fig. 2. For controllable games (here), Uppaal-Tiga plays the controller and the user is the opponent. The first transition is for the controller (Uppaal-Tiga), the second for the opponent (user), and the third is for the opponent but it will be countered by the controller (Uppaal-Tiga takes the transition to ℓ_2), hence it is greyed – the opponent cannot take this action.

Applications. Uppaal-Tiga has recently been used for an industrial case study with the company Skov A/S specializing in climate control systems used for modern pig and poultry stables [DJLR07]. The synthesis capability of the tool has been combined with Simulink and Real-Time Workshop to provide a complete tool chain for synthesis, simulation, and automatic generation of production code.

Uppaal-Tiga has also been recently applied to check for (bi-)simulation between timed automata and timed game automata [CDL07]. Given two timed
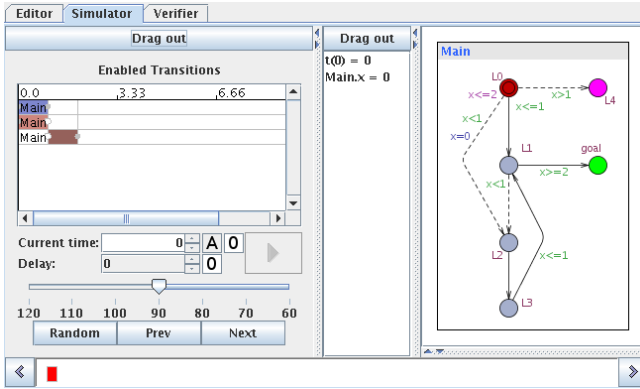
Fig. 2. View of the simulator when playing a strategy

automata, the tool can check if one (bi-)simulates the other and similarly for timed game automata with applications for controller synthesis with partial observability. This technique has been applied to the compositional verification of the ZeroConf protocol.

Our tool is being used in the AMAES project[2], a French national project on Advanced Methods for Autonomous Embedded Systems. UPPAAL-TIGA has been successfully applied for controlling the autonomous robot Dala [LC04] in charge of taking pictures and transmitting them back to Earth during limited transmission windows. It is desirable in such control problems to optimize the moves to save power.

3 What Is New?

Input Language. The new generation of UPPAAL-TIGA inherits the enriched input language of UPPAAL-4.0, with the only exception of priorities. The user has now access to C-like syntax to declare functions, custom types, *etc.* In addition, users can now define more complex winning conditions by means of logical formulas in the indicated subset of TCTL.

Strategies. UPPAAL-TIGA is able to generate a strategy for the controller, which ultimately corresponds to a control program, or a "counter-strategy" for the opponent as a proof that the controller cannot win. These strategies can now be output as decision graphs (hybrid BDD/CDD graphs). The discrete part of the states is represented as a BDD and the symbolic part as a CDD [LPWY99]. Finally, UPPAAL-TIGA allows the user to play against the strategy in both the GUI[3] and CLI.

Performances. UPPAAL-TIGA is faster by several orders of magnitude on large examples (more than 1000 times faster) and consumes much less memory (100

---

[2] http://www-verimag.imag.fr/∼krichen/AMAES/
[3] This is a major new feature for the next major release.

times less) than the first prototype presented at CONCUR'05. This comes from the full integration of the algorithm in the UPPAAL-4.0 [BDH$^+$06][4] framework. This brings to UPPAAL-TIGA some of the reliability and performances achieved by years of UPPAAL developments. Also improvements have been made on the DBM library, in particular on subtractions, partitions, federations, specific operations for timed games (*e.g.* the computation of controllable predecessors *w.r.t.* delay), and the essential operation of merging several difference bound matrices (DBMs) into one.

4   Conclusion

UPPAAL-TIGA is a new (and the only efficient) tool for controller synthesis for control problems modeled as timed game automata. Its performance and usability has greatly been improved by its integration with UPPAAL-4.0. In a future version methods for preventing synthesis of strategies generating zeno behaviour will be provided. Also the computationally much more complex problem of synthesis under partial observability is under investigation.

References

[AD94]   Alur, R., Dill, D.: A Theory of Timed Automata. Theoretical Computer Science 126(2), 183–235 (1994)

[AMPS98]   Asarin, E., Maler, O., Pnueli, A., Sifakis, J.: Controller Synthesis for Timed Automata. In: Proc. IFAC Symp. on System Structure & Control, pp. 469–474. Elsevier, Amsterdam (1998)

[AT02]   Altisen, K., Tripakis, S.: Tools for controller synthesis of timed systems. In: RT-TOOLS'02 (2002)

[BCFL04]   Bouyer, P., Cassez, F., Fleury, E., Larsen, K.G.: Optimal Strategies in Priced Timed Game Automata. In: Lodaya, K., Mahajan, M. (eds.) FSTTCS 2004. LNCS, vol. 3328, pp. 148–160. Springer, Heidelberg (2004)

[BDH$^+$06] Behrmann, G., David, A., Håkansson, J., Hendriks, M., Larsen, K.G., Pettersson, P., Yi, W.: UPPAAL 4.0. In: Proc. of 3rd Int. Conf. on the Quantitative Evaluation of Systems, pp. 125–126. IEEE Computer Society Press, Los Alamitos (2006)

[CDF$^+$05] Cassez, F., David, A., Fleury, E., Larsen, K.G., Lime, D.: Efficient On-the-fly Algorithms for the Analysis of Timed Games. In: Abadi, M., de Alfaro, L. (eds.) CONCUR 2005. LNCS, vol. 3653, pp. 66–80. Springer, Heidelberg (2005)

[CDL07]   Chatain, T., David, A., Larsen, K.: Playing Games with Games (Unpublished)

[DAHM01]   De Alfaro, L., Henzinger, T.A., Majumdar, R.: Symbolic Algorithms for Infinite-State Games. In: Larsen, K.G., Nielsen, M. (eds.) CONCUR 2001. LNCS, vol. 2154, pp. 536–550. Springer, Heidelberg (2001)

[DJLR07]   David, A., Jessen, J.J., Larsen, K.G., Rasmussen, J.I.: Guided Controller Synthesis for Climate Controller Using UPPAAL-TIGA. Unpublished

---

[4] http://www.uppaal.com

[LC04]   Lemai-Chenevier, S.: IxTeT-eXeC: planification, réparation de plan et contrôle d'exécution avec gestion du temps et des ressources. PhD thesis, Institut National Polytechnique de Toulouse (2004)

[LPWY99] Larsen, K.G., Pearson, J., Weise, C., Yi, W.: Clock Difference Diagrams. Nordic Journal of Computing 6(3), 271–298 (1999)

[MPS95]   Maler, O., Pnueli, A., Sifakis, J.: On the Synthesis of Discrete Controllers for Timed Systems. In: Mayr, E.W., Puech, C. (eds.) STACS 95. LNCS, vol. 900, pp. 229–242. Springer, Heidelberg (1995)