UPPAAL STRATEGO*

Alexandre David, Peter Gjøl Jensen, Kim Guldstrand Larsen, Marius Mikučionis, and Jakob Haahr Taankvist

Department of Computer Science, Aalborg University, Selma Lagerlöfs Vej 300, 9220 Aalborg Øst, Denmark

Abstract. UPPAAL STRATEGO is a novel tool which facilitates generation, optimization, comparison as well as consequence and performance exploration of strategies for stochastic priced timed games in a user-friendly manner. The tool allows for efficient and flexible "strategy-space" exploration before adaptation in a final implementation by maintaining strategies as first class objects in the model-checking query language. The paper describes the strategies and their properties, construction and transformation algorithms and a typical tool usage scenario.

1 Introduction

Model checking may be used to verify that a proposed controller prevents an environment from causing dangerous situations while, at the same time, operating in a desirable manner. This approach has been successfully pursued in the setting of systems modeled as finite-state automata, timed automata, and probabilistic automata of various types with nuSMV [7], FDR [11], UPPAAL [3] and PRISM [13] as prime examples of model checking tools supporting the above mentioned formalisms. Most recently the simulation-based method of *statistical* model checking has been introduced in UPPAAL SMC [4], allowing for highly scaleable analysis of *fully* stochastic Sriced Timed Automata with respect to a wide range of performance properties. For instance, expected waiting-time and cost, and time-bounded and cost reachability probabilities, may be estimated (and tested) with an arbitrary precision and high degree of confidence. Combined with the symbolic model checking of UPPAAL this enables an adequate analysis of mixed critical systems, where certain (safety) properties must hold with absolute certainty, whereas for other quantitative (performance) properties a reasonably good estimation may suffice, see e.g. [10].

Rather than verifying a *proposed* controller, synthesis – when possible – allows an algorithmic construction of a controller which is guaranteed to ensure that the resulting systems will satisfy the desired correctness properties. The extension of controller synthesis to timed and hybrid games started in the 90s with the seminal work of Pnueli et al. [1,14] on controller synthesis for timed games where

* The research has received funding from the European FET projects SENSATION and CASSTING, the ARTEMIS project MBAT as well as the Sino-Danish Basic Research Center IDEA4CPS.

C. Baier and C. Tinelli (Eds.): TACAS 2015, LNCS 9035, pp. 206–211, 2015.
DOI: 10.1007/978-3-662-46681-0_16

the synthesis problem was proven decidable by a symbolic dynamic programming technique. In UPPAAL TIGA [2,5] an efficient on-the-fly algorithm for synthesis of reachability and safety objectives for timed games has been implemented, with a number of successful industrial applications having been made including zone-based climate control for pig-stables [12] and controllers for hydraulic pumps with 60% improvement in energy-consumption compared with industrial practice at the time [6,15].

However, once a strategy has been synthesized for a given objective no further analysis has been supported so far. In particular it has not been possible to make a deeper examination of a synthesized strategy in terms of other additional properties that may or may not hold under the strategy. Neither has it been possible to optimize a synthesized non-deterministic safety strategy with respect to desired performance measures. Both of these issues have been addressed by the authors in recent work [8,9], and in this paper we present the tool UPPAAL STRATEGO which combines these techniques to generate, optimize, compare and explore consequences and performance of strategies synthesized for stochastic priced timed games in a user-friendly manner. In particular, the tool allows for efficient and flexible "strategy-space" exploration before adaptation in a final implementation.

UPPAAL STRATEGO[1] integrates UPPAAL and the two branches UPPAAL SMC [4] (statistical model checking), UPPAAL TIGA [2] (synthesis for timed games) and the method proposed in [9] (synthesis of near optimal schedulers) into one tool suite. UPPAAL STRATEGO comes with an extended query language where strategies are first class objects that may be constructed, compared, optimized and used when performing (statistical) model checking of a game under the constraints of a given synthesized strategy.

Consider the jobshop scheduling problem shown in Fig. 1 which models a number of persons sharing a newspaper. Each task process reads a section of the paper, whereas only one person can read a particular section at a time. Each reader wants to read the newspaper in different orders, and the stochastic environment chooses how long it takes to read each section. This makes the problem a problem of finding a strategy, rather than finding a static scheduler as in the classical jobshop scheduling problem.

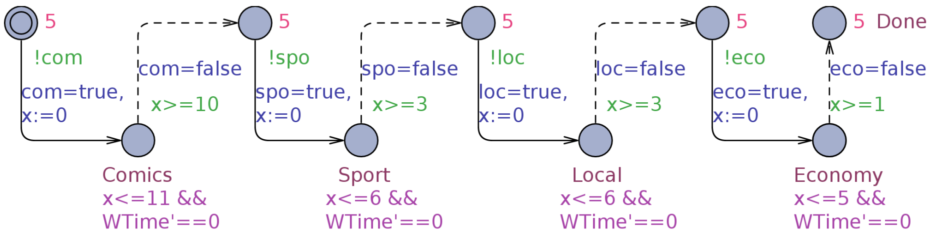


Fig. 1. UPPAAL STRATEGO template of a single person reading a newspaper

[1] UPPAAL STRATEGO is available at `http://people.cs.aau.dk/~marius/stratego/`

Figure 1 shows a stochastic priced timed game (SPTG) which models one person reading the newspaper. The circles are locations and the arrows are transitions. The solid arrows are transitions controlled by the controller and the dashed are transitions controlled by the stochastic environment. The model reflects the reading of the four sections in the preferred order (here comics, sport, local and economy) for the preferred amount of time. In the top locations the person is waiting for the next section to become available; here four Boolean variables are used to ensure mutex on the reading of a section. In the bottom locations, the person is reading the particular section for a duration given by a uniform distribution on the given interval, e.g. [10,11] for our person's reading of sport. The stopwatch `WTime` is only running in the waiting locations thus effectively measuring the accumulated time when the person is waiting to read. Given a complete model with several persons constantly competing for the sections, we are interested in synthesizing strategies for several multi-objectives, e.g. synthesize a strategy ensuring that all persons have completed reading within 100 minutes, and then minimize the expected waiting time for our preferred person.

2 Games, Automata and Properties

Using the features of Uppaal Stratego we can analyze the SPTG in Fig. 1. Internally, Uppaal Stratego has different models and representations of strategies, an overview of these and their relations are given in Fig. 2. The model seen in Fig. 1 is a SPTG, as `WTime` is a cost function or price with location dependent rate (here 0 or 1), and we assume that environment takes transitions according to a uniform distribution over time.

As shown in Fig. 2 we can abstract a SPTG into a timed game (TGA). This abstraction is obtained simply by ignoring the prices and stochasticity in the model. Note that since prices are observers, this abstraction does not affect the possible behavior of the model, but merely forgets the likelihood and cost of various behaviors. The abstraction maps a 1½-player game, where the opponent is stochastic into a 2-player game with an antagonistic opponent.

Given a TGA ($\mathcal{G}$) we can use Uppaal Tiga to synthesize a strategy σ (either deterministic or non-deterministic). This strategy can, when put in parallel with the TGA, $\mathcal{G}|\sigma$, be model-checked in the same way as usual in Uppaal. We

$$\begin{tikzcd}
\text{Timed Game} & & \text{Strategy} & \text{(Statistical) Model Checking Timed Automata} \\
\mathcal{G} \arrow[rr, "\text{Synthesis}\ \textsc{Uppaal Tiga}", "\phi"'] & & \sigma \arrow[r, dashed] \arrow[dl, dashed] & \mathcal{G}|\sigma \\
\mathcal{P} \arrow[u, "\text{Abstraction}"] \arrow[r, dashed] & \mathcal{P}|\sigma \arrow[r, "\phi", "\text{Synthesis}"'] & \sigma^{\circ} \arrow[r, dashed] & \mathcal{P}|\sigma^{\circ} \\
\text{Stochastic Priced Timed Game} & & \text{Strategy} & \text{Statistical Model Checking Stochastic Priced Timed Automata}
\end{tikzcd}$$

Fig. 2. Overview of models and their relations. The lines show different actions. The dashed lines show that we use the object.

can also use the strategy in a SPTG $\mathcal{P}$, and obtain $\mathcal{P}|\sigma$. Under a strategy it is possible to do statistical model checking (estimation of probability and cost, and comparison), which enables us to observe the behavior and performance of the strategy when we assume that the environment is purely stochastic. This also allows us to use to use prices under σ, even though they were not considered in the synthesis of σ. From both $\mathcal{P}$ and $\mathcal{P}|\sigma$ learning is possible using the method proposed in [9]. The learning algorithm uses a simulation based method for learning near-optimal strategies for a given price metric. If σ is the most permissive strategy guaranteeing some goal, then the learning algorithm can optimize under this strategy, and we will get a strategy σ° which is near-optimal but still has the guarantees of σ. As the last step we can construct $\mathcal{P}|\sigma^\circ$, which we can then do statistical model checking on.

3 Strategies

In UPPAAL STRATEGO we operate three different kinds of strategies, all memoryless. *Non-deterministic strategies* are strategies which give a *set* of actions in each state, with the most permissive strategy – when it exists – offering the largest set of choices. In the case of timed games, most permissive strategies exist for safety and time-bounded reachability objectives. *Deterministic strategies* give *one* action in each state. *Stochastic strategies* give a *distribution* over the set of actions in each state. Fig. 3 shows how strategies are generated and used. For generating strategies, we can use UPPAAL TIGA or the method proposed in [9] on SPTGs. UPPAAL TIGA generates (most permissive) *non-deterministic* or *deterministic* strategies. The method proposed in [9] generates strategies which are deterministic. A strategy generated with UPPAAL STRATEGO can undergo different investigations: model checking, statistical model checking and learning. Learning consume non-deterministic strategies (potentially multiple actions per state) and may produce a deterministic one by selecting a single action for each state, such that the final deterministic strategy is optimized towards some goal. Figure 3 shows that currently it is possible to model check only under symbolically synthesized strategies (as opposed to optimized ones) as symbolic model checking requires the strategy to be represented entirely in terms of zones (constraint systems over clock values and their differences). Statistical model checking can only be done under stochastic strategies. All deterministic strategies can be thought of as stochastic by assigning a probability of 1 to the *one* choice.

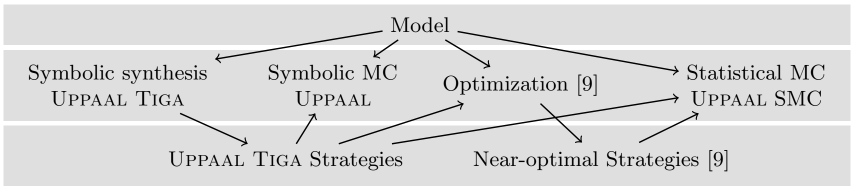


Fig. 3. Overview of algorithms and data structures in UPPAAL STRATEGO

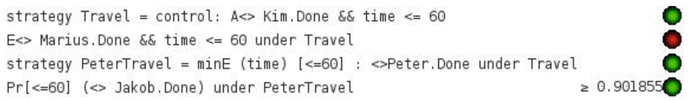
To evaluate non-deterministic strategies statistically we applying a stochastic uniform distribution over the non-deterministic choices.

4 Query Language

We let strategies become first class citizens by introducing strategy assignment `strategy S =` and strategy usage `under S` where `S` is an identifier. These are applied to the queries already used in Uppaal, Uppaal Tiga and Uppaal SMC as well as those proposed in [9]. An overview of these queries is given in Table 1. Notice that we changed the syntax of the queries presented in [9]. Recall the example with the four authors sharing a newspaper as presented in Fig. 1. We compute a strategy for Kim to reach his plane within one hour on line 1 in Fig. 4. Respecting this, we find that Marius cannot join, as the query on line 2 is not satisfied. Instead, we optimize that Peter joins in on line 3 (`[<=60]` is a bound on how long the simulations we learn from used can be). Finally, line 4 estimates that Jakob is done with probability $\geq$0.9 under Peter's optimizations.

Table 1. Types of queries

Uppaal	Safety	`A[] prop under NS`
	Liveness	`A<> prop under NS`
Tiga	Guarantee objective	`strategy NS = control: A<> prop`
	Guarantee objective	`strategy NS = control: A[] prop`
SMC	Evaluation	`Pr[bound](<> prop) under SS`
	Expected	`value E[bound;int](min: prop) under SS`
	Simulations	`simulate int [bound]{expr1,expr2} under SS`
[9]	Minimize objective	`strategy DS = minE (expr) [bound]: <> prop under NS`
	Maximize objective	`strategy DS = maxE (expr) [bound]: <> prop under NS`



```
strategy Travel = control: A<> Kim.Done && time <= 60
E<> Marius.Done && time <= 60 under Travel
strategy PeterTravel = minE (time) [<=60] : <>Peter.Done under Travel
Pr[<=60] (<> Jakob.Done) under PeterTravel                    ≥ 0.901855
```

Fig. 4. Uppaal Stratego queries and results for the model in Fig. 1

References

1. Asarin, E., Maler, O., Pnueli, A.: Symbolic controller synthesis for discrete and timed systems. In: Antsaklis, P.J., Kohn, W., Nerode, A., Sastry, S.S. (eds.) HS 1994. LNCS, vol. 999, Springer, Heidelberg (1995)
2. Behrmann, G., Cougnard, A., David, A., Fleury, E., Larsen, K.G., Lime, D.: UPPAAL-tiga: Time for playing games! In: Damm, W., Hermanns, H. (eds.) CAV 2007. LNCS, vol. 4590, pp. 121–125. Springer, Heidelberg (2007)

3. Behrmann, G., David, A., Larsen, K.G., Håkansson, J., Pettersson, P., Yi, W., Hendriks, M.: Uppaal 4.0. In: Proceedings of the 3rd International Conference on the Quantitative Evaluation of Systems, QEST 2006, IEEE Computer Society, Washington, DC (2006)
4. Bulychev, P.E., David, A., Larsen, K.G., Mikučionis, M., Poulsen, D.B., Legay, A., Wang, Z.: UPPAAL-SMC: statistical model checking for priced timed automata. In: Proceedings 10th Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL 2012, Tallinn, Estonia. EPTCS, vol. 85 (March 2012)
5. 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)
6. Cassez, F., Jessen, J.J., Larsen, K.G., Raskin, J.-F., Reynier, P.-A.: Automatic synthesis of robust and optimal controllers – an industrial case study. In: Majumdar, R., Tabuada, P. (eds.) HSCC 2009. LNCS, vol. 5469, pp. 90–104. Springer, Heidelberg (2009)
7. Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., Tacchella, A.: NuSMV 2: An openSource tool for symbolic model checking. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, p. 359. Springer, Heidelberg (2002)
8. David, A., Fang, H., Larsen, K.G., Zhang, Z.: Verification and performance evaluation of timed game strategies. In: Legay, A., Bozga, M. (eds.) FORMATS 2014. LNCS, vol. 8711, pp. 100–114. Springer, Heidelberg (2014)
9. David, A., Jensen, P.G., Larsen, K.G., Legay, A., Lime, D., Sørensen, M.G., Taankvist, J.H.: On time with minimal expected cost! In: Cassez, F., Raskin, J.-F. (eds.) ATVA 2014. LNCS, vol. 8837, pp. 129–145. Springer, Heidelberg (2014)
10. David, A., Larsen, K.G., Legay, A., Mikučionis, M.: Schedulability of herschel-planck revisited using statistical model checking. In: Margaria, T., Steffen, B. (eds.) ISoLA 2012, Part II. LNCS, vol. 7610, pp. 293–307. Springer, Heidelberg (2012)
11. Gibson-Robinson, T., Armstrong, P., Boulgakov, A., Roscoe, A.W.: FDR3 — A modern refinement checker for CSP. In: Ábrahám, E., Havelund, K. (eds.) TACAS 2014 (ETAPS). LNCS, vol. 8413, pp. 187–201. Springer, Heidelberg (2014)
12. Jessen, J.J., Rasmussen, J.I., Larsen, K.G., David, A.: Guided controller synthesis for climate controller using UPPAAL TIGA. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 227–240. Springer, Heidelberg (2007)
13. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 585–591. Springer, Heidelberg (2011)
14. Maler, O., Pnueli, A., Sifakis, J.: On the synthesis of discrete controllers for timed systems. In: Mayr, E.W., Puech, C. (eds.) STACS 1995. LNCS, vol. 900, Springer, Heidelberg (1995)
15. Zhao, H., Zhan, N., Kapur, D., Larsen, K.G.: A "hybrid" approach for synthesizing optimal controllers of hybrid systems: A case study of the oil pump industrial example (2012)