The 2016 AAAI Fall Symposium Series:
Cross-Disciplinary Challenges for Autonomous Systems
Technical Report FS-16-03

Probabilistic Verification for Cognitive Models: Controller Synthesis and Model Evaluation

Sebastian Junges and **Joost-Pieter Katoen**
RWTH Aachen University, Germany

Nils Jansen and **Ufuk Topcu**
The University of Texas at Austin, USA

Abstract

Many robotics applications and scenarios that involve interaction with humans are safety or performance critical. A natural path to assessing such notions is to include a cognitive model describing typical human behaviors into a larger modeling context. In this work, we set out to investigate a combination of such a model with formal verification. We present a general and flexible framework utilizing methods from probabilistic model checking and discuss current pitfalls. We start from information about typical behavior, obtained from generalizing specific scenarios by the usage of inverse reinforcement learning. We translate this information in order to define a formal model exhibiting stochastic behavior (whenever significant data is present) or nondeterminism (if the model is underspecified or no significant data is present) that can be analyzed. This model for a human can be combined with a robot model by using standard parallel composition. The benefit is manyfold: First, safe or optimal strategies for involved robots regarding a human can be synthesized depending on the given model. In general, verification can determine if such benign strategies are even possible. Furthermore, the cognitive model itself can be analyzed with respect to possible unnatural behaviors; thereby feedback to developers of such models is provided. We evaluate and describe our approaches by means of a well-known model for visiomotor tasks and provide a framework that can readily incorporate other models.

Methodology

In this paper, we give an intuitive overview on our methodology and provide some necessary background. We consider shared autonomy settings that roughly incorporate a human and potentially multiple controllable robots. The key issue is that optimal or safe robot strategies are dependent on *human behavior*. In order to enable formal reasoning, one can make use of a *human model*, from which data about typical behavior in specific scenarios shall be collected. Note that defining a closed-form formal model is already hard for static environments because of possibly very large state spaces. Such a model becomes even more intractable when considering dynamic environments which give rise to possibly infinite state spaces.

In the past, it has been investigated, how data about *typical human behavior* can be obtained by *reinforcement learning* (RL) (Sutton and Barto 1998). Basically, RL describes algorithms addressing the optimal control problem through learning, i.e., an agent learns how to solve a task based on accumulated experience while interacting with an environment. Costs and rewards of actions with respect to a certain measure of success are thereby quantified; resulting in a quantitative description of possible actions. So-called *Q-learning* maintains this information in form of Q-values over actions, stored in a Q-table. RL is typically performed in several *episodes* to approximate the globally optimal behavior, i. e., actions are chosen to improve previous choices with respect to already learned global information in each episode.

RL connects to typical human behavior in the following way: A human does not select globally optimal actions, but rather those which obtain locally high rewards or low cost, corresponding to a high Q-value. Executed for several episodes, globally optimal values are approximated by humans, cf. (Barto 1995; Daw and Doya 2006). When such a technique is used to collect behavioral data, randomness and noise needs to be factored in. In particular, one cannot assume that every human always takes the action with the highest value. Therefore, a human is assumed to make choices based on a *distribution over the actions*, where the distribution is based on the Q-values (e.g. by an applied softmax-function).

To illustrate such behavior using an example, consider Figure 1 which is taken with permission from Ballard from (Rothkopf and Ballard 2013). The setting is as follows: A human walks down a sidewalk containing *features* of different kinds (obstacles, litter, and a pathway) and is asked to attend to three tasks:

(i) avoid obstacles,

(ii) collect litter,

(iii) and follow a specific path on the walkway.

Figure 1 shows an avatar inside a simulation environment for our scenario. We see a division of the features of the given scenario into features relevant for the specific tasks. The arrows depict point for each of the tasks to the closest relevant feature. This *modularization* can be utilized to first describe the behavior if only one task would be given, while the action values are obtained following the results of several episodes of RL for this objective. The so obtained Q-table maps a relative position of the human with respect to relevant

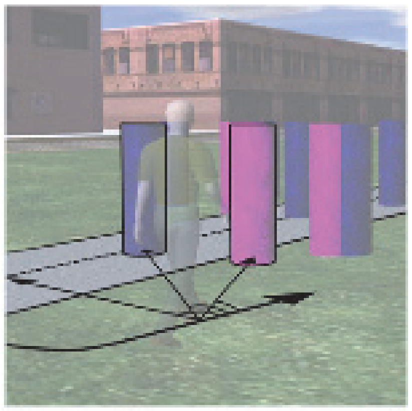

Figure 1: Visiomotor setting with different tasks while walking down a sidewalk shown in a simulation environment. The state space is divided into the modular tasks (i) approach targets (blue), (ii) avoid obstacles (purple), and (iii) follow walkway (gray).

features in the environment and an action-choice to an action value.

Notice, that in this example setting a particular human might have a different interpretation of the *objectives*, for instance one might prefer to collect as much litter as possible while a more risk-averse person rather tries to avoid obstacles at all cost. If such different objectives of humans can be identified, data— interpreted as probability distributions over possible actions— *with respect to each individual objective* needs to be collected as explained above.

The question is then how these different data sets can be combined formally by, e. g., assigning *weights* to different actions to reflect a preference over different objectives. One approach uses the assumption of modularity of the human mind, see for instance (Pinker 1999). Modularity now means that a human assigns weights to possible actions according to his/her preference of individual objectives. The assumption is, that the human actually wants to maximize the objective, e. g., collecting as much litter as possible. Put differently, the choice of actions is assumed to be according to RL and the reward or cost of a chosen action is inferred under this assumption. Such methods are called *inverse reinforcement learning* (IRL) (Ng, Russell, and others 2000). Given individual Q-tables for different objectives and obtained weights on these objectives, the result is a multi-dimensional Q-table over the relative position with respect to several relevant features, mapping this relative position and action-choice to an action-value for the *combined objectives*. This is for instance employed in (Rothkopf and Ballard 2013).

Our goal is to use these weighted Q-tables together with a description of the environment to obtain a formal description of typical human behavior *within this environment*. Note that the variability or noise in the action-values give rise to probability distributions over actions inside the environment. In addition to the weights over actions the model might also employ *nondeterminism* for two reasons. First, often the discretized state set is an abstraction, and different behaviors are grouped within one state, which leads to *under-specification*. Second, not all combinations above might have been considered in the experiments, which might lead to a *lack of data*. In such cases, it seems reasonable to assume the worst-case and not assume any probability distribution over the actions.

The underlying model for the human can be specified by a Markov decision process (MDP) (Puterman 1994). An MDP is a transition system having *nondeterministic choices*[1]. Each choice is associated with a probability distribution over successor states. Starting from some initial state, a run of an MDP is thus based on a series of choices that all result in a probabilistic determination of the next state. If all possible nondeterministic choices in an MDP are resolved—by an entity called scheduler or strategy—the resulting system is fully stochastic and a probability measure can be associated to events like finally reaching a certain state.

In our setting, we want to compute a strategy of the human that induces a certain measure of success. We also want to account for best-case or worst-case behaviors of the human. Naturally, the method of choice is probabilistic model checking, cf. (Baier and Katoen 2008), referring to the question if a given property, like "The probability of reaching a bad state shall be lower than 10%", is satisfied or not. If yes, one is returned a strategy that induces such a satisfactory event.

The necessary techniques are provided by state-of-the art probabilistic model checkers like `PRISM` (Kwiatkowska, Norman, and Parker 2011), `ISCAS-MC` (Hahn et al. 2014) or `PROPhESY` (Dehnert et al. 2015). In case there is no such benign strategy, one is interested in diagnostic information, e. g. in the form of *probabilistic counterexamples* (Ábrahám et al. 2014). Tool-support is given by `DiPro` (Aljazzar et al. 2011) or `COMICS` (Jansen et al. 2012).

Assume now we have such an MDP representing typical behavior of humans in a certain specific context, i. e., inside the given environment. This model is merged with a given *model of robot movement*, adding a second level of nondeterminism. The resulting model is then called a 2-player stochastic game. Model checking enables to resolve any nondeterminism in the possible human actions by assuming worst-case behavior of the robot. Such an analysis can be performed by (Chen et al. 2013a).

Note that as with any model-based approach, our result is as most as good as the model, while due to the formal guarantees we can state that our result is in fact as good as the model. To improve the *quality of models*, we propose to utilize counterexamples. In particular, a technique providing feedback on errors based on the modeling language (Wimmer et al. 2015) might provide feedback to the model learning block (yielding a guided RL idea), as the feedback focusses on the most relevant parts of the human behavior with respect to our robot. On top of that, one can employ *model repair* (Bartocci et al. 2011; Chen et al. 2013b; Pathak et al. 2015), where a model is automatically repaired towards satisfaction of a property. The changes can reflect what changes need to be made to the original model.

[1]This is often called actions; however these actions do not correspond to the actions the human takes in our setting.

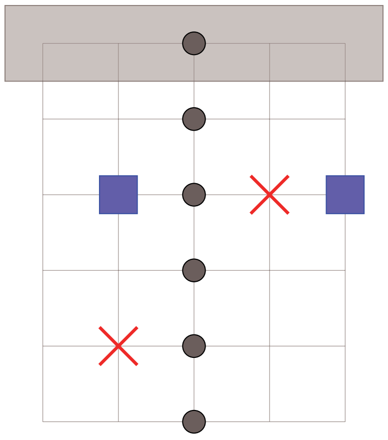

Figure 2: Graphical representation of our gridworlds

Case Study

To show the applicability of the technique described above, we chose the specific scenario based on the original case study from (Rothkopf and Ballard 2010). Recall that we have a human subject who is to collect litter while avoiding obstacles when following a specific path. This is depicted in Figure 1. Consider moreover Figure 2, where the *path to follow* is abstracted by waypoints depicted as black dots. *Obstacles to avoid* are red crosses and the blue squares indicate *litter to collect*. These three entities are called *features*.

Addressing the problem of potentially very large state spaces, in (Rothkopf and Ballard 2013) an approach was suggested which was based on *modular IRL*, assuming that behavior is organized in a modular way. Basically, separate representations of individual tasks are available and actions influence these tasks individually. As described above, our starting point is this methodology, based on an cognitive architecture for visiomotor tasks as in Figure 2, first presented in (Rothkopf and Ballard 2010).

Moreover, the methodology used in (Rothkopf and Ballard 2013) leaves room for nondeterminism in movements, either by inherent underspecification in the sense that only the closest litter and obstacles is accounted for (which is not uniquely defined in the grid), or because there is very low confidence on which movement to take.

While the underlying state space is indeed shown to grow exponentially in the *number of features* (obstacles or litter) within the scenario, the encoding in the PRISM-format can be done in a cubic fashion. Notice that the parallel composition of modules – typically used for parallel (modular) processes – does not help to encode the human behavior due to the nature of the modularity considered here. Domain specific insights allow us to cut-off the model when specific conditions are met, with automatic error bounds obtained. This ensures that both the construction of the model as well as the final state space can be managed towards the originally considered grid-sizes. First experiments show that for some scenarios the underspecification is serious while for others this is only a minor issue. An important insight is that the resulting MDP graph has an irregular structure. This irregularity renders the internal symbolic representation of the MDP (Baier et al. 1997; Parker 2002) larger than for the typical benchmarks from the probabilistic verification community as for instance given by the PRISM benchmark suite (Kwiatkowska, Norman, and Parker 2012). This means that the memory consumption for models typically referred to as medium-sized (10^6 - 10^7 states) already takes a significant amount of computation resources. Moreover, typical reduction strategies such as bisimulation (Katoen et al. 2007) are mostly ineffective. However, by applying aforementioned optimizations on the model construction and by selecting the best model checking configurations in terms of solving method and state space representations, the scalability is significantly improved.

We conclude that while the methodology is promising, its practical realization suffers from scalability issues on several levels (translation, encoding, internal representation). Tailored techniques for the representation or abstraction are promising future directions to remedy these problems.

Acknowledgements

This work has been partly funded by the awards AFRL # FA9453-15-1-0317, ARO # W911NF-15-1-0592 and ONR # N00014-15-IP-00052. We want to thank Dana H. Ballard, Mary M. Hayhoe, and Matt Tong for their kind providing of data, resources, and advice for this work.

References

Ábrahám, E.; Becker, B.; Dehnert, C.; Jansen, N.; Katoen, J.; and Wimmer, R. 2014. Counterexample generation for discrete-time markov models: An introductory survey. In *SFM*, volume 8483 of *Lecture Notes in Computer Science*, 65–121. Springer.

Aljazzar, H.; Leitner-Fischer, F.; Leue, S.; and Simeonov, D. 2011. DiPro – A tool for probabilistic counterexample generation. In *Proc. of SPIN*, volume 6823 of *LNCS*, 183–187. Springer.

Baier, C., and Katoen, J.-P. 2008. *Principles of Model Checking*. The MIT Press.

Baier, C.; Clarke, E. M.; Hartonas-Garmhausen, V.; Kwiatkowska, M. Z.; and Ryan, M. 1997. Symbolic model checking for probabilistic processes. 430–440.

Barto, A. G. 1995. 11 adaptive critics and the basal ganglia. *Models of information processing in the basal ganglia* 215.

Bartocci, E.; Grosu, R.; Katsaros, P.; Ramakrishnan, C.; and Smolka, S. A. 2011. Model repair for probabilistic systems. In *Proc. of TACAS*, volume 6605 of *LNCS*. Springer. 326–340.

Chen, T.; Forejt, V.; Kwiatkowska, M.; Parker, D.; and Simaitis, A. 2013a. PRISM-games: A model checker for stochastic multi-player games. In *Proc. 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'13)*, volume 7795 of *LNCS*, 185–191. Springer.

Chen, T.; Hahn, E. M.; Han, T.; Kwiatkowska, M.; Qu, H.; and Zhang, L. 2013b. Model repair for Markov decision processes. In *Proc. of TASE*, 85–92. IEEE CS.

Daw, N. D., and Doya, K. 2006. The computational neurobiology of learning and reward. *Current opinion in neurobiology* 16(2):199–204.

Dehnert, C.; Junges, S.; Jansen, N.; Corzilius, F.; Volk, M.; Bruintjes, H.; Katoen, J.-P.; and Abraham, E. 2015. Prophesy: A probabilistic parameter synthesis tool. In *Proc. of CAV*, volume 9206, 214–231.

Hahn, E. M.; Li, Y.; Schewe, S.; Turrini, A.; and Zhang, L. 2014. iscasMc: A web-based probabilistic model checker. In *Proc. of FM*, volume 8442 of *LNCS*, 312–317. Springer.

Jansen, N.; Ábrahám, E.; Volk, M.; Wimmer, R.; Katoen, J.-P.; and Becker, B. 2012. The COMICS tool – Computing minimal counterexamples for DTMCs. In *Proc. of ATVA*, volume 7561 of *LNCS*, 349–353. Springer. (to appear).

Katoen, J.; Kemna, T.; Zapreev, I. S.; and Jansen, D. N. 2007. Bisimulation minimisation mostly speeds up probabilistic model checking. In *TACAS*, volume 4424 of *Lecture Notes in Computer Science*, 87–101. Springer.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2011. PRISM 4.0: Verification of probabilistic real-time systems. In *Proc. of CAV*, volume 6806 of *LNCS*, 585–591. Springer.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2012. The PRISM benchmark suite. In *Proc. of QEST*, 203–204. IEEE CS.

Ng, A. Y.; Russell, S. J.; et al. 2000. Algorithms for inverse reinforcement learning. In *Icml*, 663–670.

Parker, D. 2002. *Implementation of Symbolic Model Checking for Probabilistic Systems*. Ph.D. Dissertation, University of Birmingham.

Pathak, S.; Ábrahám, E.; Jansen, N.; Tacchella, A.; and Katoen, J. 2015. A greedy approach for the efficient repair of stochastic models. In *NFM*, volume 9058 of *Lecture Notes in Computer Science*, 295–309. Springer.

Pinker, S. 1999. How the mind works. *Annals of the New York Academy of Sciences* 882(1):119–127.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons.

Rothkopf, C. A., and Ballard, D. H. 2010. Credit assignment in multiple goal embodied visuomotor behavior. *Embodied and grounded cognition* 217.

Rothkopf, C. A., and Ballard, D. H. 2013. Modular inverse reinforcement learning for visuomotor behavior. *Biological cybernetics* 107(4):477–490.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Wimmer, R.; Jansen, N.; Vorpahl, A.; Ábrahám, E.; Katoen, J.; and Becker, B. 2015. High-level counterexamples for probabilistic automata. *Logical Methods in Computer Science* 11(1).