WiP Abstract: Interactive Explanation for Planning-Based Systems

Ellin Zhao
Dept. of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA, USA
ellinzhao@berkeley.edu

Roykrong Sukkerd
Institute for Software Research
Carnegie Mellon University
Pittsburgh, PA, USA
rsukkerd@cs.cmu.edu

ABSTRACT

As Cyber-Physical Systems (CPSs) become more autonomous, it becomes harder for humans who interact with the CPSs to understand the behavior of the systems. Particularly for CPSs that must perform tasks while optimizing for multiple quality objectives and acting under uncertainty, it can be difficult for humans to understand the system behavior generated by an automated planner. This work-in-progress presents an approach at clarifying system behavior through interactive explanation by allowing end-users to ask *Why* and *Why-Not* questions about specific behaviors of the system, and providing answers in the form of contrastive explanation.

CCS CONCEPTS

• **Human-centered computing** → *User interface design.*

KEYWORDS

Interactive Explanation, Explainable Planning, Markov Decision Process Planning

ACM Reference Format:
Ellin Zhao and Roykrong Sukkerd. 2019. WiP Abstract: Interactive Explanation for Planning-Based Systems. In *ICCPS '19: ACM/IEEE International Conference on Cyber-Physical Systems, April 16–18, 2019, Montreal, QC, Canada.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3302509.3313322

1 INTRODUCTION

As the use of artificial intelligence in CPSs increases and systems become more autonomous, it becomes harder for humans who interact with the systems to understand the rationale behind system behavior. In many application domains of CPSs, the systems must perform tasks while optimizing for multiple quality objectives and operating under uncertainty. Such a problem can be formulated as a Markov decision process (MDP) and solved using an automated planner. Our work concerns making the behaviors of these systems more transparent using interactive explanation.

The explanation is directed towards end-users of these autonomous systems who do not know the technicalities of the systems or planning. Through explanation, end-users should gain awareness of how the systems make decisions in terms of the domain-specific qualities of concern and gain confidence that the system is making the best decisions in terms of those concerns.

Users can ask *Why* and *Why-Not* questions about the system's behavior and provide answers using a contrastive explanation approach in which we point out the consequence of the counterfactuals in the question [4]. Explanations describe why the system makes certain choices, in terms of the consequences or properties of those choices, and explain how the choices affect the goals and objectives of the system.

2 APPROACH

Our method of interactive explanation applies to MDPs with multi-objective cost functions, characterized by $\langle S, A, Pr, C, s_0 \rangle$ and an optimal policy, π^* [1] [5]. Using linear scalarization, the multiple cost functions are combined into one, resulting in $C(s, a, s') = k_1 \cdot C_1(s, a, s') + ... + k_n \cdot C_n(s, a, s')$ [2]. We demonstrate a tool in the domain of robot navigation planning, in which a self-adaptive robot plans a route from a start state to a goal, while optimizing for various objectives such as travel time, level of intrusiveness and number of collisions.

Explanation generation

When a user expects a system to behave differently than it does in π^*, they will have questions. A user might ask "Why does the system do X?", which means that X occurs in π^*, but the user does not expect X to happen. In this case, we will show the user that not doing X is less preferred to doing X, because of the consequences on the objective values (e.g. travel time, energy consumed, intrusiveness). Similarly, when a user asks, "Why does the system not do Y?", it is

ICCPS '19, April 16–18, 2019, Montreal, QC, Canada

ACM ISBN 978-1-4503-6285-6/19/04.
https://doi.org/10.1145/3302509.3313322

expected that Y occurs in π^*, but it does not. We will show the user that doing Y is less preferred to not doing Y in terms of consequences on objective values.

To generate contrastive explanations, we demonstrate what happens if the system behaves as the user expects. User questions are translated into constraints on the planning problem, resulting in a new MDP formulation that we pass into the planner to get an alternate policy, π. Using probabilistic model checking of π^* and π [3], we obtain the objective values for both policies and use these to show the trade-off among competing optimization objectives [7]. We justify the behavior of π^* by explaining that the trade-offs made in π are less preferable and the trade-offs in π^* are better than the user initially thought.

User Interface

Users ask questions using *Why* and *Why-Not* question templates that capture their expectations for system behavior. A *Why* question is formulated as "why does the system do X?" and translates to the expectation that the system should not do X. Similarly, a *Why-Not* question, "why does the system not do Y?", reflects the expectation that the system should do Y. In the templates, the user fills in X and Y by specifying an action and optionally, the conditions for that action.

The template takes the form of "why does the robot (not) do a when P?", where a is an action and P is a predicate on the state variables. As such, X and Y target an action in any context or a conditional action. For example, a user can ask "why does the robot recharge at location 5 when energy > 50?" to describe a conditional action or can simply ask "why does the robot recharge?". For the former question, $a = recharge$ and $P = (location \in \{5\}) \land (energy > 50)$.

"Why" Questions

The process for translating questions into constraints depends on the type of question that is asked. Consider translating "why does the robot do X?", where X is specified by (a, P). To translate this into a constraint, we need to disallow a from occurring when P is satisfied. If the predicate is empty, we can remove a from the action set, resulting in a new action set, A_1. If the predicate is not empty, we add $\neg P$ to the existing preconditions for a. Solving the modified MDP, with either A_1 or the modified preconditions, yields an alternate policy, π, in which X does not occur. π^* and π are then contrasted in terms of their objective values.

For example, if the user asks, "why does the robot use full speed at location 13?", an explanation might be: "The robot plans to use full speed at location 13 because although using half speed would consume less energy (by 100 mWh), it would take more travel time (by 0.5 minutes) to get to the goal. The decrease in energy consumption is not worth the increase in travel time".

"Why-Not" Questions

When handling the question "why does the robot not do Y?", where $Y = (a^*, P)$, we need to force a^* to occur under P. One approach is to create an additional cost function, $C^*(s, a, s')$ that assigns 1 when $a = a^*$ and $P(s) = true$, and create the constraint that the expected total cost of the policy is greater than some value; that is, $E[C^*] \geq k$. As such, k is the expected number of occurrences of a under P in the policy. In some cases, the value of k unknown because we cannot anticipate what number of steps in the robot's policy will be affected by the question. When k is unknown, we need to use different constraints - one option is to use temporal logic to formulate the behavioral constraints of the robot [6]. Handling a subset of *Why-Not* questions is still a work-in-progress.

Handling questions about unconditional actions ("why does the robot not recharge?") is possible using the constraint that $E[C^*] \geq 1$. Planning with C^* yields π, which we contrast with π^* to highlight the differences between the trade-offs in terms of objective values.

ACKNOWLEDGMENTS

This work is supported in part by award N00014172889 from the Office of Naval Research, award H9823018D0008 from the NSA, and award FA8750-16-2-0042 from AFRL and DARPA. Any views, opinions, findings and conclusions expressed in this material are those of the authors, and do not necessarily reflect the views of the sponsoring agencies.

REFERENCES

[1] Dimitri P. Bertsekas and John N. Tsitsiklis. 1991. An Analysis of Stochastic Shortest Path Problems. *Math. Oper. Res.* 16, 3 (Aug. 1991).

[2] Massimiliano Caramia and Paolo Della-Olmo. 2008. *Multi-objective Management in Freight Logistics: Increasing Capacity, Service Level and Safety with Optimization Algorithms.*

[3] Vojtěch Forejt, Marta Kwiatkowska, Gethin Norman, and David Parker. 2011. Automated verification techniques for probabilistic systems. In *International School on Formal Methods for the Design of Computer, Communication and Software Systems.* Springer, 53–113.

[4] Tim Miller. 2017. Explanation in Artificial Intelligence: Insights from the Social Sciences. *CoRR* abs/1706.07269 (2017). arXiv:1706.07269

[5] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, USA.

[6] Jonathan Sprauel, Florent Teichteil-Königsbuch, and Andrey Kolobov. 2014. Saturated Path-constrained MDP: Planning Under Uncertainty and Deterministic Model-checking Constraints. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI'14).* AAAI Press, 2367–2373.

[7] Roykrong Sukkerd, Reid Simmons, and David Garlan. 2018. Towards Explainable Multi-objective Probabilistic Planning. In *Proceedings of the 4th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS '18).* ACM, New York, NY, USA, 19–25.