Student Project:

Beyond Shortest Paths: Maximizing Rewards in Classical Planning

Supervisors: David Speck Jendrik Seipp

Planning is the art and practice of thinking before acting and is considered a central part of artificial intelligence. In classical planning, the objective is to derive a course of actions, i.e., a plan, that allows an intelligent agent to move from any situation it finds itself in to one that satisfies its goals. Most commonly, planning assumes that each action has a specific cost, and the goal is to determine a cost-optimal plan – a sequence of actions whose total cost is minimized. If all actions have the same unit cost, the problem of finding such a cost-optimal plan is equivalent to finding a shortest path.

However, there is a corpus of real-world problems that are best described by actions that induce rewards and the goal of maximizing the sum over those rewards. Problems with a reward maximization property range from applications in computational linguistics and bioinformatics (e.g., "Longest common subsequence"[1]), to reconfiguration of 24-7 systems such as power grids (e.g., "Reconfiguring Independent Sets"[1]), to the theory of error-correcting codes (e.g., "Snake in the box" or "Coil in the box"[2]).

Maximizing Rewards. In principle, it is possible to cast a maximization problem with rewards to a minimization problem with costs by negating the costs. However, finding a plan that actually minimizes plan costs with negative cost values poses serious problems in practice, as most modern planning techniques cannot be readily applied to this problem. This is due to the nature of the underlying problem: finding a longest plan is different from finding a shortest plan.

[1] https://en.wikipedia.org/wiki/Longest_common_subsequence

[2] https://en.wikipedia.org/wiki/Snake-in-the-box

An important condition is that a plan must be loopless, i.e., each world state can be visited only once during the execution of the plan. Given this, when searching for the longest plan within a state space, it's not necessarily true that the solution can be constructed from optimal solutions to its subproblems [2]. In other words, if a longest plan to reach a state s_* is given, the prefix of that plan that lead to a state s may not necessarily represent the longest plan to reach s. This fundamental difference between finding a longest and shortest plan is also highlighted by the fact that finding the shortest path in a compact state space representation is PSPACE-complete [3], while finding the longest path in the same representation is NEXPTIME-hard [4], which is considered much harder.

Another interesting observation is that there are close connections to deterministic Markov decision processes (e.g., [5]) and oversubscription planning, where the goal is to determine an end state that maximizes a utility function [6, 7].

Project Outline. The overall plan of this project can be roughly broken down into four tasks.

- T1: Familiarize yourself with classical planning and the problem of finding the longest path in a state space. Formally describe and define a planning task with rewards and the (decision) problem of finding a reward-optimal plan or, in the case of a unit cost setting, a longest plan.
- T2: Create a benchmark set for reward/plan-length maximization by modeling practically relevant and exemplary domains, such as "Snake in the box", "Coil in the box", "Longest common subsequence", and perhaps others from the literature on deterministic Markov decision processes.
- T3: Implement a baseline explicit search algorithm in a modern planner like SymK [8] or Fast Downward [9], such as A*, that searches over paths instead of states, iteratively finding longer plans.
- T4: Conduct experiments to compare and analyze the newly implemented A* approach for discovering high-reward plans on the compiled benchmark set with an existing symbolic search approach.

If you are interested in this project of teaching a machine to think before it acts, please feel free to contact us.

References

[1] Remo Christen, Salomé Eriksson, Michael Katz, Christian Muise, Alice Petrov, Florian Pommerening, Jendrik Seipp, Silvan Sievers, and David Speck. “PARIS: Planning Algorithms for Reconfiguring Independent Sets”. In: *Proceedings of the 26th European Conference on Artificial Intelligence (ECAI 2023)*. 2023.

[2] Roni Stern, Scott Kiesel, Rami Puzis, Ariel Felner, and Wheeler Ruml. “Max Is More than Min: Solving Maximization Problems with Heuristic Search”. In: *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 4324–4330.

[3] Tom Bylander. “The Computational Complexity of Propositional STRIPS Planning”. In: *Artificial Intelligence* 69.1–2 (1994), pp. 165–204.

[4] Christos H. Papadimitriou and Mihalis Yannakakis. “A Note on Succinct Representations of Graphs”. In: *Information and Control* 71.3 (1986), pp. 181–185.

[5] Ian Post and Yinyu Ye. “The Simplex Method is Strongly Polynomial for Deterministic Markov Decision Processes”. In: *Mathematics of Operations Research* 40.4 (2015), pp. 859–868.

[6] David E. Smith. “Choosing Objectives in Over-Subscription Planning”. In: *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*. Ed. by Shlomo Zilberstein, Jana Koehler, and Sven Koenig. AAAI Press, 2004, pp. 393–401.

[7] Carmel Domshlak and Vitaly Mirkis. “Deterministic Oversubscription Planning as Heuristic Search: Abstractions and Reformulations”. In: *Journal of Artificial Intelligence Research* 52 (2015), pp. 97–169.

[8] David Speck, Robert Mattmüller, and Bernhard Nebel. “Symbolic Top-k Planning”. In: *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2020)*. Ed. by Vincent Conitzer and Fei Sha. AAAI Press, 2020, pp. 9967–9974.

[9] Malte Helmert. “The Fast Downward Planning System”. In: *Journal of Artificial Intelligence Research* 26 (2006), pp. 191–246.