AI Planning and Search Research Review

by Menghe Lu

Introduction

Automated planning is an important field of AI [1], at the intersection between logic and search. A *classical planning* problem can be formally defined as an abstract information of the environment, a set of actions and a goal to reach. Solving a planning problem consists in finding a sequence of actions which, when executed in order, allow the agent to reach the goal. The search is constrained by the initial state of the environment, the fact that actions require the environment to be in a certain state to be executed, and the fact that actions themselves modify the environment. One illustrative example of that would be for an agent to move objects around: moving an object from A to B requires it to be initially at A and the effect of the action is that the object is not in A anymore but is in B.

Real life planning problems include: traffic control, elevator control, power generation or planning actions of a rover on Mars. The *International Planning Competitions* [5] provide problem specifications and data for those who wants to try and evaluate their systems on planning tasks.

Planning as an abstract problem

One noticeable aspect of a planner is that it does not address the question of *how* to perform actions (a task left to subsequent modules called controllers), but rather *what* actions to perform, and *when*. As a result of this, each planning problem can be seen as an instance of a symbolic problem formulated in logical terms (typically in first-order logic, or a restriction of it). Because the planner is abstracted away from operational matters, it can rely on generic planning algorithms to perform its task.

Defining standard planning languages has been an important concern in AI planning. ADL (Action Definition Language) or PDDL (Planning Domain Definition Language) and their variant constitute notorious attempts to standardize problem formulation in planning. Such languages are not only useful for specifying a problem's constraints, but also for communicating the semantic of a problem consistently (e.g. instead of natural language between humans), making it easier to reuse prior research and algorithms.

Algorithms for classical planning

As one could expect, a planning problem comes with the inherent complexity of the language used to formalize it. For instance, solving satisfiability in predicate calculus lies in the complexity class of NP-complete problems, which are considered hard to solve: in the worst case, naive algorithms require an exponential time to terminate (determining if polynomial algorithms exist for solving these problems is related to the famous open question P =? NP, which is out of the scope of this report).

Classical planning is thus addressed by search algorithms combined with heuristics. Essentially, these algorithms consist in exploring graphs: one can view forward (or backward) chaining as a search starting from the initial state (the goal) where neighbors in the graph are state of the environment linked by a particular action. Classical algorithms such as depth-first/breadth-first search, iterative deepening or A*-like heuristics are directly relevant, as well as randomized strategies (e.g. genetic algorithms or tabu search). However specific heuristics for planning are at

the core of research in planning: by factoring problem representations or estimating the distance to the goal, these heuristics are decisive in reducing the amount of time spent to find a solution. One can refer to the planning competition leaderboards to see how important heuristics can be (in these competitions, system are evaluated on the number of tasks they can solve given bounds on computation time and available memory).

Extensions for realistic planning

As discussed above, viewing planning has a pure logical problem has some advantages, the main one being casting the problem into a generic framework. However, in the real world things might not go as expected: what if an accident occur? What if external events influence the environment? What we do not even know precisely about the effect of some action? All these questions can be addressed by enriching the framework with realistic assumptions such as time constraints (so it become both a problem of planning and *scheduling*), probabilistic transitions (i.e. the effect of an action is not deterministic), or unobservable variables (the agent works with belief states). One interesting extension is where the planning is not dissociated from execution, meaning that the planner receives feedback and can interrupt or modify the sequence of actions when new events occur. In turn, these enriched problems require specific solutions and constitute active area of research [1,2].

Conclusion: areas of growth

Automated planning and its extensions have been thoroughly studied as domain-independent problems, giving rise to a rich literature about search algorithms and heuristics. These areas of research are still active, and one can trace the progress through competitions that are organized yearly. Noticeable areas of growth include multi-agent environments (what if other agents are planning concurrently?), reasoning about time (what if the duration of an action depends on the state at which it is executed?) and acquiring domain specific knowledge (see [2]). The last topic appears very interesting in the light of the latest progress in reinforcement learning (especially with deep learning approaches), and it seems that both planning and reinforcement learning can benefit each other (see [3]).

References

[1] S. Russel and P. Norvig, *Artificial Intelligence: A Modern Approach (3rd edition)*, 2009
[2] D. S. Nau, *Current Trends in Automated Planning*, AI Magazine Vol 8 No 4, 2007.
[3] Ioannis Partalas, Dimitris Vrakas and Ioannis Vlahavas, *Reinforcement Learning and Automated Planning: A Survey*, Artificial Intelligence for Advanced Problem Solving Techniques. IGI Global, 2008.
[4] E. Keyder and B. Bonet, *Heuristics For Planning*, ICAPS, 2009.
[5] http://www.icaps-conference.org