Developments in Planning

Linear vs Nonlinear planning

Many planning problems have multiple conjunctive goals, such as in Block World that A must be B, and B on C. Early planning algorithms attempted to solve problems with multiple goals by planning for each goal separately, then executing the plans for each goal serially. Gerald Sussman *A Computational Model of Skill Acquisition* (1973) pointed out that some problems cannot be solved this way. This became know as the Sussman Anomaly and points out that merging linear goals serially cannot solve many problems. Sussman's system, HACKER, attempted to address this using what he called debugging where the system would recognize a bug with it's final plan, and create a new constraint that would prevent the bug. This is essentially a variant of backtracking.

Sacerdoti in *The Nonlinear Nature of Plan* (1975) named systems that must use backtracking to solve the anomaly Linear Planners, and showed that it's possible to create systems that do create independent plans for each goal, but treats each plan as only a partial ordering. It then will need to merge those plans, resolving conflicts in the merge to create a final total ordering that is complete. Sacerdoti did this in his system NOAH without backtracking.

However, (and I'm unclear on this) I think the merging of these partial orderings is NP-HARD (www.cs.cmu.edu/~reids/planning/handouts/Partial_Ordering.pdf). Either way, linear-only systems are always incomplete and interleaving of plans to achieve multiple goals is required, but is complex to do.

PDDL

In order to state a planning problem, a planning language is needed. The foundation for all such languages is STRIPS introduced by Fikes, Nilsson in *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving* (1971). This paper contained an entire planner, but the name stuck to the language more than the planner. STRIPS used a closed-world assumption (all unmentioned literals are false) and did not support negative literals. All literal rules were conjunctive and exhaustive.

ADL (Action Description Language) was introduced in in the book Reasoning About Actions & Plans by Edwin Pednault in 1987. ADL is based on the STRIPS syntax but used an open world assumption and allowed existential quantifiers in goals (and thus disjunctive goals) as well as negative literals. It was shown much later (in 2011 in Nebel's *On the Compilability and Expressive Power of Propositional Planning Formalisms*) that ADL can be compiled to STRIPS, but in exponential time. That should be no surprise given that converting a disjunctive statement to conjunctive statement is also exponential.

PDDL is a comprize between the two introduced in 1998 to enable the International Planning Competition(IPC). It keeps the closed-world assumptions of STRIPS and the conjunctive form of fluents, but allows for negative literals.

The IPC still runs today, with a cadance of every two or three years. There will be a competition in 2018, which includes a modified PDDL to describe stochastic problem spaces (http://www.icaps-conference.org/index.php/Main/Competitions)

More Recent Work

All of the above achievements are over two decades old. I wanted to find some more recent advancements, but it's hard to know what is a significant contribution close to the fact. Additionally the field has appeared to stagnate. For this third, I'll focus on the paper that won Best Paper in ICAPS 2017. Trevizan et al in *Occupation Measure Heuristics for Probabilistic Planning* introduces a technique that address planning on stochastic shortest path problems, a much more complex and interesting space than deterministic problems we've been looking at.

The paper introduces the first heuristic for stochastic problems that is both admissible, and uses the probabilities of action results rather than falling back to deterministic techniques. This heuristic is based off of the deterministic operator counting heuristic that estimates the number of times actions will be used to achieve a goal using an a linear program (LP). This heuristic runs a stochastic block-world (where blocks have a .25% of falling) with up to 10 blocks, which only one prior deterministic heuristic could do. That still seems pretty disappoint to me though.