Planning Research Review

The sub-field of planning within AI evolved from the pragmatic requirements of robotics and operations scheduling, and it has undergone several significant shifts over the years. [1]

Partial order planning

Prior to the 1970s, planning was done primarily through total order planning. A planning system would figure out action sequences that would produce each subgoal, and then would string them together to produce a full solution. The problem is there were some types of simple plans (like the Sussman anomoly) that could not be handled by this kind of total ordering. [2]

This prompted the creation of planning systems that could build "task networks", plans where the real contribution is the idea of proposing necessary tasks in ways the provide the *least* possible ordering constraints, thus an output plan describes only strict orderings where one task is dependent on the prior completion of another, they "partially ordered" plans. [3]

State Space Planning

Partial order planning still has the problem of dealing with actions that "undo" pieces of the successful plan already completed. It can be complicated to do sufficient back-chaining for complicated states to find a reasonable path to the goal. In 1996, the UNPOP paper showed that finding a decent heuristic and applying normal search processes with it was an improvement on attempting to chain order-independent sub-plans.

The proposed heuristic used the min count of necessary actions to achieve the goal from the current state, ignoring any delete effects they use, and then using conventional search patterns to navigate to a solution. [4]

Graphplan

A new perspective on planning was advanced in 1995-1997 by the emergence of GRAPHPLAN. This is where the encoded representation of a "planning graph" comes from, and marked a big step forward in terms of efficiency for generating a reasonable plan for complex domains.

A planning graph, rather than representing the possible state space, uses a compact representation of all the possible states after each number of possible actions (each action level has all possible actions at that point, each state level has all possible resulting states from the prior actions, incompatibilities are stored via mutex lists).

The planning graph itself provides a useful heuristic for estimating the distance from solution for a given state, but it also is possible to extract a working plan from the encoding using the GRAPHPLAN algorithm. The intuition is that the planning graph is extended to the first level at which the goal fluents all exists without mutex, and then backward-chaining is initiated from the last state level, searching for actions that *add* the most goal fluents in the previous action level (thus resulting in a new state, at the prior level, and a search for actions in the prior action level that achieves the most of the *remaining* unaccounted for goal fluents).

This was a major step for the field at the time because the algorithm was empirically demonstrated to outperform total planning, partial planning, and 'Prodigy' on state-of-the-art benchmarks. [5]

[1] Peter Norvig, Stuart Russel. Artificial Intelligence, A Modern Approach. Pearson, 2015.

[2] Sussman, G. J. A Computer Model of Skill Acquisition. Elsevier/North-Holland, 1975.

[3] Sacerdoti, Earl D. A Structure for Plans and Behavior. Elsevier/North-Holland, 1977.

[4] McDermott, Drew A Heuristic Estimator for means-ends analysis in planning Third Int. Conf. on AI Plannng Systems, 1996

[5] Avrim Blum, Merrick Furst Fast Planning Through Planning Graph Analysis Artificial Intelligence, 90:281–300, 1997