Research Review - AI Planning and Search

By Vijai Narayanan

Fikes and Nilsson's STRIPS[1] represented a significant advancement in the representation of planning domains in computer science. While simple AI search problems could be represented using traditional data structures such as lists and matrices (like our Sudoku problem in the previous module), real-world problems encountered by Stanford Research Institute's (SRI) path finding robot required a more flexible and intuitive system. This language system was useful in identifying states and searching through space for a wide range of applications in AI planning.

STRIPS defines a world model using what it calls well-formed functions (WFFs) for an initial world, a set of operators (also called an action schemata), and a goal condition. Problem search becomes more efficient when the world is defined using STRIPS. Rather than search every possible eventuality in the world state (as we did in the Sudoku project), STRIPS narrows the search focus by comparing the initial world to the goal condition using feasible operations in the action schemata. Intermediate goals could also be generated as needed using action preconditions to arrive at the goal state. In doing so, STRIPS employed some of the best elements of both progression and regression search.

When solving the Sudoku problem, a (deep) copy of the world was generated for each successive state and the values of each square on the board was recorded. In STRIPS implementation, only relevant states affected by actions are recorded. This allows for memory usage to be kept at a minimum (quite important for real world problems where the number of states can be large). On an unrelated note, it was quite interesting to read the authors' ambitious definition of intelligence -- "an intelligent system ought to solve more difficult problems than any it has solved before."

The concepts introduced in STRIPS were further extended by Universal Method Composition Planner (UMCP), and action definition language (ADL) to allow for even more flexibility. The Planning Domain Definition Language (PDDL)[2] was aimed at standardizing some of these concepts for the AIPS-98 planning competition. This paper exhaustively documents the notation for goals, actions, domains, etc. and had a larger impact on the field than just the competition.

The key concept introduced in PDDL was the separation of the domain (world model) from problems. Using Extended Backus-Naur notation, the domain can be developed to express the physics of that particular world. This domain can then be extended for specific problems that needed to be solved. As an example, PDDL could state that a room and all the items in it should be painted red without explicit definition of the items in the room. Using STRIPS would require explicit definition of all the items in the room.

Whereas STRIPS and PDL were advances in planning language definition, Graphplan[3] represented a way to take a planning language (STRIPS) and make search more efficient. This is done by organizing planning problems in a graph with levels alternating between proposition levels and action levels. The edges of the graph, in turn, are precondition edges, add edges, and delete edges.

Another integral part of the planning graph is the inclusion of mutual exclusion relations. While this results in significant time spent on graph creation, it means that Graphplan will always return the shortest possible plan (when one exists). The most significant limitation of Graphplan is that it does not allow for PDDL problems.

Citations

[1] Fikes, Richard E., and Nils J. Nilsson. "STRIPS: A new approach to the application of theorem proving to problem solving." *Artificial intelligence* 2.3-4 (1971): 189-208.

[2] Aeronautiques, Constructions, et al. "PDDL| The Planning Domain Definition Language." (1998).

[3] Blum, Avrim L., and Merrick L. Furst. "Fast planning through planning graph analysis." *Artificial intelligence* 90.1 (1997): 281-300.