Udacity Artificial Intelligence Nanodegree
Development Of AI – Research Review
Submission By: Lim Si Jie

1. STRIPS

Stanford Research Institute Problem Solver (STRIPS) is an automated planner that was developed and designed by Richard Fikes and Nils Nilsson in 1971 at SRI International. STRIPS was designed as a problem solver that attempts to find a sequence of operators in a space of world models to transform a given initial world model where a given goal formula can be proven to be true (Fikes & Nilsson, 1971). By using means-ends analysis and a resolution theorem prover to answer questions of particular models, STRIPS can turn into a desired goal-satisfying model.

As a classical planning language, STRIPS is composed of states, goals and set of actions (Sethi, 2017). State is a conjunction of positive literals, which cannot contain variables and invoke functions. Similar to the state, Goal is conjunction of positive and ground (no variables and no functions) literals. Actions, or operators, include pre-conditions and post-conditions, which are both represented as a conjunction of function-free literals. Pre-conditions describe the state of world required to perform action, while post-conditions describe state of the world after action is executed (Shrott R. J., 2017).

The representation language used by STRIPS has been far more influential on the impact of artificial intelligence than its algorithmic approach. The current "classical" language that is employed today is close to what STRIPS used (Norvig & Russell, 2018).

2. Planning Graphs

Avrium Blum and Merrick Furst at Carnegie Mellon developed a new approach to planning in STRIPS-like domains in 1997. This approach was based on constructing and analyzing a compact structure, which is termed as Planning Graph (Blum & Furst, 1997). Planning Graph built on the initial work done by Richard Fikes and Nils Nilsson to bring significant practical value in solving planning problems. If a valid plan exists in the STRIPS formulation, the plan must also exist as a subgraph of the Planning Graph (Shrott R. , 2017).

In the GraphPlan algorithm, greedy searching is not used. Instead, a Planning Graph object is first created. The Planning Graph inherently encodes useful constraints explicitly, thereby reducing the search overhead in the future. Planning Graphs have similar features to dynamic programming problem solvers. They are not only based on domain information. They also take the goals and initial conditions of the problem and makes an explicit consideration on the time taken to reach the goal. For any plan that the algorithm finds, it is bound to be a legal plan and it will always find a plan if one exists. The GraphPlan algorithm also has a termination guarantee that is not provided by most planners.

In essence, the GraphPlan algorithm uses a planning graph to guide its search for a plan and guarantees that the shortest plan will be found.

According to Blum and Furst, searches made by the Planning Graphs approach are fundamentally different from the searches of other common planning methods. Thus, they provide a new perspective on the planning problem that is also proven to be sensible by empirical evidence (Blum & Furst, 1997).

3. Heuristic Search Planning

Heuristic Search Planning (HSP) furthers the research of Planning Graphs by Avrium Blum and Merrick Furst. It uses the ideas of heuristic function to search from an initial state to a goal state to provide an estimate of the distance to the goal. The HSP algorithm transforms planning problems into a heuristic search by automatically extracting heuristics from the STRIPS encodings (Bonet & Geffner, 2001). In both phases of HSP (forward propagation and regression search), ideas are borrowed from GraphPlan. The biggest difference between HSP and GraphPlan is the implementation of the search that takes advantage of the plan graph with good efficiency.

HSP showed that simple state space search algorithms guided by a general domain-independent heuristic produce a family of planners that are competitive with some of the best planners at that point in time. Interestingly, the planner that showed the best performance turned out to be the simplest planner (HSP2) which uses a best-first forward search with re-computation at every state (Bonet & Geffner, 2001).

Works Cited

Blum, A. L., & Furst, M. L. (1997). Fast Planning Through Planning Graph Analysis. *Artificial Intelligence*, 281–300.

Bonet, B., & Geffner, H. (2001). Planning As Heuristic Search. *Artificial Intelligence*, 5-33.

Fikes, R. E., & Nilsson, N. J. (1971). Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 189-208.

Norvig, P., & Russell, S. (2018). *Artificial Intelligence: A Modern Approach (3rd Edition).* Prentice Hall.

Sethi, H. (2017, March 30). *Evolution of Planning in AI*. Retrieved from LinkedIn.com: https://www.linkedin.com/pulse/evolution-planning-ai-harpreet-sethi/

Shrott, R. (2017, Septemeber 4). *AI Planning Historical Developments*. Retrieved from TowardsDataScience.com: https://towardsdatascience.com/ai-planning-historical-developments-edcd9f24c991

Shrott, R. J. (2017, September 06). *AI Planning Historical Developments*. Retrieved from LinkedIn.com: https://www.linkedin.com/pulse/ai-planning-historical-developments-ryan-j-shrott/