Research review of AI planning and search

1. Introduction

Artificial Intelligence (AI) planning arose from investigations into state-space search, theorem proving, and control theory and from the practical needs of robotics, scheduling, and other domains. [1]. In this short review I will discuss a few major historical developments of intelligent planning in the field of AI.

2. G.P.S.

The first planner I will talk about is called General Problem Solver (G.P.S.). It was a computer program written in IPL programming language, created in 1959 by Herbert A. Simon, J.C. Shaw, and Allen Newell [2]. It was intended to work as a universal problem solver machine, By separating the knowledge of the problem from the strategy it could in principle solve any problem as long as it was defined using logical formulas as a directed graph from initial state to goal. The user defined objects and operations that could be taken on the objects. GPS would then generate heuristics by means-ends analysis and by breaking down the goal into subgoals it would get closer to the goal. GPS was a state space search program which worked well on well defined problems, but struggled with real life problems where the problem state space explodes computationally. GPS eventually evolved into Soar architecture for artificial intelligence.[3]

3. STRIPS

STRIPS (Stanford Research Institute Problem Solver) is a planner designed by Richard Fikes and Nils Nilsson in 1971 at SRI (Stanford Research Institute) International as the planning component of the software for the Shakey robot project[1]. STRIPS was modelled on the GPS planner described earlier, but could solve more complex and more general problems than GPS. STRIPS attempts to find a plan as "*a sequence of operators in a space of world models*" [4] that can be executed from the initial state and that leads to a goal state. STRIPS operators transform one state into another on a path to a goal state. The formal representation language introduced by STRIPS contributed to the area of intelligent planning way more than the algorithm itself. The language has become a major inspiration for what we now refer to as "classical" representation planning languages.

4. PDDL

Planning Domain Definition Language (PDDL) is an attempt to standardize Artificial Intelligence (AI) planning languages which have proliferated over the decades. PDDL was developed by Drew McDermott and his colleagues in 1998 [5], primarily inspired by STRIPS and ADL (Action Description Language). PDDL is slightly less restricted than STRIPS whose preconditions and goals cannot contain negative literals [1]. This allows to model more realistic world problems more easily. Besides the huge value which the standardisation brings, the big power of PDDL is to represent large number of actions that can be taken on variables succinctly in a simple action schema that clearly defines the conditions for when can the particular action be taken.

References:

[1] Russell, S. J., Norvig, P. (2010), Artificial intelligence: A modern approach
[2] Newell, A.; Shaw, J.C.; Simon, H.A. (1959). Report on a general problem-solving program. Proceedings of the International Conference on Information Processing
[3] https://en.wikipedia.org/wiki/Soar_(cognitive_architecture)
[4] Richard E. Fikes, Nils J. Nilsson. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving", (Winter 1971)
[5] https://en.wikipedia.org/wiki/Planning_Domain_Definition_Language