Research review of historical developments in the field of AI planning and search

for AIND planning project by A. Tkachenko

In this paper we will provide a brief overview of domain and problem description languages used in the classical planning problem solving. That is, languages used for planning environments that are fully observable, deterministic, static, finite and discrete [1].

STRIPS is one of such languages - developed in early seventies but still widely in use today [2, 3]. The name (Stanford Research Institute Problem Solver) originally referred to the planner used by the authors to solve a robot control problem, but now it is commonly used for the language they developed to provide inputs to that planner. STRIPS provides a framework for approaching planning problems by representing the world using a set of well-formed formulas (WFFs) – an approach inspired by an even earlier General Problem Solver program [4]. STRIPS can be used to define the goal states; allowable actions: each having its own preconditions and effects (postconditions); and the initial state.

Limitations of STRIPS, shown in [1] (p. 379, Fig. 11.1), have led to the development of other languages taking inspiration from it, such as Action Description Language (ADL) [5]. ADL added more expressive capabilities that were not supported in STRIPS, such as negative literals in states, open world assumption (unmentioned literals are unknown), conditional effects, and support for equality and variable types.

After the development of STRIPS and ADL other language variants, such as UMCP for hierarchical task-network planning [6] have followed. Eventually a standardized syntax was developed in order to improve interoperability between various research groups – a Planning Domain Definition Language (PDDL) [7]. PDDL is not a monolithic all-or-nothing approach – a planner can support a certain subset of PDDL while not supporting others. This allows a certain degree of flexibility, but also resulted in a situation where many planners support only commonly used STRIPS subset, but not the others [8]. Development of PDDL is an ongoing effort, with newer version introducing additional features such as numeric (non-boolean) fluents and more.

References

[1]. P. Norvig and S. Russel, Artificial Intelligence: A modern approach (3rd edition), Prentice Hall, 2009
[2]. R. E. Fikes, and N. J. Nilsson, “STRIPS: A new approach to the application of theorem proving to problem solving”, AIJ, 2 (2-3), p. 189-208, 1971.
[3]. R. E. Fikes and N. J. Nilsson, “STRIPS, a retrospective”, Artificial Intelligence, 59, p.227-232.
[4]. A. Newell, J. C. Shaw, H. A. Simon, “Report on a general problem-solving program”, 1958
[5]. E. P. D. Pednault, “Formulating multiagent, dynamic-world problems in the classical planning framework”, Reasoning about Actions and Plans: Proc. 1986 Workshop, p. 47-82, 1987
[6]. K. Erol, J. Hendler, and D. Nau, UMCP: A sound and complete procedure for hierarchical task-network planning. Proc. 2nd Intl. Conf. on AI Planning Systems, p. 249-254, 1994
[7]. M. Ghallab et al., “PDDL – The Planning Domain Definition Language. Version 1.2”, AIPS-98 Planning Competition Committee, 1998.
[8] Web, “Writing Planning Domains and Problems in PDDL”, URL: https://www.ida.liu.se/~TDDC17/info/labs/planning/2004/writing.html, Last accessed: 2017-06-18.