Planning search: Prodigy and Graphplan

1. Introduction

We study some major evolution of planning first via two of their implementations, we will discuss Prodigy architecture, described in [1] and GraphPlan from [2].
Prodigy and Graphan plan both returns plans (sequences of operators) in order to achieve a goal given controls. The main issue in a graph search is to know which node to expend next. These 2 mainly apply to STRIP like domains just regular planning domain, explained in [3] although the system of Prodigy is far more general than the Graphplan.

2. Prodigy

Prodigy is a general-purpose problem solver that is characterized by the use of central general solver for one of the first time. However most notable improvement is the use of differentiate learning where the unlike most system created before, the learning process is also selective, which means that the system does not only learn. Secondly, the system makes use of multiple learning strategies only resorting to depth search

3. Graphplan

First is worth mentioning that the use of the GraphPlan is more restricted than the Prodigy system in the sense that it is restricted to planning graphs (eg no state space problem). From a high level view, at each iteration, the algorithm extends the graph one time step (the next action level and the following proposition level), and then searches
the extended Planning Graph for a valid plan. Then it just returns the solution or the absence of solution.
To extend the graph, for each operator and each precondition, the algorithm creates some mapping of possible precondition and operators. For instance, if a cargo was unload at location A with plane X then it cannot be unload at location A by plane Y at the next iteration. The time spent on creating such mapping is compensated by a higher speed of the search following this extension. Then the search of a plan goal is to make use of precondition level by level so that preconditions of a level i-1 make the sub goal of the iteration i. Finally, the algorithm makes use of the technique of learning in case a set of action has been proven unsolvable (just adds the case to a table).

4. Discussion

Obviously, the fact that Graphplan is more specialize and more modern makes it more efficient and faster than the Prodigy system. On most problems in [2], the Graphplan outperforms the Prodigy system.

Following the features that were discussed, the features that appeared to be relevant are in first place the way the algorithm captures the mutual exclusion as explained above, its ability to learn the “dead end” in a search and avoid them.

Bibliography

[1] O. E. ,. A. G. ,. R. J. ,. C. K. ,. S. M. ,. M. V. Jaime Carbonell, "PRODIGY: An Integrated Architecture for Planning and Learning," *SIGART Bulletin,* 1990.

[2] M. L. F. Avrim L. Blum, "Fast Planning Through Planning Graph Analysis," *Artificial Intelligence,* 1997.

[3] N. J. N. Richard E.Fikes, "STRIPS: A New Approach to the Application of Theorem proving to Problem Solving," *Artificial Intelligence,* 1971.