# Research Review – History of AI Planning

AI Planning has long been an interest in the field of Computer Science. In order to get a better understanding of where the field is at the moment, it is crucial to understand the important achievements that brought us to today. As part of this review, three of such achievements will be summarised; the STRIPS language, the GRAPHPLAN algorithm and the BLACKBOX algorithm.

When talking about automatically solving problems, the first step is generally to formulise the problem, such that the constraints and the components of the problem can be easily understood by a computer program. One of the most notable developments in representing planning problems was the introduction of the **STRIPS** language (Fikes and Nilsson, 1971). Although the contribution of STRIPS was not limited to the language itself, the language proved to be a far more important milestone in AI Planning than the actual planner introduced with STRIPS. STRIPS simply allowed decomposing wide set of problems into propositional expressions that were easy to interpret by planning algorithms.

STRIPS language achieved this by representing planning problems as a combination of initial and goal states that were defined by conjunctions of propositional literals, and an action schema that defined how one state transitioned into another. A sample STRIPS specification for the Dinner-Date problem (Weld, 1999) is shown in Figure 1.

|  |
| --- |
| Initial Conditions: (and (garbage) (cleanHands) (quiet))  Goal: (and (dinner) (present) (not (garbage)))  Actions:  cook :precondition (cleanHands)  :effect (dinner)  wrap :precondition (quiet)  :effect (present))  carry :precondition  :effect (and (not (garbage)) (not (cleanHands)))  dolly :precondition  :effect (and (not (garbage)) (not (quiet))) |

Figure - STRIPS Specification of the Dinner-Date Problem (Weld, 1999)

The STRIPS language and its variants such as ADL (Pednault, 1989) and PDDL (Ghallab et al., 1998) helped formulising planning problems and creating efficient planners that could execute on these statements. One such planner that was orders of magnitude faster than its predecessors was the **GRAPHPLAN** algorithm (Blum and Furst, 1995, 1997). The algorithm leveraged planning graphs and used binary mutual exclusion relations (mutex) between nodes at the same level to reduce the search space making it more efficient than any other algorithm at the time. A good and precise description of how the algorithm works is given by Weld (1999) as below.

*“GRAPHPLAN alternates between two phases: graph expansion and solution extraction. The graph-expansion phase extends a planning graph forward in “time” until it has achieved a necessary (but possibly insufficient) condition for plan existence. The solution-extraction phase then performs a backward-chaining search on the graph, looking for a plan that solves the problem; if no solution is found, the cycle repeats by further expanding the planning graph.” (Weld, 1999)*

GRAPHPLAN was the basis for various planning algorithms to come. The **BLACKBOX** planner successfully combined speedy search offered by GRAPHPLAN with the ability to compile planning problems into propositional formulas (Kautz and Selman, 1998). This allowed using stochastic methods to extract solutions by performing a greedy search on different levels as opposed to visiting each level in order like GRAPHPLAN did. Figure 2 shows high level architecture of BLACKBOX (Weld, 1999).

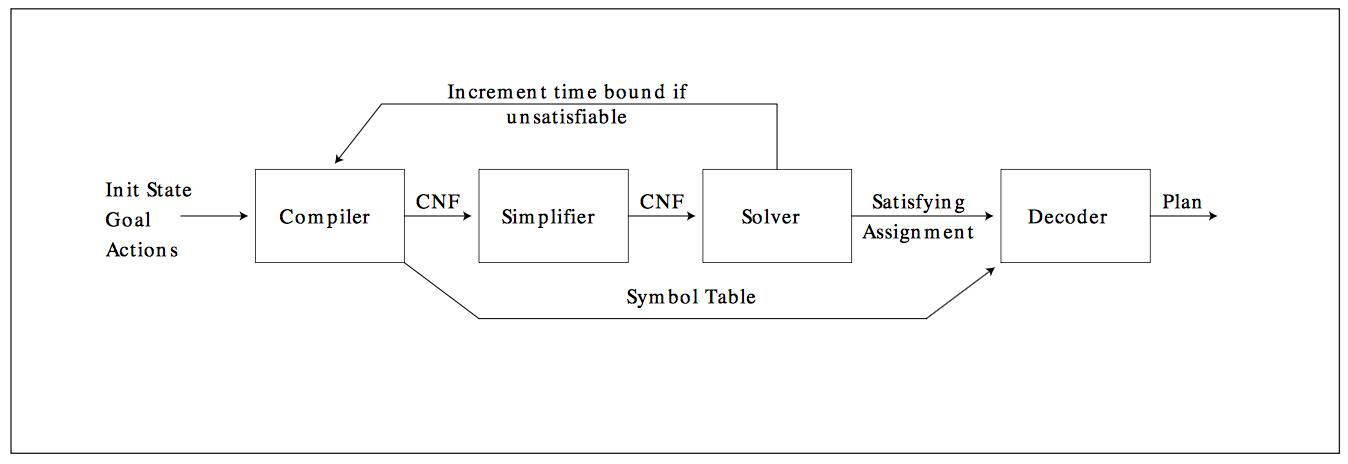


Figure - Architecture of BLACKBOX (Weld, 1999)

As seen in Figure 2, the planning graph is converted into a CNF expression by a compiler and a SAT solver is used to extract a plan which can then be decoded. This method is faster than either of the two algorithms it combines; GRAPHPLAN and SATPLAN. This was because GRAPHPLAN suffers from using a strict backtracking search which is replaced with WALKSAT in BLACKBOX, allowing the search to be greedy and hence faster. SATPLAN also suffered from unnecessarily evaluating a large set of states and actions which are eliminated as being impossible by BLACKBOX, thanks to the mutexes used in the underlying planning graph.

### References

Fikes, R., and Nilsson, N. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. Journal of Artificial Intelligence 2(3–4): 189–208.

Weld, D. S., 1999. Recent Advances in AI Planning. AI Magazine Volume 20 Number 2: 93-123.

Pednault, E. 1989. ADL: Exploring the Middle Ground between STRIPS and the Situation Calculus. In Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning, 324–332. San Francisco, Calif.: Morgan Kaufmann.

Ghallab, M.; Howe, A.; Knoblock, C.; McDermott, D.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. The Planning Domain Definition Language. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, Yale University.

Blum, A., and Furst, M. 1997. Fast Planning through Planning Graph Analysis. Journal of Artificial Intelligence 90(1–2): 281–300.

Blum, A., and Furst, M. 1995. Fast Planning through Planning Graph Analysis. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, 1636–1642. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Kautz, H., and Selman, B. 1998. BLACKBOX: A New Approach to the Application of Theorem Proving to Problem Solving. In AIPS98 Workshop on Planning as Combinatorial Search, 58–60. Pittsburgh, Penn.: Carnegie Mellon University.