Research Review

December 5th, 2017

Jesús Martínez

STRIPS

In the year 1971, at the Stanford Research Institute, Richard Fikes and Nils Nilsson developed STRIPS, the planning component of Shakey, the first machine capable of reasoning about its own actions and considered by many people the precursor of self-driving cars, drones and other robots and intelligent agents. What's really interesting about STRIPS is the representational language used by it, which far exceeded the scope of the Shaky project and became a ground-breaking advancement in the field of AI.

STRIPS as a planning language is comprised of several key components: **State**, which is a conjunction of positive literals that cannot contain functions; **Goal**, which for all purposes is a State (again, only positive literals, no functions) and, finally **Actions**, which define transitions between States via preconditions and postconditions (effects).

In STRIPS, actions' effects are added to the world if and only if they're positive. If they're negative, they're removed.

It is worth mentioning that the real impact of STRIP is its representational power rather than its algorithmic approach.

ADL and PDDL

Although the appearance of STRIP was a turning point in the history of Artificial Intelligence and, in particular, in the field of planning, it has noticeable limitations that were relaxed by ADL, which stands for Action Description Language (Pednault, 1986). ADL doesn't assume that not mentioned literals are false (as happens in STRIP), but unknown. It also allows negative literals, disjunctions and quantified variables in goals, unlike STRIP.

The Problem Domain Description Language (Ghallab et al, 1998) was born as a mean to standardize the ecosystem of planning languages (such as STRIPS, ADL, among others) and has been used as the standard language for the planning competitions at the AIPS conference, beginning in 1998.

GRAPHPLAN

GRAPHLAN surged up in the mid nineties by the hands of Avrim Blum and Merrick Furst (1995, 1997) and it revitalized the field of planning, mainly due to the fact of being orders of magnitude faster than the partial-order planning algorithms of the time.

GRAPHPLAN works by, first, checking if all the goal literals are present in the current level of the graph and no pair of them are connected by mutual exclusion links. Then, if that's the case a solution might exist within the current graph. If not, then the graph is expanded by adding the actions of the current level and the state literals for the next level. Eventually, either a solution is found or the algorithm determines that the planning problem is unsolvable.