Research Review

GRAPHPLAN. Whereas the AIND-Planning project constructs and analyzes a planning graph for the level-sum heuristic, GRAPHPLAN constructs a planning graph and then performs a search over it in order to directly solve a planning problem. The GRAPHPLAN algorithm is iterative, with two phases in each iteration: *graph expansion* and *solution extraction* (Daniel S. Weld, *Recent Advances in AI Planning*, 1998). In graph expansion, the algorithm expands the graph with one additional action and state level; then, if all goal literals are present without mutex, a solution-extraction search is conducted from the last state level to the root in order to find a satisfying plan. If some leveling-off conditions are met and no plan found, then a failure is returned.

Weld reports that GRAPHPLAN in many cases is able to solve problems "orders of magnitude faster" than previous state-of-the-art systems (Daniel S. Weld, *Recent Advances in AI Planning*, 1998). Furthermore, according to Weld, "The representations used by Graphplan form the basis of the most successful encodings of planning problems into propositional SAT," informing the next development we look at.

Compilation of planning to SAT. Applying a propositional satisfiability (SAT) solver to a planning problem requires converting the problem representation to conjunctive normal form (CNF) and then iteratively choosing larger and large candidate lengths for the solution, applying SAT techniques to find a plan of the current candidate length that satisfies all conditions. One such solver, the BLACKBOX system, was one of the world's faster solvers at the time of Weld's *Recent Advances in AI Planning*. The planning graph used by GRAPHPLAN can be automatically converted into the notation used by SAT solvers, and BLACKBOX does precisely this (Daniel S. Weld, *Recent Advances in AI Planning*, 1998).

REPOP. Reviving Partial Order Planning (REPOP) shows that the techniques that led to contemporary advances in planning technologies, some of which are used in GRAPHPLAN, can be adapted to partial-order-planning (POP) algorithms in order to boost their efficiency (XuanLong Nguyen and Subbarao Kambhampati, *Reviving Partial Order Planning*, 2001). Nguyen and Kambhampati show that REPOP outperforms GRAPHPLAN in terms of both efficiency and desirability of solutions (i.e., solution length and flexibility) in several parallel domains, where a parallel domain is one that allows parallel execution of some actions, or execution of some actions while other parts of the plan are blocked or paused (XuanLong Nguyen and Subbarao Kambhampati, *Reviving Partial Order Planning*, 2001).

An important contribution of REPOP is in making POP competitive with other solvers in terms of efficiency. At the time of REPOP's development, POP remained attractive, despite its relative inefficiency, due to the minimal "commitment" inherent in the plans it returns: POP tries to impose the least amount of ordering constraints on the plan as possible, allowing for flexible real-world execution of the plan (XuanLong Nguyen and Subbarao Kambhampati, *Reviving Partial Order Planning*, 2001).