Planning and Logic Research

One of the most influential additions to AI was logic and planning. Arguably, one of the biggest contributions to this influence was the invention and popularization of STRIPS, invented by Richard Fikes and Nils Nilsson in 1971. Instead of simply defining a state, the user provides preconditions, actions and the results of those actions. This allows for heuristics to be more easily derived from all of this extra information, often automatically. With the use of STRIPS came the most common planning language, PPDL. The main significance here was its simplicity, since it allowed the user to write their plans in close to plain english. Not only did PPDL and STRIPS provide one universal technique for defining plans, but it was a key piece in the next huge leap in planning logic, the GraphPlan.

Invented by Blum and Furst in 1997, planning graphs supplied a way to easily derive constraints explicitly without the need to provide a full state tree, which is exponentially more expensive to traverse. Built on top of STRIPS, planning graphs take actions into account, as well as objects and goal propositions. GraphPlan also allowed both total order and partial order plans to be addressed, meaning time in space could be a factor, or the order of actions could be ignored, depending on the problem. While planning graphs were a huge leap forward in the planning space, they still had limitations like being forced to find the shortest plan, or the inability to solve problems with continuous environments. This led to a very important follow up paper four years later by Nguyen, Kambhampati & Nigenda.

This paper, **Planning graph as the basis for deriving heuristics for plan synthesis by state space and CSP search (2001)**, suggests that planning graphs can be combined with CSP search to reap the benefits of both planning techniques without the hinderance of each technique's disadvantages. On one side, planning graphs guarantee to find the optimal solution, but they require a full traversal of the search space, up until finding the solution. On the other, many state space searches like HSP tend to make the assumption that there is independence between subgoals, or more simply, they do not account for inter-subgoal interactions. This causes the heuristic to be inadmissible, and no longer guarantee an optimal solution.

Up until this point, one of the biggest issues in AI search was deriving an admissible heuristic from a state problem. To address this, a multitude of heuristics can be utilized. The first one is the sum heuristic, which takes the sum of each individual proposition. Since this grossly overestimates, an admissible approach is the max heuristic. Once again this is not an ideal approach, since taking a max instead of summing, greatly underestimates the cost. The next idea, sum mutex heuristic, helped address the issue of mutexes not being accounted for, but eventually falls short, since it cannot always identify an inconsistent state. This causes the rest of the plan to fly dramatically off course, stopping the heuristic from being very accurate.

In order to test their planning techniques against other popular ones, Nguyen et al. wrote AltAlt, which is a planner built on top of STAN (used for planning graphs) and HSP-r (a heuristic search planner). Not only was AltAlt as good as or better than its competition in almost every problem, but was able to greatly reduce the cost of heuristic computation without really damaging the quality of the solution. The final application of the planning graph was in CSP searches. In this case, the constrained value is the one with the fewest actions supporting it. Using a planning graph to derive heuristics for ordering these values, they then performed a

backwards PlanGraph search, ordering the most constrained values first. This gave the search a speed increase of up to 400%.

I think the results of Nguyen et al. were clearly very important findings in the field of state space search AI. Not only do AltAlt's results show that this technique performs magnitudes better than other planning techniques, but it does so in a way that is both optimal and admissible. The combination of multiple heuristics and planning graphs with state space search also provided a planning solution that works on a broad variety of different planning domains, while outperforming the previous competition for each one. It's also a good lesson for future AI developers, sometimes the greatest advancements in AI are only their most successful when skillfully combined with other methods.