This document is a summary of three points on one branch in the evolution of the field of planning problems. We briefly describe some of the terms and paradigms used in the field, and then briefly discuss some systems on the road to efficient and competitive partial order planners, with reference to (but without going into any detail) competing approaches and systems.

The classical planning problem - wherein the world is regarded as being static, in a given state, and an agent has the task of arranging the world in a different, desired, state, was most famously investigated at the Stanford Research Institute (SRI) in the late 60s and early 70s.

That research, using Shakey the Robot to move boxes around rooms, lead to the development of the STRIPs (STanford Research Institute Problem Solver) automatic plan generator, which was a combination of a state-space heuristic search and a theorem resolver. Notable aspects of STRIPs were the famous A* heuristic search algorithm, which arose out of the project, as well as the STRIPs representation language. [4] This made the novel (for the time) closed-world assumption: a plan only modifies those parts of the world explicitly mentioned in its deletion and action operators; all conditions not mentioned are assumed false. It was also assumed that only one action could occur at a time, actions took effect immediately – and, as to be expected from the static world assumption, Shakey was the only agent making changes to the world.

Arguably the most durable aspects to arise from STRIPs was its formalism of the classical planning problem. Restrictions in the original STRIPs language have been relaxed, leading first to the Action Description Language ADL [8] and thence to Problem Domain Description Language PDDL [5], which today is the standard language for the International Planning Competition [10]. On a theoretical level, although the general planning problem lies in PSPACE (> NP) the formalism allows the development of very accurate, and more tractable, domain-independent heuristics to guide solvers towards a solution [5, 10].

The desired outcome of a planning algorithm is, not surprisingly, a plan, or sequence of actions, to be taken from an initial state to achieve the desired goal. STRIPS and its descendants search through the space of all possible plans, starting with the plan consisting of the just the initial and goal states. In contrast to STRIPS, which used backwards chaining to explore plan states, many of STRIPS descendants used forward chaining to try and find a solution plan.

Solution plans can either be totally ordered (each action has one precedent and antecedent action, and the order of all the actions is explicitly defined) or partially ordered (POP), where actions, or several actions sequences, can be specified in parallel with one another. [6] Generally, partial order planners are seen to be superior to totally ordered [1], intuitively because they're more flexible, as it's easier to work on solving smaller sub-problems than finding one precise solution to a large problem.

One system which encapsulate the ideas of forward chaining and partially ordered plans is UCPOP [9] and, like similar systems, fell out of favour in the late 1990s [10] for being slower than planners based on constraints (CSP, or Constraint Satisfaction Problem planners, epitomised by GraphPlan [2, 3], for two main reasons – forward chaining problems frequently expand rapidly into large state spaces, and forward chaining planners often explore irrelevant actions [10]. As can be imagined, a tendency to explore the irrelevant states of a large state space leads to a slow and inefficient planner!

The pinnacle of Partially Ordered Planning, and hence the end of our discussion, was epitomised by RePOP [7] which improved the efficiency of UCPOP by incorporating several of the successful techniques used by the CSP planners. RePOP is as fast as competing CSP planners on problems of the same size, leading to shorter plans with better execution flexibility [11].

There are many approaches to solving planning problems. The CSP approach was briefly touched on here, but others, such as Heuristic Search Planners (HSP) and planners based on Binary Decision Diagrams (BDD) weren't mentioned at all. The main takeaway from this summary is that, after their heydey in the 90s, and after being eclipsed by other approaches, Partial Order Planners (POP) are once again a competitive and attractive choice for such problems.

Rerefences:

[1] Barrett, A. and Weld, D. (1992). Partial Order Planning: Evaluating Possible Efficiency Gains. Technical Report 92-05-01, University of Washington, Department of Computer Science and Engineering

[2] Blum, A. L. and Furst, M. (1995). Fast planning through planning graph analysis. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp.1636–1642.

[3] Blum, A. L. and Furst, M. (1997). Fast planning through planning graph analysis. *Artificial Intelligence Journal*, 90(1–2), 281–300.

[4] Fikes, R. E. and Nilsson, N. J. (1993). STRIPS, a retrospective. *Artificial Intelligence Journal*, 59(1–2), 227–232.

[5] Ghallab, M., Howe, A., Knoblock, C. A., and McDermott, D. (1998). PDDL—The planning domain definition language. Tech. rep. DCS TR-1165, Yale Center for Computational Vision and Control.

[6] Kalisch, M., & König, S. (2005). *Introduction to AI Planning*. Retrieved June 22, 2017, from http://www.cogsys.wiai.uni-bamberg.de/teaching/ws0405/s_planning/slides/Introduction_AI_Planning_Folien.pdf

[7] Nguyen, X. and Kambhampati, S. (2001). Reviving partial order planning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 459–466.

[8] Pednault, E. P. D. (1986). Formulating multiagent, dynamic-world problems in the classical planning framework. In *Reasoning about Actions and Plans: Proc. 1986 Workshop*, pp. 47–82.

[9] Penberthy, J. S. and Weld, D. S. (1992). UCPOP: A sound, complete, partial order planner for ADL. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pp. 103–114.

[10] Russell, S. and Norvig, P. (2009) *Artificial Intelligence: A Modern Approach* (3rd ed.). Prentice Hall Press, Upper Saddle River, NJ, USA.

[11] Smith, D., Frank, J. and Jonsson, A. Bridging the gap between planning and scheduling. In Knowledge Engineering Review, 15(1):47-83. 2000.