Modern AI Planning: Reading List

P@trik Haslum

1. Introduction

Every textbook on AI has a chapter on planning, which begins, and often ends, with STRIPS (Fikes & Nilsson 1971) (more recent ones also mention TWEAK (Chapman 1987)). There's more to it than that.

For the past decade, many advances have been made in the field of AI planning (Smith *et al.* (2000) present an overview of recent planning, and scheduling, techniques). Nevertheless, much research in planning done even today has inherited something from those early systems: Certain key assumptions.

1.1 Classical Planning Assumptions

Planning, in the STRIPS system, meant stringing together a sequence of discrete operational steps, each of which causes a well specified change to a discrete model of the world state, such that the entire sequence transforms an initial world state into a final specified goal state [1]. The STRIPS system assumed (*i*) that the planner has complete information about all relevant aspects of the initial world state, (*ii*) that the planner has a perfectly correct and deterministic model of the effects of any action it can take and (*iii*) that no change in the world is ever caused by anything other than the plan put together by the planner. These assumptions are known as "the STRIPS assumptions" and the resulting class of planning problems are known as "classical planning".

Even under such harsh restrictions the planning problem is computationally difficult and sufficiently challenging to still be a research issue. At the same time, many potential practical planning applications have, not surprisingly, turned out to require one or more of the STRIPS assumptions to be relaxed (although, notably, some do not (Koehler 2000)). Doing this, without losing too much of what precious little efficiency classical planning has achieved, can be seen as the second challenge in planning.

A less frequently noted assumption made by the STRIPS system was that even though the planner had complete and deterministic information about the effects on the world state of any action it could take, it had no information about the utility of the actions with respect to achieving a particular goal (the intention being, after all, that it should reason this out by itself). Substantial improvements of the planner capabilities can be achieved if the planner is given domain- or problem-specific advice on how to use the available actions.

1.2 Planning Problem Modelling Languages

The model of action effects employed in the STRIPS system is fairly simple: An action is characterized by its *preconditions*, a set of atomic facts that have to hold in the world state for the action to be executable, and its *postconditions*, consisting of a set of atomic facts *added*, *i.e.* caused to hold true after completion the action, and another set of facts *deleted*, *i.e.* caused to cease being true after completion of the action.

This simple form of action description, also sharing the name "STRIPS", is still in common use in planning. The richer modelling language ADL (Pednault 1988) allows actions to have more complex preconditions (essentially first-order formulas, involving negation, disjunction and quantification) and effects that may depend on the state in which the action is taken. As long as states are finite, an ADL description can always be translated into STRIPS, though sometimes at the cost of making it exponentially larger.

PDDL (McDermott *et al.* 1998; Bacchus 2000) is a recent "standardization" of planning modelling languages. It contains STRIPS and ADL. PDDL 2.1 (Fox & Long 2002) extends PDDL with several novel features: (*a*) the use of real-valued functions in the world model, and actions whose preconditions include inequalities between expressions involving those functions[2], (*b*) the possibility of specifying the duration of actions for temporal planning and scheduling, and (*c*) the possibility to specify different kinds of plan metrics for optimal planning.

[1] Chapman (1987) showed that the planning problem is undecidable if the object domain is infinite, so since then it's been "a *finite* discrete model of the world state"

[2] This makes the world model essentially infinite, and therefore it is possible to specify undecidable problems in PDDL 2.1 (Helmert 2002).

1.3 Search

AI planning of almost all kind is heavily concerned with search. Pearl's book (1984) is a classic on the subject. Korf (1999; 1992) reviews standard search and heuristic search algorithms.

2. Classical Planning

Classical planning algorithms can be classified, among other things, according to how they structure the search space. Two very common such classes are *state space* planners and *plan space* planners. In the first class are both very early planners, *e.g.* STRIPS (Fikes & Nilsson 1971), and some successfull recent planners, *e.g.* Graphplan (Blum & Furst 1997) and its descendants IPP (Koehler *et al.* 1997) and STAN (Long & Fox 1999), HSP (Bonet & Geffner 2001), GRT (Refanidis & Vlahavas 1999) and FF (Hoffman 2000). In the second class are *causal link* planners, *e.g.* SNLP (McAllester & Rosenblitt 1991), UCPOP (Penberthy & Weld 1992), VHPOP (Younes & Simmons 2002) and constraint-based planners, *e.g.* HSTS (Muscettola 1994).

A third class of classical planners encode the planning problem as a problem of some other kind, *e.g.* SAT (Kautz & Selman 1992) or CSP (van Beek & Chen 1999), and solve this problem.

State of the Art In terms of speed, the best classical planners today are based on heuristic state space search (having quite thoroughly beaten the previously ruling causal link planners around half a decade ago). In terms of expressivity, the ideas of plan space search in the form of constraint-based planners are somewhat more successfull in tackling complex (*i.e.* "non-classical") domains.

It should be mentioned that domain-independent planning in general is for the most part not up to dealing with any problem of realistic size and complexity, and that in practical planning applications, domain-specific solutions are almost always used.

2.1 State Space Search and Heuristics

Search in the state space is most often done either by *forward-chaining*, *i.e.* searching from the initial state to the goal state (*a.k.a. progression*), or by *backward-chaining*, *i.e.* searching from the goal state to the initial state (*a.k.a. regression*).

The state space in any non-trivial planning problem is very large, so planners that search this space depend on having good heuristic guidance. Recent examples of successfull forward-chaining state space planners are UNPOP (McDermott 1996; 1999), HSP (Bonet & Geffner 2001) (a variant of HSP which does regression search), GRT (Refanidis & Vlahavas 2001b; 2001a) and FF (Hoffman & Nebel 2001; Hoffman 2001).

Graphplan (Blum & Furst 1997) does regression search in the state space, but makes use of something called a *planning graph* to compute the heuristic and to reduce branching. Much recent work in planning builds on the planning graph idea, by improving the efficiency of the basic Graphplan algorithm (Long & Fox 1999; Baioletti, Marcugini, & Milani 2000; Kambhampati 2000), extending it to more complex domains, as in IPP (Koehler *et al.* 1997), or by using the graph as a basis for finding good search heuristics, as in FF (Hoffman 2000) or (Nguyen & Kambhampati 2000).

An alternative to relying on heuristic guidance is to use such an efficient representation of states and sets of states that complete breadth-first search in the state space becomes possible. This approach, with Binary Decission Diagrams (BDDs) used to represent sets of states, is found in MIPS (Edelkamp & Reffel 1999) and MBP (Cimatti *et al.* 1997).

2.2 Plan Space Search and Partial-Ordered Planning

The plan space consists of incomplete and complete plans, which, in contrast to the notion of plan in the state space view, do not have to be sequences or actions or sets of parallel actions.

One alternative is to view the plan as a partially ordered set of actions. The potential advantage of this view is that one partially ordered set can represent many linear plans, and that the planner needs only to enforce the orderings that are absolutely necessary (whereas in a liner plan, many action orderings are quite arbitrary). This is called the principle of least commitment (Weld 1994), and underlies so called "causal link" planners, *e.g.* SNLP (McAllester & Rosenblitt 1991) and UCPOP (Penberthy & Weld 1992). Recently, causal link planning has been combined with heuristics similar to those used by state space search planners, with promising results (Nguyen & Kambhampati 2001; Younes & Simmons 2002).

2.3 Encoding Planning as a Different Problem

A different approach to solving a planning problem is to turn it into another kind of problem, via encoding. This approach was first used in SATPLAN (Kautz & Selman 1992; 1996; Ernst, Millstein, & Weld 1997) which compiles a planning problem into a propositional satisfiability problem. The potential advantage is that advances in general SAT-solving techniques can in this way be applied to planning.

A potential disadvantage is that one in the process of encoding loses some helpful information. To avoid this, one can apply successfull techniques from SAT solvers, *e.g.* local search and propagation, to the planning problem directly (Rintanen 1998; Hoos 1999).

Other problems into which planning has been encoded include CSPs (van Beek & Chen 1999) and integer programs.

The idea of encoding has also been combined with the plan graph idea (Kautz & Selman 1999; Do & Kambhampati 2000), which in a sense incorporates the heuristic information in the graph into the encoded problem.

2.4 Abstraction Planning

The idea of abstraction planning is to find first an *abstract* plan, containing only the most important actions, and then to iteratively *refine* it by "filling in the details", *i.e.* the less important actions. This idea goes back to early planning research (Sacerdoti 1974). More recent developments have focused on automatically finding the abstraction hierarchy, *i.e.* determining the "importance" of goals and actions (Knoblock 1992; 1994).

3. Domain Analysis

By analysing a planning problem in depth, much usefull information can be extracted and used to help guide the search for a plan. Various forms of domain analysis have been used in causal link planning (Peot & Smith 1993) and plan graph-based search (Fox & Long 1999; 2000).

Domain analysis is also being developed on its own, without direct coupling to any specific planner (Nebel, Dimopoulos, & Koehler 1997; Gerevini & Schubert 1998; Fox & Long 1998; Scholz 2000; Rintanen 2000).

3.1 Learning in Planning

Machine learning techniques can be used to extract useful knowledge, from solutions to similar problem instances in the past or from previous failed atempts to solve the present problem instance.

This has been used to improve planning efficiency, *e.g.* (Minton 1988; Ruby & Kibler 1989; Katukam & Kambhampati 1994; Kambhampati 2000), and to improve plan quality, *e.g.* (Pérez 1995; Estlin & Mooney 1997; Ambite, Knoblock, & Minton 2000). The PRODIGY planner (Veloso *et al.* 1995) incorporates several forms of learning.

A planning approach based more on learning than search is *case-based* planning, *e.g.* (Hammond 1986).

4. Planning in Complex Domains

Beside making planners faster and capable of handling larger problems, the second focus of work in planning has been making planners capable of dealing with more complicated problems than those that fall in the classical STRIPS framework.

The core STRIPS assumption is that of complete knowledge of the initial state and accurate deterministic models of actions, and relaxing this to planning under uncertainty is one direction to increase the scope of planners. Another fundamental restriction of the STRIPS and ADL models is the limitation to finite and discrete states, and to discrete actions: Thus, another direction of greater complexity is the introduction of *e.g.* numeric-valued state variables, metric time and the possibility of continuous change.

4.1 Planning with Metric Time

In classical planning actions are assumed to take "unit time". This assumption is not critically important, as long as there's no deadline too meet and one doesn't try to optimize the actual execution time, or *makespan*, of the plan.

An early planner to deal with actions of different duration and goals with deadlines is Deviser (Vere 1983). It's based on the idea of partial-order planning, in which the simple partial order over the actions in the plan are replaced by more complex constraints on their starting and ending times. The idea has been picked up in several later planners, *e.g.* TripTic (Rutten & Hertzberg 1993), Zeno (Penberthy & Weld 1994) (which also allows for some limited forms of continuous change) and IxTeT (Ghallab & Laruelle 1994; Laborie & Ghallab 1995).

The same approach is taken further in contraint-based planners like HSTS (Muscettola 1994; Jonsson *et al.* 2000) and *parc*Plan (Lever & Richards 1994), where the entire plan is represented in a form of constraint network. Planning with time (and resources) is the area of planning where local search has been applied most frequently. Local search planning may also be seen as a form of plan space search, but different in that intermediate plans are "complete but faulty", rather than incomplete. This approach makes it easier to handle complex forms of action, as in *e.g.* the ASPEN system (Chien *et al.* 2000). The ECAI 2000 workshop (Nareyek 2001) presents an overview.

TGP (Smith & Weld 1999) extends the Graphplan approach to planning with metric time, while Sapa (Do & Kambhampati 2001) and TP4 (Haslum & Geffner 2001) extend the heuristic state space search approach. Recently, temporal planning has becomme a very active area of research, and almost every classical planning approach has been adapted to deal with durative actions (the AIPS 2002 workshop (Fox & Coddington 2002) presents a good overview). The LPG planning system (Gerevini & Serina 2002) combines the plangraph representation with local search.

4.2 Planning with Resources

When numeric-valued state variables are introduced in planning, they're often thought of as *resources*.

Numeric state variables have been introduced in the Zeno planner (Penberthy & Weld 1994), based on partial-order planning, in RIPP (Koehler 1998), based on Graphplan, in the heuristic search planner GRT-R (Refanidis & Vlahavas 2000) and in planning based on encoding into mixed integer-linear programs (Rintanen & Jungholt 1999; Vossen *et al.* 1999) and a combination of LP and SAT (Wolfman & Weld 1999).

Many of the temporal planning systems mentioned above also support planning with resources to some extent.

4.3 Planning with Incomplete Information

Another often-critized assumption made by most planners is that all relevant information is available in the problem description, and that the effects of actions are perfectly predictable.

There have been several approaches to relaxing this assumption, by introducing probabilistic information and by generating *conformant* and *contingent* plans (a conformant plan is one that doesn't change depending on the circumstances, but nevertheless is built to succeed no matter what they are, while a contingent plan may contain "branches").

The probabilistic representation has been used with partial-order planners (Kushmerick, Hanks, & Weld 1994), with Graphplan (Blum & Langford 1999), but most of all in the form of Markov Decision Problems (Puterman 1994; Kaelbling, Littman, & Cassandra 1998).

Contingent and conditional planners have also been built based on partially ordered planning (Warren 1976; Peot & Smith 1992; Collins & Pryor 1995; Golden, Etzioni, & Weld 1994), on Graphplan (Smith & Weld 1998; Weld, Anderson, & Smith 1998), and on state space search, using heuristics (Bonet & Geffner 2000) and using a BDD representation (Cimatti, Roveri, & Traverso 1998).

The most conditional plan possible is one that selects the next action based only on the present situation, and knows what to do in every situation. Such a plan is called a *universal* plan (or a reactive controller). As long as the state space is finite, universal plans can be automaticallt synthesized (Musliner 1994; Musliner, Durfee, & Shin 1995; Kabanza, Barbeau, & St-Denis 1997; Betroli *et al.* 2001; Jensen, Veloso, & Bowling 2001), though they may grow exponentially large (Jonsson & Bäckström 1996).

A different approach, which makes use of domain-dependent knowledge, to synthesizing recative controllers is taken by McDemott, Beetz *et al.* (1992; 1994; 1998).

5. Domain-Dependent Planning

Planning applications in practice may be large and involve complicated actions, but they commonly also have a great deal of structure, known to people experienced with the domain. There are many potential plans that are (to the human domain expert) obviously useless, and sometimes simple criteria can be found for sifting out the "promising" partial solutions.

If this knowledge is encoded and given to the planner, it should help to speed up the process of finding a plan. This idea leads to what is called "domain-dependent planning": A simple and early example is the Blockhead planner (Kibler & Morris 1981). If, however, the planner is given too much help, it can no longer be said to be solving any problem: The advice becomes like a program that the planner is only executing (while this is certainly desirable from an application point of view, it does make the research aspect less interesting).

5.1 Hierarchical Task Network Planning

The idea behind Hierarchical Task Network (HTN) planning is to provide the planner with a set of skeletal, recursive "plan schemas", and restrict search in the space of plans to only those that conform to the schemas.

The HTN planning idea has been around for some time (Sacerdoti 1975; Tate 1977) and has been used in several applications (Wilkins 1990; Wilkins & desJardins 2000).

The UMCP (Erol, Hendler, & Nau 1994) and SHOP (Nau *et al.* 1999) planners provide a more general (less application-oriented) view of HTN planning.

5.2 Domain-Dependent Search Control

Many search-based planners allow the specification of domain-dependent heuristics and rules for controling (and hopefully, reducing) search.

Two recent planners, TLPlan (Bacchus & Kabanza 1995; 1996) and TALplanner (Doherty & Kvarnström 1999; Doherty, Kvarnström, & Haslum 2000) depend entirely on domain-specific search control, given in the form of logical formulas.

Planning Resources on the Web

Planning Systems

BLACKBOX Combines the plan graph representation with encoding in propositional logic.
`http://www.cs.washington.edu/homes/kautz/blackbox/`

FF Forward-chaining, non-optimal heuristic search. Supports at least part of ADL.
`http://www.informatik.uni-freiburg.de/~hoffmann/ff.html`

GRAPHPLAN Regression search, based on the plan graph representation. Does not accept PDDL input.
`http://www.cs.cmu.edu/~avrim/graphplan.html`

GRT Forward-chaining, non-optimal heuristic search, with some support for resources.
`http://www.csd.auth.gr/~lpis/GRT/main.html`

HSP, HSPr, HSP2 Variations on heuristic search planning.
`http://www.ldc.usb.ve/~hector/#software`

IPP Extends the GRAPHPLAN algorithm to part of ADL. A version called RIPP has some support for resources.
`http://www.informatik.uni-freiburg.de/~koehler/ipp.html`

MIPS Forward-chaining, breadth-first search planner that uses BDDs to represent sets of states efficiently.
`http://www.informatik.uni-freiburg.de/~edelkamp/Mips/index.html`

*parc*Plan Constraint-based planner, supporting time and resources. Only a demo (not the complete planner) can be downloaded.
`http://www.icparc.ic.ac.uk/parcPlan/`

PGP Extends GRAPHPLAN with probabilistic actions.
`http://www.cs.cmu.edu/~avrim/pgp.html`

SGP Extends GRAPHPLAN with uncertainty and sensing actions.

`http://www.cs.washington.edu/ai/sgp.html`

STAN Plan graph/heuristic search planner making much use of domain analysis for improved performance.

`http://www.dur.ac.uk/~dcs0www/research/stanstuff/stanpage.html`

TLPLAN Forward-chaining planner based on domain-dependent search control rules (specified in LTL).

`http://logos.uwaterloo.ca:/~fbacchus/tlplan.html`

UCPOP Partial-order planner, supporting part of ADL. Several variants exist of this planner, supporting *e.g.* time, resources and uncertainty, but most are not available.

`http://www.cs.washington.edu/ai/ucpop.html`

UMCP Generic HTN planner, implemented in LISP.

`http://www.cs.umd.edu/projects/plus/umcp/manual/`

Problem Collections

IPC The 1998, 2000 and 2002 International Planning Competitions are a source of problems (in PDDL).

`http://www.cs.toronto.edu/aips2000/`

`http://www.dur.ac.uk/d.p.long/competition.html`

General Links

AI Planning Systems More links to planning systems.

`http://www.csc.ncsu.edu/faculty/stamant/planning-resources.html`

`http://www.dai.ed.ac.uk/daidb/students/stevep/planref.html`

`http://www.isi.edu/~ariadne/cs541/index.html` (Course page with installation guides for several planning systems.)

PlanLib Links to planning and domain analysis systems, domains, test results and more.

`http://www.intellektik.informatik.tu-darmstadt.de/~planlib/`

Colloquium of the ETAI Electronic Journal on Planning & Scheduling.

`http://www.informatik.uni-ulm.de/ki/Etai/ec/ec.html`

UK Planning & Scheduling SIG `http://www.research.salford.ac.uk/plansig/`

References

Ambite, J.; Knoblock, C.; and Minton, S. 2000. Learning plan rewriting rules. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*. AAAI Press.

Bacchus, F., and Kabanza, F. 1995. Using temporal logic to control search in a forward chaining planner. In *Proc. 3rd European Workshop on Planning (EWSP'95)*.

Bacchus, F., and Kabanza, F. 1996. Planning for temporally extended goals. In *Proc. 13th National Conference on Artificial Intelligence (AAAI'96)*.

Bacchus, F. 2000. Subset of PDDL for the AIPS 2000 planning competition. `http://www.cs.toronto.edu/aips2000/pddl-subset.ps`.

Baioletti, M.; Marcugini, S.; and Milani, A. 2000. DPPlan: An algorithm for fast solution extraction from a planning graph. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*.

Beetz, M., and Grosskreutz, H. 1998. Causal models for concurrent reactive robot action plans. *Linköping Electronic Articles in Computer and Information Science* 3(7).

Beetz, M., and McDermott, D. 1992. Declarative goals in reactive plans. In *Proc. 1st International Conference on AI Planning Systems (AIPS'92)*.

Beetz, M., and McDermott, D. 1994. Improving robot plans during their execution. In *Artificial Intelligence Planning Systems: Proc. 2nd Conference (AIPS'94)*.

Betroli, P.; Cimatti, A.; Roveri, M.; and Traverso, P. 2001. Planning in nondeterministic domains under partial observability via symbolic model checking. In *Proc. 17th International Joint Conference on Artifical Intelligence (IJCAI'01)*.

Blum, A., and Furst, M. 1997. Fast planning through graph analysis. *Artificial Intelligence* 90(1–2):281 – 300. `http://www.cs.cmu.edu/~avrim/graphplan.html`.

Blum, A., and Langford, J. 1999. Probabilistic planning in the Graphplan framework. In *Proc. 5th European Conference on Planning (ECP'99)*.

Bonet, B., and Geffner, H. 2000. Planning with incomplete information as heuristic search in belief space. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*. `http://www.ldc.usb.ve/~hector/`.

Bonet, B., and Geffner, H. 2001. Planning as heuristic search. *Artificial Intelligence* 129. `http://www.ldc.usb.ve/~hector/` or `http://www.cs.ucla.edu/~bonet/`.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence* 32:333 – 377.

Chien, S.; Rabideau, G.; Knight, R.; Sherwood, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barrett, T.; Stebbins, G.; ; and Tran, D. 2000. ASPEN – automating space mission operations using automated planning and scheduling. In *Proc. 6th International Symposium on Technical Interchange for Space Mission Operations*.

Cimatti, A.; Giunchiglia, E.; Giunchiglia, F.; and Traverso, P. 1997. Planning via model checking: A decision procedure for AR. In *Proc. 4th European Conference on Planning (ECP'97)*. http://www.mrg.dist.unige.it/bin/publistsearch/?cimatti.

Cimatti, A.; Roveri, M.; and Traverso, P. 1998. Automatic OBDD-based generation of universal plans in non-deterministic domains. In *Proc. 15th National Conference on Artificial Intelligence (AAAI'98)*. http://www.mrg.dist.unige.it/bin/publistsearch/?cimatti.

Collins, G., and Pryor, L. 1995. Planning under uncertanity: Some key issues. In *Proc. 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*. http://www.pobox.com/~pryor/papers/keys-and-boxes.ps.

Do, M. B., and Kambhampati, S. 2000. Solving planning-graph by compiling it into CSP. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*. http://rakaposhi.eas.asu.edu:80/papers.html.

Do, M. B., and Kambhampati, S. 2001. Sapa: A domain-independent heuristic metric temporal planner. In *Proc. 6th European Conference on Planning (ECP'01)*.

Doherty, P., and Kvarnström, J. 1999. TALplanner: An empirical investigation of temporal logic-based forward-chaining planner. In *Proc. TIME'99*.

Doherty, P.; Kvarnström, J.; and Haslum, P. 2000. Extending TALplanner with concurrency and resources. In *Proc. 14th European Conference on Artificial Intelligence (ECAI'00)*.

Edelkamp, S., and Reffel, F. 1999. Deterministic state-space planning with BDDs. Technical Report 120, Institut für Informatik, Albert-Ludwigs Universität, Frieburg. http://www.informatik.uni-freiburg.de/tr/1999/Report120/.

Ernst, M.; Millstein, T.; and Weld, D. 1997. Automatic SAT-compilation of planning problems. In *Proc. 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*. http://www.cs.washington.edu/homes/kautz/blackbox/.

Erol, K.; Hendler, J.; and Nau, D. 1994. UMCP: A sound and complete procedure for hierarchical task-network planning. In *Proc. 2nd International Conference on AI Planning Systems (AIPS'94)*. AAAI Press.

Estlin, T., and Mooney, R. 1997. Learning to improve both efficiency and quality of planning. In *Proc. 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, 1227–1233. Morgan Kaufmann.

Fikes, R., and Nilsson, N. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2:189 – 208.

Fox, M., and Coddington, A., eds. 2002. *AIPS 2002 Workshop on Planning for Temporal Domains*.

Fox, M., and Long, D. 1998. The automatic inference of state invariants in TIM. *Journal of AI Research* 9:367 – 421.

Fox, M., and Long, D. 1999. The detection and exploitation of symmetry in planning domains. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*. http://www.dur.ac.uk/~dcs0www/research/stanstuff/papers.html.

Fox, M., and Long, D. 2000. Utilizing automatically inferred invariants in graph construction and search. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*. AAAI Press.

Fox, M., and Long, D. 2002. PDDL2.1: An extension to PDDL for expressing temporal planning domains. http://www.dur.ac.uk/d.p.long/pddl2.ps.gz.

Gerevini, A., and Schubert, L. 1998. Inferring state constraints for domain-independent planning. In *Proc. 15th National Conference on Artificial Intelligence (AAAI'98)*.

Gerevini, A., and Serina, I. 2002. LPG: A planner based on local search for planning graphs with action costs. In *Proc. 6th International Conference on AI Planning Systems (AIPS'02)*.

Ghallab, M., and Laruelle, H. 1994. Representation and control in IxTeT, a temporal planner. In *Proc. 2nd International Conference on AI Planning Systems (AIPS'94)*.

Golden, K.; Etzioni, O.; and Weld, D. 1994. Omnipotence without omniscience: Efficient sensor management for planning. In *Proc. 12th National Conference on Artificial Intelligence (AAAI'94)*.

Hammond, K. 1986. CHEF: A model of case-based planning. In *Proc. ??th National Conference on Artificial Intelligence (AAAI'86)*.

Haslum, P., and Geffner, H. 2001. Heuristic planning with time and resources. In *Proc. 6th European Conference on Planning (ECP'01)*.

Helmert, M. 2002. Decidability and undecidability results for planning with numerical state variables. In *Proc. 6th International Conference on AI Planning Systems (AIPS'02)*.

Hoffman, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of AI Research* 14. http://www.informatik.uni-freiburg.de/~hoffmann/ff.html.

Hoffman, J. 2000. A heuristic for domain independent planning and its use in an enforced hill-climbing algorithm. Technical Report 133, Institut für Informatik, Albert-Ludwigs-Universität, Freiburg.

http://www.informatik.uni-freiburg.de/~hoffmann/ff.html.

Hoffman, J. 2001. FF: The fast-forward planning system. *AI Magazine* 22(3):57 – 62.

Hoos, H. 1999. SAT encodings, search space structure and local search performance. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*.

Jensen, R.; Veloso, M.; and Bowling, M. 2001. OBDD-based optimistic and strong cyclic adversarial planning. In *Proc. 6th European Conference on Planning (ECP'01)*.

Jonsson, P., and Bäckström, C. 1996. On the size of reactive plans. In *Proc. 13th National Conference on Artificial Intelligence (AAAI'96)*.

Jonsson, A.; Morris, P.; Muscettola, N.; Rajan, K.; and Smith, B. 2000. Planning in interplanetary space: Theory and practice. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*.

Kabanza, F.; Barbeau, M.; and St-Denis, R. 1997. Planning control rules for reactive agents. *Artificial Intelligence* 95(1).

Kaelbling, L.; Littman, M.; and Cassandra, A. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101:99 – 134.

Kambhampati, S. 2000. Planning graph as a (dynamic) CSP: Exploiting EBL, DDB and other CSP search techniques in Graphplan. *Journal of AI Research* 12:1 – 34.

Katukam, S., and Kambhampati, S. 1994. Learning explanation-based search control rules for partial order planning. In *Proc. 12th National Conference on Artificial Intelligence (AAAI'94)*, 582–587. AAAI Press.

Kautz, H., and Selman, B. 1992. Planning as satisfiability. In *Proc. 10th European Conference on Artificial Intelligence (ECAI'92)*. http://www.cs.washington.edu/homes/kautz/blackbox/.

Kautz, H., and Selman, B. 1996. Pushing the envelope: Planning, propositional logic and stochastic search. In *Proc. 13th National Conference on Artifical Intelligence (AAAI'96)*. http://www.cs.washington.edu/homes/kautz/blackbox/.

Kautz, H., and Selman, B. 1999. Unifying SAT-based and graph-based planning. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*. http://www.cs.washington.edu/homes/kautz/blackbox/.

Kibler, D., and Morris, P. 1981. Don't be stupid. In *Proc. 7th International Joint Conference on Artificial Intelligence (IJCAI'81)*.

Knoblock, C. A. 1992. An analysis of ABSTRIPS. In *Artificial Intelligence Planning Systems: Proc. 1st International Confrence (AIPS'92)*.

Knoblock, C. 1994. Automatically generating abstractions for planning. *Artificial Intelligence* 68:243 – 302.

Koehler, J.; Nebel, B.; Hoffman, J.; and Dimopoulos, Y. 1997. Extending planning graphs to an ADL subset. In *Proc. 4th European Conference on Planning (ECP'97)*, 275 – 287.

Koehler, J. 1998. Planning under resource constraints. In *Proc. 13th European Conference on Artificial Intelligence (ECAI'98)*.

Koehler, J. 2000. Elevator control as a planning problem. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00)*.

Korf, R. 1992. Search. In *Encyclopedia of Artificial Intelligence*. Wiley.

Korf, R. 1999. Artificial intelligence search algorithms. In *Handbook of Algorithms and Theory of Computation*. CRC Press. chapter 36.

Kushmerick, N.; Hanks, S.; and Weld, D. 1994. An algorithm for probabilistic least-commitment planning. In *Proc. 12th National Conference on Artificial Intelligence (AAAI'94)*.

Laborie, P., and Ghallab, M. 1995. Planning with sharable resource constraints. In *Proc. 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*.

Lever, J., and Richards, B. 1994. *parc*PLAN: A planning architecture with parallel actions, resources and constraints. In *Proc. 9th International Symposium on Methodologies for Intelligent Systems (ISMIS'94)*. http://www.icparc.ic.ac.uk/parcPlan/publications.html.

Long, D., and Fox, M. 1999. Efficient implementation of the plan graph in STAN. *Journal of AI Research* 10:87 – 115.

McAllester, D., and Rosenblitt, D. 1991. Systematic nonlinear planning. In *Proc. 9th National Conference on Artificial Intelligence (AAAI'91)*.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL - the planning domain definition language. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computitational Vision and Control.

McDermott, D. 1996. A heuristic estimator for means-ends analysis in planning. In *Proc. 3rd International Conference on Artificial Intelligence Planning and Scheduling (AIPS'96)*.

McDermott, D. 1999. Using regression-match graphs to control search in planning. *Artificial Intelligence* 109:111 – 159.

Minton, S. 1988. *Learning Effective Search Control Knowledge: An Explanation-Based Approach.* Kluwer Academic Publishers.

Muscettola, N. 1994. Integrating planning and scheduling. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling.* Morgan-Kaufmann.

Musliner, D.; Durfee, E.; and Shin, K. 1995. World modeling for the dynamic construction of real-time control plans. *Artificial Intelligence* 74(1).

Musliner, D. 1994. Using abstraction and nondeterminism to plan reaction loops. In *Proc. 12th National Conference on Artificial Intelligence (AAAI'94).*

Nareyek, A., ed. 2001. *Local Search for Planning and Scheduling*, volume 2148 of *LNAI.* Springer.

Nau, D.; Cao, Y.; Lotem, A.; and Muñoz-Avila, H. 1999. SHOP: Simple hierarchical ordered planner. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99).*

Nebel, B.; Dimopoulos, Y.; and Koehler, J. 1997. Ignoring irrelevant facts and operators in plan generation. In *Proc. 4th European Conference on Planning (ECP'97)*, 338 – 350.

Nguyen, X., and Kambhampati, S. 2000. Extracting effective and admissible state space heuristics from the planing graph. In *Proc. 17th National Conference on Artifical Intelligence (AAAI'00).* http://rakaposhi.eas.asu.edu:80/papers.html.

Nguyen, X., and Kambhampati, S. 2001. Reviving partial order planning. In *Proc. 17th International Joint Conference on Artifical Intelligence (IJCAI'01).*

Pearl, J. 1984. *Heuristics.* Addison-Wesley.

Pednault, E. 1988. Synthesizing plans that contain actions with context-dependent effects. *Computational Intelligence* 4:356 – 372.

Penberthy, J., and Weld, D. 1992. UCPOP: A sound, complete, partial order planner for ADL. In *Proc. International Confrence on Knowledge Representation and Reasoning (KR'92).*

Penberthy, J., and Weld, D. 1994. Temporal planning with continous change. In *Proc. 12th National Conference on Artificial Intelligence (AAAI'94).*

Peot, M., and Smith, D. 1992. Conditional nonlinear planning. In *Artificial Intelligence Planning Systems: Proc. International Confrence (AIPS'92).*

Peot, M., and Smith, D. 1993. Threat-removal strategies for partial-order planning. In *Proc. 11th National Confrence on Artificial Intelligence (AAAI'93).*

Pérez, A. 1995. *Learning Search Control Knowledge to Improve Plan Quality.* Ph.D. Dissertation, School of Computer Science, Carnegie Mellon University. Also available as Technical Report CMU-CS-95-175.

Puterman, M. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley.

Refanidis, I., and Vlahavas, I. 1999. A domain-independent heuristic for STRIPS worlds based on greedy regression tables. In *Proc. 5th European Conference on Planning.*

Refanidis, I., and Vlahavas, I. 2000. Heuristic planning with resources. In *Proc. 14th European Conference on Artificial Intelligence.*

Refanidis, I., and Vlahavas, I. 2001a. The GRT planner. *AI Magazine* 22(3):63 – 65.

Refanidis, I., and Vlahavas, I. 2001b. The GRT planner: Backward heuristic construction in forward state-space planning. *Journal of AI Research* 15.

Rintanen, J., and Jungholt, H. 1999. Numeric state variables in constraint-based planning. In *Proc. 5th European Conference on Planning (ECP'99).* http://www.informatik.uni-freiburg.de/~rintanen/CV.html.

Rintanen, J. 1998. A planning algorithm not based on directional search. In *Principles of Knowledge Representation and Reasoning: Proc. 6th International Conference (KR'96).* http://www.informatik.uni-freiburg.de/~rintanen/CV.html.

Rintanen, J. 2000. An iterative algorithm for synthesizing invariants. In *Proc. 17th National Conference on Artificial Intelligence (AAAI'00).* http://www.informatik.uni-freiburg.de/~rintanen/CV.html.

Ruby, D., and Kibler, D. 1989. Learning subgoal sequences in planning. In *Proc. 11th International Joint Conference on Artificial Intelligence (IJCAI'89)*, 609–614.

Rutten, E., and Hertzberg, J. 1993. Temporal planner = nonlinear planner + time map manager. *AI Communications* 6(1):18 – 26.

Sacerdoti, E. 1974. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence* 5:115 – 135.

Sacerdoti, E. 1975. The nonlinear nature of plans. In *Proc. 4th International Joint Conference on Artificial Intelligence (IJCAI'75)*, 206 – 214.

Scholz, U. 2000. Extracting state constraints from PDDL-like planning domains. In *Proc. AIPS Workshop on Analyzing and Exploiting Domain Knowledge for Efficient Planning*, 43–48.

Smith, D., and Weld, D. 1998. Conformant Graphplan. In *Proc. 15th National Conference on Artificial Intelligence (AAAI'98).*

Smith, D., and Weld, D. 1999. Temporal planning with mutual exclusion reasoning. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99).*

Smith, D.; Frank, J.; and Jonsson, A. 2000. Bridging the gap between planning and scheduling. *Knowledge Engineering Review* 15(1).

Tate, A. 1977. Generating project networks. In *Proc. 5th International Joint Conference on Artificial Intelligence (IJCAI'77)*, 88 – 93.

van Beek, P., and Chen, X. 1999. CPlan: A constraint programming approach to planning. In *Proc. 16th National Conference on Artificial Intelligence (IJCAI'99)*.

Veloso, M.; Carbonell, J.; Perez, A.; Borrajo, D.; Fink, E.; and Blythe, J. 1995. Integrating planning and learning: The PRODIGY architecture. *Journal of Theoretical and Experimental Artificial Intelligence* 7(1). `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/prodigy/Web/papers.html`.

Vere, S. 1983. Planning in time: Windows and durations for activities and goals. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 5:246 – 267.

Vossen, T.; Ball, M.; Lotem, A.; and Nau, D. 1999. On the use of integer programming models in AI planning. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*.

Warren, D. 1976. Generating conditional plans and programs. In *Proceedings of the Summer Conference on AI and Simulation of Behaviour (AISB'76)*.

Weld, D.; Anderson, C.; and Smith, D. 1998. Extending Graphplan to handle uncertainty & sensing actions. In *Proc. 15th National Conference on Artifical Intelligence (AAAI'98)*.

Weld, D. 1994. An introduction to least commitment planning. *AI Magazine* 15(4):27 – 61.

Wilkins, D., and desJardins, M. 2000. A call for knowledge-based planning. In *Proc. AIPS Workshop on Analysing and Exploiting Domain Knowledge for Efficient Planning*.

Wilkins, D. 1990. Can AI planners solve practical problems? *Computational Intelligence* 6(4):232 – 246.

Wolfman, S., and Weld, D. 1999. The LPSAT engine & its application to resource planning. In *Proc. 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*.

Younes, H., and Simmons, R. 2002. On the role of ground actions in refinement planning. In *Proc. 6th International Conference on AI Planning Systems (AIPS'02)*. `http://www-2.cs.cmu.edu/~lorens/vhpop.html`.