AI Planning: Solutions for Real World Problems

R.S.Aylett
G.J.Petley
Centre for Virtual Environments, Salford University
Salford, UK, M5 4WT
R.S.Aylett@salford.ac.uk Gary@angmar.iti.salford.ac.uk
P.W.H.Chung
B. Chen
Chemical Engineering Dept., Loughborough University
Loughborough, UK, LE11 3TU
J.Soutter
BG, Gas Research Centre
Ashby Road, Loughborough, UK
David W. Edwards
Chemical Engineering Dept., Loughborough University

Abstract

This paper argues that AI planning is a technology ripe for use on real world problems as shown by a number of current applications. An introduction is given to AI planning, and then followed by a consideration in detail of the successful application of this technology to generating operating procedures for chemical plants. A description is given of the methodology for developing planning domains and finally the results of its application to operating procedure synthesis discussed.

1.0 Introduction

Historically there has been something of a gulf between the Knowledge Based Systems community and workers in Artificial Intelligence (AI) planning. The former have demonstrated notable success in solving real-world problems, often using fairly general-purpose Knowledge Representations (KRs) - e.g. rules, frames, predicate calculus - and general purpose inferencing techniques on those KRs - e.g. data-driven reasoning and goal-driven reasoning. Planning was seen as just another application area for techniques such as rules or constraint propagation [Stefik 81a, 81b; Hayes 90].

Meanwhile workers in AI planning investigated KRs based on the planning-specific concepts of actions, pre-conditions and post-conditions together with algorithms specifically intended for sequencing actions and dealing with interactions

between them. Indeed, given that a general-purpose planning system provides a predetermined problem-solving method and a set of generic actions and data-structures, it is closer to an automated knowledge acquisition tool like KNACK [Klinker 88] or OPAL [Musen et al 87] than it is to an expert system shell.

However the AI planning community was often seen as preoccupied with theory and unsuccessful in applying that theory to real-world problems. In this paper we argue that AI planning has now reached a significant maturity in which the specialised techniques it offers are being successfully applied to real-world problems. We support this argument by discussing our own work in applying AI planning to the generation of operating procedures for chemical process plant, and we draw some general lessons for others applying this technology.

The maturity of AI planning technology can be seen in the ARPA/Rome Labs Planning Initiative (ARPI), which was launched in the US from 1990, and is currently at phase IV, with total funding to date of $70 million. Early in this programme, a number of integrated feasibility demonstrations showed, in operational environments, the relevance of generative planning in the domain of military air campaign planning. A 1994 US department of commerce report stated that the deployment of a single logistics support aid called DART (the first of a series of demonstrators within ARPI) during the Desert Storm campaign paid back all US government investment in AI/KBS research over a 30 year period.

One may also cite a number of very successful Space systems. For example, the Optimum-AIV system [Aarup et al 92] is now used operationally by the European Space Agency for planning the integration of equipment into the loading bays of the Ariane rocket. NASA routinely uses the planning system DPLAN [Chien et al 97] for operations planning of its Deep Space Communications Network, and Multimission VICAR [Chien 94] to organise vision-processing modules for scientists extracting data from space probes or satellites.

On May 17th, 1999, NASA activated the Remote Agent experiment (RAX) on board the Deep Space 1 (DS1) spacecraft. RAX is an autonomous agent architecture that comprises of three modules: a constraint-based Planning and Scheduling system that generates plans from first principles using a temporal domain model, a Smart Executive for executing the plans, and a model-based Mode Identification and Recovery system which carries out fault diagnosis and suggests mode reconfigurations to the Smart Executive. Experiments carried out during May have shown that the planning system is capable of successfully controlling the spacecraft given very high-level commands from the ground, and of coping with both equipment and sensor failures.

The domains of these systems are all rather different: what they have in common is that they solve difficult and potentially combinatorial sequencing problems. We will show that generating operating procedures for chemical process plant has these same characteristics, and we argue that AI planning is a good solution for problems involving difficult and potentially combinatorial sequencing problems.
This success forms part of the impetus behind an increasing number of international planning-centred events: the biennial international AI Planning and Scheduling Conference, the transformation of the European Planning and Scheduling Workshop

series into a full conference from 1998 and the recent (October 98) setting up of an EU funded network of excellence for planning, PLANET.

2.0 AI Planning

Planning is the task of choosing and ordering the sequence of actions (steps) needed to achieve a set of objectives [Weld 94]. We distinguish it from scheduling, in which the main issue is resource allocation for the steps found in a plan. A planning problem is usually defined by a domain model and by two states of that model: the initial state and the goal state. The domain model describes the objects in a domain as well as the actions, these actions are normally described by operators, that can be performed with the objects and the constraints on these actions.The initial state describes the state of the domain immediately before any actions have been carried out, with the goal state describing the facts which must be true after the plan has been completed.

The planning task can be split into two closely related subtasks. The first subtask involves finding the steps needed to solve each objective of the procedure. For example, consider three blocks A, B, and C all on a table, the planner has an action which allows it to move a block from the table to on top of another block if both blocks are clear. Now possible actions are move A onto C or move B onto C, therefore if the goal is to get A on B on C, then moving A onto C will mean the planner will have to back track and choose action move B onto C.

The second subtask involves detecting and resolving conflicts between the steps needed to achieve different objectives. This can be carried out by reordering conflicting actions, inserting actions to resolve the conflict, or by replanning, as discussed in [Chapman 87].

During planning, the search space can become enormous if no techniques are used to limit its size. Least-commitment planning [Penberthy & Weld 92] is an approach to reducing search spaces. It encompasses non-linear planning, in which only essential ordering constraints between actions are introduced, leaving all others unordered (in pseudo-parallel), allowing a whole set of plans to be represented at once. It also includes constraining the possible instantiations of an object used in the plan rather than committing to a particular instantiation. Hierarchical Task Network (HTN) planning also reduces the search space by representing a problem as a hierarchy of tasks that need to be achieved, allowing a plan or part of a plan to be represented by a high level of abstraction, with the lower levels, and more detailed part of the plan, left for later expansion.

In this paper we will look at how AI planning has been successfully applied to a real word problem of generating operating procedures for chemical plants.

3.0 Operating Procedure Synthesis

Operating Procedure Synthesis (OPS) is the task of automatically generating operating procedures for chemical process plants through the use of computer algorithms. During the last twenty years, OPS research has been carried out by the chemical engineering community into the automatic planning of OPS [Rivas & Rudd 74;

Ivanov et al 80; Foulkes et al 88; Fusillo & Powers 88; Aelion & Powers 91; Crookes & Macchietto 92], rather than by AI researchers. Yet there is an intuitively obvious relationship between an operating procedure and the output of an AI Planning system.

The steps in a procedure are actions to be carried out; the procedure is designed to take a plant from a start state to an end state; each step in the procedure must be carried out in the appropriate state and will result in a new state. Only [Aelion & Powers 91] of the works referenced above have seriously considered AI Planning technology (in this case a linear STRIPS type engine) and modern hierarchical and least-commitment techniques have not been applied. This has limited the scope of the systems developed to 'toy' plant domains.

3.1 Chemical Engineering Planner (CEP)

The Chemical Engineering Planner (CEP), has been developed over the last five years, initially as a PhD project [Soutter 97] and in the last three years as part of the EPSRC-funded INTergrating OPerability (INT-OP) project*,**. CEP has been developed incrementally through case studies of increasing scope and complexity, and as we show is now more capable than any of the systems referenced earlier. We will only summarise the structure of CEP in this paper.

CEP divides the tasks involved in OPS into three areas: planning using operators, the handling of safety considerations and valve sequencing. The first two of these three areas are handled by a state-of-the-art least-commitment planner [Penberthy & Weld 92], which uses the concept of ‘goals of prevention’ [Soutter & Chung 96] to prevent actions being incorporated into the operating procedure that will take a plant through any unsafe states. Safety is clearly a particular concern in a chemical plant domain: a plan which moves the plant to a desired end-state is unacceptable if - for example - along the way explosive gases have been mixed together. ‘Goals of prevention’ are defined as safety restrictions as part of the overall description of the plant.

CEP deals with valve sequencing as a special case. A characteristic of the opening and closing of valves in a chemical plant - actions required in order to produce flows of chemicals to specified vessels or other components - is that the effect of the action at a particular valve is dependent on associated actions at other valves. However an assumption of the standard AI planner representation of actions is that the effect of an action should be represented through its post-conditions - and should therefore always be the same. Valve operations violate this assumption [Aylett et al 98].

Therefore, valve sequencing is handled by a specialist module in CEP that uses an approach that we call ‘action synergy’ and is based on work by [O’Shima 78; Foulkes et al 88]. A maze searching algorithm is used to find a route for a flow between given start and end points. All the valves around this route are then closed and finally those actually on the route are opened. Thus CEP can be seen as a general-purpose AI planner with domain-related specialist additions.

* Academic Partners: Loughborough University and Salford University.

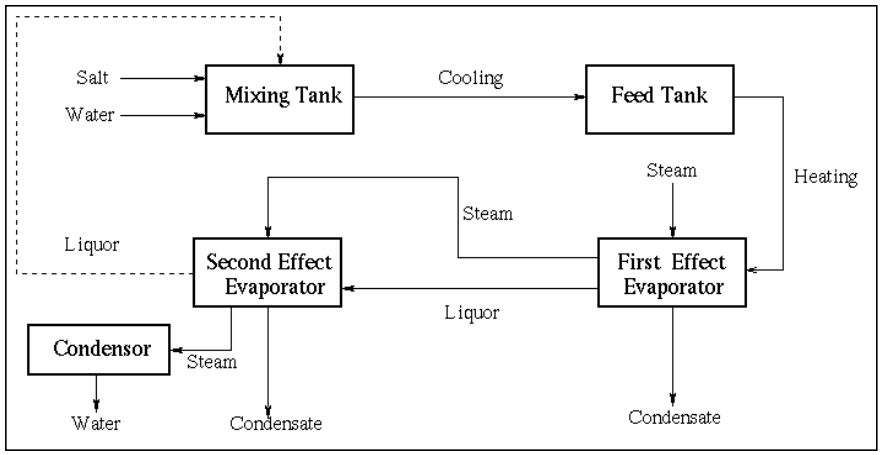

FIGURE 1. Basic Process for Double Effect Evaporator Test Rig

** Industrial Partners: BG Plc, BP, Cogsys Ltd, ICI, Science Systems Ltd, and TCCL.

3.2 OPS Domains

An incremental approach has been taken to our case studies. Initially we looked at 'toy' problems discussed in operating procedure synthesis literature. The next level of case study was a test rig in the Chemical Engineering department at Loughborough University, a Double Effect Evaporator (DEE). Finally, the successful completion of the DEE case study led to another move up in domain complexity to actual chemical plant case studies, an ICI ammonia plant and a BP acetic acid plant.
In this paper we will describe the results from the DEE, with the chemical plant case studies discussed in less detail due to industrial secrecy. A brief outline of the process for the DEE test rig is shown in Figure 1.

The DEE is a complex domain and is much nearer to a real-world chemical plant than the domains used in previous work. Not only does the DEE set-up contain a larger number of components (67) than in most previous domains, but the number of different types of equipment is also large (15), with valves, controllers, pumps, heaters, coolers, evaporators, feed tank, mixing tank and a barometric condenser.

4.0 Planning Methodology

Two closely-coupled steps are involved in applying a planner to a new domain: knowledge acquisition and domain modelling. In the DEE case-study, knowledge was acquired by reading the documentation on the test rig, visiting the double effect evaporator installation, and by interviewing a Loughborough University colleague with an understanding of the working of the test rig. While there are important issues here we will not touch on them in this paper.

We will however discuss domain modelling in more detail, as the amount of time and effort required to construct a particular domain model is a major obstacle

to the use of AI planning in the solution of real-world problems [Chien et al 96]. If CEP is to be used by design engineers for real industrial plant, it must be straightforward to construct the domain model. We therefore report the lessons learned from our case studies.

After knowledge acquisition, the information acquired must be transformed into a form that the planner CEP can understand and use. All planners require the first four parts listed below in their domain model, and CEP introduces a new element called pairs:

- Domain description (plant model)
- Planning operators (actions that can be carried out)
- Safety restrictions (constraints on states)
- Pairs (knowledge association)
- Domain problem (the plant operations to be carried out)

4.1 Domain Description - A planner will require a description of what is in the domain. For example, CEP uses an implementation of a hierarchical frame-based description developed during earlier work at Loughborough [Chung 93] to model individual components in a particular plant. The manual entry of these descriptions for every component in a particular plant using CEP's syntax is non-trivial: it is both very time-consuming and prone to error. An automatic system was therefore developed for producing the domain description. A popular drawing package, AutoCAD, has been adapted to provide the standard chemical engineering equipment symbols for the user. When a new piece of equipment is added to the plant diagram, a text box appears prompting the user to add the name and connections for it. Thus on completion of the drawing, the necessary information has been collected to allow the automatic creation of a domain description file.

4.2 Planning Operators - The next stage is to develop the planning operators that describe the actions that can be used to produce a plan. Where the domain description gives the static content and layout of a plant, planning operators define its behaviour. A CEP operator consists of a goal(s) that can be achieved when the pre-condition(s) for the operator are true - essentially the STRIPS [Fikes & Nilsson 71] representation still widely used in AI planning systems in spite of all the other changes in the field since then. An example operator is shown in Figure 2. However, the primitive operator was supplemented with macro operators, which allow information about the order in which the pre-conditions must be satisfied to be entered [Aylett et al 98].

Operator development is a time-consuming and difficult part of domain development and one on which there is minimal guidance in the available literature. Yet the correctness and efficiency of the planning process in a domain depends very heavily on operator definition.

The more generic the operators, the fewer required, and, even more important, the greater the scope for re-use. Therefore, a consequence of producing generic planning operators is the opportunity to create a library of operators for objects commonly found in a particular domain. For example, in OPS domains, operators

FIGURE 2. Operator - Control Valve

```
operator OperateControlValve
{
      aperture ?state1;
      aperture ?state2;
      controller ?c;

      ?state1 != ?state2;

      achieve
         * aperture of ?c is ?state2;
      using
         aperture of ?c is ?state1;
      end

      print (?n) 'Set controller ' ?c ' and turn '
                        [name of ?state2 is ?n;];
}
```

can be organised around a class hierarchy of equipment found in chemical plants, such as valves, pumps, heat exchangers etc. [Petley et al 98]. A suitably comprehensive library standardises the design of operators by reducing the task to one of selection, with possibly some scope for specialisation, and this is the route taken in our work. Indeed, without such a library it is hard to see how CEP could be applied to new plants in an industrial context since one could not expect a plant design engineer to design the planning operators from scratch. The DEE case study discussed here provided the first generic operators for the library which was extended by the two later case studies, the ICI and BP plants, as new components were encountered. When creating generic operators around a component class hierarchy, one must also make operators specific enough in relation to the domain description to prevent vast amounts of search when instantiating pre and post-conditions [Aylett & Jones 96] and to capture appropriate differences in functionality. For example, in the DEE an operator at the level of vessel would be too general since there are significant differences in functionality between, say, an evaporator and mixing tank, and at instantiation the planner would have to consider every vessel in a plant.

A hierarchical structure is required for operators in this domain - we found there to be a substantial difference in granularity between the task requirement level (e.g. start-up plant in single-evaporator mode) and the primitive action level (e.g. open valve HV5) in OPS domains. This shows [Aylett & Jones 96] a clear need for a hierarchical structure in all the operators in the model. The task of starting up the plant in double-evaporator mode is represented as a high level operator with an effect which expands into a set of goals satisfied by operators at the next level of expansion. These in turn may expand the effects further to a new level of operators. The result is a goal-hierarchy which represents the declarative structure of planning in the domain [Aylett et al 97].

Figure 3. Example Safety Restriction

```
restrictions
{
      prevent
            state of GP1 is started;
            state of GC1 is stopped;
      end
}
```

4.3 Safety Restrictions - most planners will require these, for example, in CEP the constraints prevent unsafe situations from occurring during planning through the specification of incompatible states. An example of a safety restriction for the DEE specifies that glass preheater GP1 is not allowed to be started if the state of the glass cooler GC1 is stopped, see Figure 3. The reason for this restriction is to prevent energy from entering the plant - a heater on - before there is a mechanism for energy to leave the plant - a cooler on. Safety restrictions allow issues of safety to be dealt with separately from the design of planning operators.

4.4 Domain Problems - Finally, a definition of a problem in the domain for the planner to solve is required. The problem definition requires two domain states, one at the start and the other at the end of the problem. In general a domain state is defined by setting the state of each component in the plant, though in practice this is not necessary for end-states in which only the high-level goals to be achieved are specified. From this a planner, such as CEP, will produce a plan consisting of a sequence of actions that bring about the specified change in the plant, if one exists for the given operators and restrictions. The AutoCAD tool used to provide the domain description also provides a method for defining the state of the equipment for a domain state, allowing the domain problem descriptions to be developed in parallel with the domain description.

4.5 Pairs - A chemical plant is a large and complex configuration of numerous components, producing enormous search spaces when planning. Engineering Line Drawings (ELDs) contain extra information associating elements of plant specific knowledge which CEP can use to narrow down the search space. For example, from the DEE domain state that heat exchanger HE1's source of heat is from Input4, and this input is supplying steam.

5.0 Results

CEP successfully produced operating procedures for the DEE domain. A single model of the domain allowed procedures to be created for the start-up, shutdown and the isolation of pieces of equipment for maintenance.

The start-up procedure generated by CEP for the DEE contained 52 steps, with 19 the total number of operators used in planning. The time taken to generate the operating procedure was under 5 seconds on a Sparc 5. An example section of a operating procedure generated by CEP can be seen in Appendix A. Moreover CEP proved capable of finding alternative procedures via backtracking at the users request.

5.1 Comparison with Previous Systems

The planner CEP has been used to produce operating procedures using AI planning for a domain more complex than any other previously attempted. The work of the early 80s [Ivanov et al 80, Kinoshita et al 81] using state-graphs limited sample problems to plants containing a handful of valves because of the number of states they generated: 20 valves each with 2 states produces 1,048,576 nodes in a state-graph. This demonstrates that OPS is indeed a difficult and combinatorial sequencing problem.

Other workers used larger plant [Rivas & Rudd 74] but only considered valves and not vessels. A real-world nuclear fuel processing plant was used in [Crookes & Macchietto 92], but this work concentrated on optimising a hand-generated plan. CEP has successfully solved every sample problem reported in the OPS literature except for those requiring numerical calculations. We therefore argue that CEP's success in our case studies demonstrates a big step in the state-of-the-art for OPS.

5.2 Quality of Results

The procedures produced were evaluated by the domain experts who had been used for the knowledge acquisition. The expert for the DEE found the generated procedures adequate - in the sense that the start-up procedure would successfully start up the plant. This is an important result for a domain of this complexity and validates the overall approach of using AI planning technology on this problem. However he also found the generated procedures in some ways naive - in the sense that they were not always identical to the ones an expert would produce.

A major example of this concerned the use of the glass preheater (GP1 in the top-left of Figure 3). It is possible to start up the plant without using this preheater, and accordingly CEP originally generated a procedure that did not use it. The reasons for using the preheater during start-up are: the temperature of the brine can be increased in stages, protecting the glass lined vessels, and the control of the temperature of the brine entering the first evaporator is easier with two heaters. An expert in operability, seeing that the design contained a glass preheater, would infer that it was there for the purpose of start-up and accordingly use it. This operability knowledge does not appear to be representable within the confines of planning operators and we are currently examining the issue in more detail.

5.3 Linking and Ordering Actions

The output of a partial-order planner such as CEP is a plan-network in which only those actions which must follow each other are ordered with respect to each other. Actions which may be taken in any order appear in parallel in such a representation.

It is usual for the last step in an AI planner to be the transformation of the plan net into a linear sequence, since usually only one-step-at-a-time execution is planned for. This process is known as linearization. The plan net supports user interaction in the linearization process since it shows what actions can be moved in a particular linearization and which cannot. The re-ordering of actions may be desirable for two reasons. Firstly, grouping actions together for operating a certain piece of equipment makes the plan clearer. Secondly, a 'better' plan may be produced by taking the actions in a certain order. For example, if two valves have to be opened manually, then these actions should be together if they are geographically next to each other in the plant. CEP cannot currently take geographical proximity into account, since the ELD from which it works contains only the topological relationship of equipment.

Linearization of the CEP's output is now grouped into sequences of actions where each sequence is linked to a high level goal. This results in the output appearing in chunks related to the operation of certain parts of the plant and allows the human operator to understand it more easily. An extension would allow the user to reorder actions manually, subject to the constraints of the plan net, seen as highly desirable by prospective users.

5.4 Generic Operators

In the DEE case study, every component is now operated by a generic operator, all of which are found in a component operator library. Moreover, thirteen of these operators were reused for a subsequent case-study using the back-end loop section of the ICI ammonia plant, and six for the corrosion metal removal system of the BP acetic acid plant, demonstrating the value of a library of planning operators.

A more important issue concerns the extent to which real-world components in process plant are generic. For example, of the two heat exchangers in the double effect evaporator test rig, one has an extra out port for the steam used to heat the material passing through the exchanger. In real plants, components are sourced from a variety of manufacturers and so there may be differences in the way each is constructed and operated. If this variation is empirically shown to be very large, a generic library of operators might be impossible. An obvious approach to this problem is to consider attaching the library of operators more firmly to the component hierarchy, with the use of object-oriented inheritance and specialisation mechanisms to control variation.

6.0 Conclusions

A number of conclusions can be drawn from our research, some specific to the domain of Process Plant Operating Procedure Synthesis, and others of more general relevance to industrial applications of AI planning.

A positive conclusion is that the case studies validate the use of state-of-the-art AI planning techniques in OPS. As discussed above, this has made it possible to deal successfully with a large and complex domain where conventional KBS technology might well have struggled with the combinatorial sequencing.

During the case studies, it became clear that a number of improvements were needed: in the linearization process, in the valve sequencing component and in the incorporation of general operability knowledge: all of these improvements have now been made. We argue that this indicates that though general-purpose hierarchical least-commitment planning algorithms are powerful, real domains also require domain-specific problem-solving along with a great deal of domain-specific knowledge. Hopefully, in the same way as work in knowledge-engineering methodologies have identified specific approaches to different types of diagnosis, work in a wider range of real-world planning domains will begin to establish abstract categories in such domains which will support a more principled approach to the choice of planning technologies for particular problems [Valente 95, Aylett & Jones 96].

A number of knowledge engineering issues arose from the DEE case study. Firstly, the time and effort required to develop the domain was substantial, with about 20 person-days of effort involved in developing the operators without any tools (though a proportion of this was due to a learning curve which would be climbed quicker next time as a result of this experience). Development of tools to assist in domain development is, we believe, vital to the use of AI planning to solve real-world problems. Planning attacks problems more complex than the average KBS and therefore tools are much more important. The automatic generation of the domain description for CEP from AutoCAD was one tool developed, and the library of generic operators derived from the case study is another even more important one. Not discussed here were other tools to help the user in running CEP, collectively called CEP-Run. Finally, CEP-Tool was created to give further user support to the development of operators, providing operator templates, an operator editor and a interface for interacting with the operator library. Validation and verification tools such as those described in [Chien 96] are also important.

The production of a library of generic operators in the DEE case study illuminated a particular problem. It is often possible to solve problems in a particular planning domain by ad hoc fixes, frequently in the planning operators. As CEP's capabilities were increased, so it became possible to solve each problem in a more general and principled way. Thus the ability to produce a library of generic operators is not only an indispensable tool for domain development in the future, it is also, we argue, indirectly a measure of the adequacy of a planner.

We argue that AI planning technology has now reached a level of maturity where it can be successfully applied to difficult real-world problems. Just as KBS technology in general has made a powerful contribution to the management of manufacturing systems, so AI planning has the potential to solve problems in this area previously seen as too complex to be tackled successfully.

In particular, Operating Procedure Synthesis is a new applications area for AI planning, but we suggest one of considerable promise. The INT-OP project developed a system that can be used in a real-industrial environment to produce operating procedures and with further development could be used to produce operating procedures earlier in the plant life-cycle with real savings in time and effort. We note that plant operating procedures are only one example of the need for accurate, efficient and safe procedures in manufacturing industries and see many possibilities for

extending the AI planning approach to the generation of other types of operational procedure.

References

1. Aarup, M.; Arentoft, M.M.; Parrod, Y.; Stokes, I.; Vadon, H. & Stader, J. (1992) Optimum -AIV: A Knowledge-based Planning and Scheduling System for SpaceCraft AI. In: Intelligent Scheduling, ed M.Zweben & M.S.Fox, pp451-470, Morgan Kaufmann.
2. Aelion, V. & Powers, G.J. (1991) A Unified Strategy for the Retrofit Synthesis of Flowsheet Structures for Attaining or Improving Operating Procedures. In: Computers and Chemical Engineering, vol. 15 no 5: pp349-360, Pergamon.
3. ARPI (ARPA Planning Initiative) http://arpi.isx.com/.
4. Aylett, R.S. & Jones, S.D. (1996) Planner and Domain: Domain Configuration for a Task Planner. International Journal of Expert Systems vol. 9 no2: pp279-318, JAI Press.
5. Aylett, R.R; Petley, G.J; Chung, P.W.H; Soutter, J. & Rushton, A. (1997) Planning and Chemical Plant Operating Synthesis: A Case Study. eds. S.Steel & R.Alami. Recent Advances in AI Planning. Springer Lecture Notes in AI no1348 1997 pp 39-51
6. Aylett, R.S; Soutter, J; Petley, G.J; Chung, P.W.H. & Rushton, A. (1998) "AI Planning in a Chemical Plant Domain". Proceedings, ECAI '98, pp 622-26.
7. Chapman, D. (1987) Planning for Conjunctive Goals. Artificial intelligence, 29, 1987
8. Chien, S, (1994) "Using AI Techniques to Automatically Generate Image Processing Procedures: A Preliminary Report" Proc. AIPS 94, Chicago IL, pp219-24.
9. Chien, S.A. (1996) Static and Completion Analysis for Planning Knowledge Base Development and Verification. Proceedings, 3rd International Conference on AI Planning Systems, pp53-61, AAAI Press.
10. Chien, S.A; Hill, R.W; Wang, X; Estlin, T; Fayyad, K.V. & Mortenson, H.B.(1996) Why Real-world Planning is Difficult: a Tale of Two Applications. In: New Directions in AI Planning, M.Ghallab & A.Milani, eds, IOS Press, Washington DC, pp 287-98.
11. Chien, S.A; Govindjee, A; Estin, T; Wang, X; Griesel, A. & Hill, R. jnr. (1997] Automated Generation of Tracking Plans for a Network of Communication Antennas. Proc. 1997 IEEE AeroSpace Conference, Aspen, CO,Feb 1997
12. Chung, P.W.H. (1993) Qualitative Analysis of Process Plant Behavior. Proceedings Industrial and Engineering Applications of AI and Expert Systems, ed. P.W.H.Chung, G.Lovegrove & M.Ali, pp277-83 Gordon & Breach.
13. Crooks, C.A. & Macchietto, S. (1992) A Combined MILP and Logic-Based Approach to the Synthesis of Operating Procedures for Batch Plants. Chemical Engineering Communications 114: pp117-144.
14. Fikes, R.E. & Nilsson, N.J. (1971) Strips: A New Approach to the Application of Theorem-Proving to Problem-Solving. Artificial Intelligence 2: pp189-208.
15. Foulkes, N.R.; Walton, M.J.; Andow, P.K. & Galluzo, M. (1988) Computer Aided Synthesis of Complex Pump and Valve Operations. Computers and Chemical Engineering, 12: pp1035-1044.
16. Fusillo, R.H. & Powers, G.J. (1988) Operating Procedure Synthesis using Local Models and Distributed Goals. In: Computers in Chemical Engineering, vol 12 no 9/10: pp 1023-1034, Pergamon.
17. Hayes, C.C. (1990) Machining Planning: a Model of an Expert Level Planning Process. PhD Thesis, Carnegie-Mellon University, Pittsburgh Pa.
18. Ivanov, V.A.; Kafarov, V.V.; Perov, V.L. & Reznichenko, A.A. (1980) On Algorithmiza-

tion of the Start-up of Chemical Productions. Engineering Cybernetics, 18: pp104-110.
19. Kinoshita, A.; Umeda, T & O'Shima, E. (1981) An Algorithm for Synthesis of Operational Sequences of Chemical Plants. 14th Symposium on Computerized Control and Operation of Chemical Plants, Vienna, Austria.
20. Klinker, G. (1988) KNACK: Sample-driven Knowledge Acquisition for Reporting Systems. In: Marc cus, S. (ed) Automating Knowledge Acquisition for Expert Systems, Ch5:125-74. Kluwer.
21. Musen, M; Fagan, L; Combs, D. & Shortliffe, E. (1987) Use of a domain-model to drive an interactive knowledge-editing tool. International Journal of Man-Machine Studies, 26:105-121.
22. O'Shima, E. (1978) Safety Supervision of Valve Operations. In: Journal of Chemical Engineering of Japan, vol 11 no 5: pp390-395.
23. Penberthy, J.S. & Weld, D.S. (1992) UCPOP: A Sound, Complete, Partial Order Planner for ADL. Proceedings of the 3rd International Conference on Knowledge Representation and Reasoning.
24. Petley, G.J; Aylett, R.S; Chung, P.W.H. & Rushton, A. (1998) "Development of a Reusable Operator Library for Chemical Plant Domains" Proceedings, 17th Workshop of the UK Planning and Scheduling SIG, University of Huddersfield, ISSN 1368-5708, pp145-156.
25. Rivas, J.R. & Rudd, D.F. (1974) Synthesis of Failure-Safe Operations. In: AIChE Journal, vol. 20 no 2: pp 320-325.
26. Soutter, J. & Chung, P.W.H. (1996) Partial Order Planning with Goals of Prevention. proceedings, 15th Workshop of the UK Planning and Scheduling SIG, vol. 2: pp300-11, John Moores University, Liverpool, UK.
27. Soutter, J. (1997) "An Integrated Architecture for Operating Procedure Synthesis." PhD thesis, Loughborough University, Loughborough, LE11 3TU, UK.
28. Stefik, M. (1981a) Planning with Constraints (MOLGEN: Part 1) Artificial Intelligence 16.
29. Stefik, M (1981b) Planning and Meta-Planning (MOLGEN: Part 2) Artificial Intelligence 16(2): pp141-170.
30. Valente, A. (1995) Knowledge-level Analysis of Planning Systems. ACM SIGART Bulletin Special Issue, 6(1).
31. Weld, D. (1994) "An introduction to least-commitment planning". AI Magazine, Winter,

pp27-61.

Appendix A

Set controller LRC_5 and turn on.(46)

Set controller FRC_6 and turn on.(45)

Set controller LRC_7 and turn on.(44)

Set controller FRC_8 and turn on.(43)

Set controller TRC_9 and turn on.(42)

Open valve HV7.(185)

Open valve HV25.(232)

Achieved: Flow route from Input2 to MT1 for processWater.(228)

Open valve HV6.(211)

Turn on pump P3.(210)

Achieved: Flow route from MT1 to MT1 for brine.(206)

Mixing in MT1.(194)

Open valve HV32.(202)

Achieved: Flow route from Input8 to MT1 for salt.(198)

Close valve HV32.(203)

Achieved: Stopped flow of salt to MT1.(197)

Warning: Make sure that all the salt needed has entered system before stopping-flow.(192)

Mixing brine in MT1.(190)

Open valve HV16.(59)

Open valve HV20.(60)

Achieved: Flow route from E1 to Output10 for steam.(55)

Open valve HV4.(183)

Open valve HV5.(182)

Turn on pump P3.(180)

Turn on pump P1.(181)

Achieved: Flow route from MT1 to E1 for brine.(176)

Open valve HV11.(187)

Open valve HV2.(189)

Open valve HV10.(188)

Turn on pump P2.(186)

Achieved: Flow route from E1 to MT1 for brine.(172)

Open valve HV26.(104)

Achieved: Flow route from GP1 to Output100 for steam.(100)

Open valve HV31.(111)

Open valve HV24.(112)

Achieved: Flow route from Input3 to Output4 for coolingWater.(107)

Open valve HV22.(90)

Achieved: Flow route from Input5 to GP1 for steam.(86)

Open valve HV28.(97)

Achieved: Flow route from Input5 to TD1 for steam.(93)

Wait until air flushed out of GP1.(77)

Close valve HV26.(146)

Achieved: Flow route for steam (heat source) through GP1 estab-

lished.(76)
77. Achieved: Flow route from TD2 to Output8 for steam.(161)
78. Open valve HV27.(130)
79. Achieved: Flow route from HE1 to Output800 for steam.(126)
80. Open valve HV23.(82)
81. Achieved: Flow route from Input4 to HE1 for steam.(67)
82. Open valve HV29.(123)
83. Achieved: Flow route from Input4 to TD2 for steam.(119)
84. Wait until air flushed out of HE1.(30)
85. Close valve HV27.(148)
86. Achieved: Stopped flow of steam through Output800.(147)
87. Achieved: Flow route for steam (heat source) through HE1 established.(29)
88. E1 is at operational temperature.(8)