Technical Report Supplement

O-Plan Project Evaluation Experiments

Brian Drabble, Terri Lydiard & Austin Tate

Approved for public release; distribution is unlimited

Artificial Intelligence Applications Institute
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
United Kingdom

April 30, 1997

ARPA-RL/O-Plan/TR/33 Version 1

Contents

1 Introduction

The aim of this report is to provide details of a number of experiments which have been carried out on the O-Plan project to date and to provide pointers to potential evaluation experiments for the future. During each of the three phases of the O-Plan project a great deal of experimentation has taken place to verify and validate the O-Plan model of planning and the components of the O-Plan system. However, most of the experimentation to date has taken place in the absense of a project evaluation framework in which the results can be analysed and related to the domains in which the project has been working. In order to resolve this issue the project has developed and initial approach to project evaluation based around a series of cell experiments. The cell experiments are identified from an evaluation matrix which relates planning domain features (the rows) to planner technology features in plan representation and reasoning (the columns). An initial version of the evaluation matrix has already been produced [4] and forms deliverable `E-1` from the project. Each of the experiments described in Section 2 provides details of the aim and results of experiment together with pointers to further work and experimentation. By catagorising the existing body of experimental knowledge within the O-Plan project it becomes possible to identify the types of experiment which are most appropriate to the O-Plan system and provide pointers to the types of experiments which should be carried out in the remainder of the project.

1.1 Experimentation Framework

This section describes the experimentation framework being adopted and the generic types of experiments being carried out. As described earlier a number of generic domain features were used to create the columns of the project's evaluation matrix and a series of planner technologies were used to define the rows. The aim of the project in developing this matrix was not to conduct every possible experiment but to be selective in its choice experiments. Experiments will be chosen which relate directly to the needs of the project's TIE partnerships and to the input the project is providing to the series of IFD's (e.g IFD-5 and IFD-6).

Each cell of the matrix will be used to record the outcome of an experiment. An experiment will be in one of three kinds:

- **Validation**:
  A validation experiment will determine whether a specific technique (or series of techniques) can be used to satisfy a domain requirement. In the simplest experiments the result with be either `yes` or `no`. For example, can a spatial reasoning system handle metric distance constraints between two specific points on a map? However, some experiments may provide a qualitative measure of how well the technique handled the domain requirement.

- **Comparison**:
  A comparison experiment will show how one technique performs in relationship than another in handling a specific domain requirement. For example, fixed non-sharable

resources can be handled using typed world state conditions or by specific reasoning about the allocation and deallocation of resources. An experiment could seek to determine which technique is better and where appropriate describe the quantitative or qualitative advantage it has.

- **Scalar**:
  A scalar experiment will determine the size and complexity of a problem the technique is capable of handling. In the simplest experiments the result with be a numerical value above which the techniques fails to cope, e.g. when the number of fixed unit resources exceeds 200 typed world state conditions may fail to find a solution. However, some experiments may provide more detailed measures of the complexity of the problem e.g. number of constraints involved before the technique becomes unusable. For example, the technique may fail with far fewer fixed unit resources if there are greater than 20% involved in more complex constraints.

A number of simple experiments have been conducted to validate the approach and the methodology we are using. The list of experiments described in this section will increase as more experiments are conducted. Here the list should be considered as an illustration of the proposed cell experiment design method.

2 Cell Experiments

2.1 Experiment 1: Use of Branch 1/Branch N estimators to guide issue selection

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2
The aim of the experiment was to validate that Branch1/Branch N estimators provided support to identify the most constrained issues in the plan state.

Assumptions and Background:
Experimental evidence suggests that least cost flaw repair (LCFR) techniques improve on other issue selection techniques in enabling larger and more complex problems to be solved [7].

Description of Experiment:
The experiment was to introduce branch1/branchN estimator field into each agenda entry. O-Plan's controller used this estimator first to decide which agenda entries were most constrained. Further domain information e.g. level information was used to resolve tie breaks between agenda entries with the same branch1/branchN estimators. The suite of test problems provided with the O-Plan release system was used for this experiment.

Summary of Results:
The outcome of the experiment was that LCFS techniques do provide a way of minimising solution time under certain circumstances. There is evidence to suggest that certain domains in which there are only ever single choices may cause the planner to deep dive in its search for a solution – getting stuck a poor search branches.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.2 Experiment 2: Use of Level information to guide issue selection

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1
The aim of the experiment was to validate that taking level information into account when triggering issues on the agenda would avoid the problem of the planner having to back track because it had committed to early to a choice of contributor to satisfy a condition.

Assumptions and Background:
Experimental evidence [11] suggests that delaying certain planning issues until all potential

contributors and deletors have been added to the plan enables large problems to be solved quicker. The hypothesis is that this technique avoids the planner committing early to a potential contributor and then needing to backtrack later when the choice becomes invalidated.

Description of Experiment:
The Task formalism compiler of the O-Plan system was modified to generate level information concerning world state conditions and effect. This was used to validate the domain description to ensure no domain level coding errors were present. O-Plan's controller was modified to hold back agenda entries which required further potential contributors and deletors to be added to the plan state. The suite of test problems provided with the O-Plan release system was used for this experiment.

Summary of Results:
The outcome of the experiment was that level information was useful and allowed a number of planning issues e.g. action expansion, condition satisfaction, etc to be delayed until a their correct time. In a number of cases there seemed to be conflict between the advice from the Branch1/Branch N estimators and that from the level information. This needs to be investigated further.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.3 Experiment 3: Mixed Initiative Planning Modalities

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to interact with the user and provide timely and appropriate responses.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
Experimental evidence derived from a number of planning research projects have identified the need for richer user interactions between the planning system and the user. The hypothesis behind the experiment is that the O-Plan system (through the KS-USER knowledge source and other modalities) has the ability to handle the needs of a Mixed Initiative Planning (MIP) framework.

Description of Experiment:
The experiment was carried using the Pacifica NEO domain and by allowing the user to set of number of choice points e.g. schema, variable binding and backtrack point to be under their control or the systems. The experiment also allowed the user to accept the decisions of the system or to preempt the system by requesting access to the plan state while the planner was running.

Summary of Results:
The outcome of the experiment was that the O-Plan system contained a number of modalities which were capable of supporting a wide range of user roles. These ranged from providing

user support in the area of identifying valid domain objects for binding variables to identifying suitable back track points when the current plan state was invalidated. The KS-USER knowledge source was able to access the information needed by the user and to interact with the user at a level at which they felt appropriate for their role.

Reference to Results:
No further details of the experiment are provided in [5].

2.4 Experiment 4: Or-Tree Merging and Pruning

Type: Scalar

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
Experimental evidence has shown that O-Plan generates a large number of options in dealing with the ascertion of conditions and effects. However, it is often the case that some of these options are inconsistent and that O-Plan could reduce the size of the search space by not exploring inconsistent options. In addition, there may be certain common options between different conditions and these must be asserted in the plan state. The hypothesis behind the experiment was to investigate different data structues (referred to in O-Plan as `or-trees`) inorder to decrease the size of the search space by identifying commonalities within the same or-tree and promoting them to the top of the tree as a single option across a number of alternatives.

Description of Experiment:
A number of changes were made to the `or-tree` data structure manipulated by the planning system. The `or-trees` contain the various options for dealing with the satisfaction of conditions and effects within the plan state. The algorithms which were explored were as follows:

- A basic `or-tree merger` that, when pruning branches, notices direct inconsistencies but not ones that require "propagating" constraints. This turns out not to be good enough for general use, because the resulting merged trees can be too large. (The same problem will occur with a more sophisticated merge algorithm, but in fewer cases.)
- Another algorithm looked for common steps (the same link, bind, etc) in separate branches of an `or-tree`, since common steps might be done first with branch-1 = 1. But there turned out to be hardly any common steps that were easily found in our current test suites; and rather than doing more work to extract common steps, it might be easier to build different or-trees in the first place.

Summary of Results:
The results of the experiment showed that while the techniques looked promising there were insufficient examples in the current test domains to validate this position. This issue needs

to be considered together with the general problem of how or-trees are constructed through further experimentation.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.5 Experiment 5: Condition Types

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
The main aim of the experiment was to take a fresh look at condition types and in particular the need for each type and the ways in which they should be handled in future implementations of O-Plan. The reason for this reappraisal was to address our concerns that there was confusion and criticism in the technical literature [2] about the use of the various condition types, and that a great deal of effort was required in encoding some domains for what seemed to be a small gain in search efficiency or in the quality of plans being presented.

The use of domain knowledge to restrict the plan search space is vital in any large scale problem, as the use of syntactic information concerning a condition is inadequate. The O-Plan team believe that one effective way to provide this knowledge to a planner is via condition types. Some condition type information can be gathered by lexical analysis of a problem definition. However, at present AI planning researchers know of no way to automatically deduce some of the information we can gather from user-defined types and conditions.

Description of Experiment:
The experiment was centred around the need for each particular condition types. The first point which was addressed was to provide three statements for each O-Plan condition type:

- **Purpose**: This describes the condition in domain terms for use by the domain encoder and describes the circumstances under which the condition should be used.
- **Definition**: This describes the condition in planner terms and describes in more detail how the planner goes about dealing with the condition type on behalf of the domain encoder.
- **Examples**: This clarifies of the use of a condition type.

Summary of Results:
In providing a description of each condition type in terms of its purpose and definition it became necessary to define further the meaning of a plan level and the plan circumstances in which a condition could be evaluated (i.e., when to trigger the agenda entry to have the condition

satisfied). The original definition of a plan level was too loose and vague to be used with the emerging definition of condition types and as a result a cleaner and more precise definition of a level was produced. The time at which an agenda entry is released (or triggered) for processing is very important in the search for a plan. The function of the triggers is to ensure the agenda entry is released for processing when it is possible to process the agenda entry in the planning process. By developing a clearer understanding of the ways in which conditions can be satisfied and maintained it was possible to define a cleaner and more precise definition of triggers.

Reference to Results:
Full details of this evaluation can be found in [11] and the results were published in [11].

2.6 Experiment 6: Economy of Force

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
There had been discussion within the ARPI community during 1993-4 which indicated a belief that so-called "generative planners" were inherently incapable of finding to solutions to problems in which the choice of a single action (operator schema) was necessary to address two or more separate problem requirements. This is sometimes called "economy of force". It can be one of the domain elements of evaluation to guide choice of better plans. Ginsberg at the University of Oregon had provided a simple island evacuation domain description in which such a single action choice was necessary to find the shortest solution to a problem[1].

Description of Experiment:
O-Plan does contain all economy of force solutions in its search space, as it is designed to be systematic in preserving search space completeness (modulo restrictions on the search space deliberately encoded by a domain writer through features provided for this purpose - such as condition typing). In order to prove this, the example from Ginsberg was coded in O-Plan TF and provided to O-Plan. As expected, this demonstration showed that O-Plan is easily able to find solutions to such problems.

Summary of Results:
The O-Plan condition satisfaction procedure (Question Answering) and the Operator Schema choice routines in O-Plan do in fact currently choose between open choices using a single criteria, but they preserve all choices systematically. These choices are available to the search space controller in O-Plan. The O-Plan design allows for the incorporation of heuristic prioritisation of choices made by the operator schema choice function and the pre-ordering of choices available via Question Answering to satisfy conditions. Such heuristic prioritisation is anticipated to support a range of domain dependent elements of evaluation (as described in [6]). This can

[1] Personal Communication.

include economy of force (or as we term it "kill-two-birds-with-one-stone") choice prioritisation. Note that it is not possible to build this heuristic in to a general purpose planner as a hard wired prioritisation routine, since in some domains economy of force is to be avoided. It can also run counter to other preferences such as robustness in plans.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.7 Experiment 7: Missionary and Cannibals

Type: Validation/Scalar

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
Scientists at Rome Laboratory used the Missionary and Cannibals Problem during 1993-4 to compare O-Plan and SIPE-2 [8]. While the Missionary and Cannibals Problem is not in the problem class for which O-Plan is designed (since it is essentially a mathematical *puzzle*), we used a range of Missionary and Cannibals Problem descriptions in O-Plan TF to demonstrate features of O-Plan prior to support for numeric handling and compute conditions (for external function support [10]) being added.

Description of Experiment:
Following the addition of numeric and compute condition support to O-Plan in version 2.2 (the second release to the ARPI CPE in July 1994), the Missionary and Cannibals Problem was recoded to act as a test domain for these features and to show that improved handing of the domain was possible.

Summary of Result:
These experiments were done with version 2.1 of O-Plan – the first release to the ARPI CPE. This showed that the Missionary and Cannibals Problem could be encoded using successor arithmetic.The Missionary and Cannibals TF encodings for the early and later experiments are available in the O-Plan release within the `demo/tf` directory.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.8 Experiment 8: Spanner

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
O-Plan includes handling for the satisfaction of conditions within operator schemas, where the introduction of actions to satisfy the conditions turns out to require the insertion of new actions into the plan before the temporal scope of the schema which contains the condition [11]. It has been the hypothesis for some time to explain that other planners designs are not allowing this possibility in their search spaces. This means that they are *designed* to be incomplete. Some plans that a domain encoder might expect to be possible will not be admitted. The benefits for these planners is that their search spaces can be significantly smaller.

Description of Experiment:
O-Plan provides support which will allow all solutions to be found including those needing the introduction of actions into a plan which "span" the area from the start of the plan to the point at which the condition is needed within the operator schema expansion (rather than just being in the gap between the beginning of the operator schema expansion and the point where the condition is needed).

A simple domain description called `spanner.tf` was written to describe a very simple domain in which the "spanning" capability is required in a planner. This domain is not amenable to solution by other planners designed with the more restricted definition of the legal temporal scope for the insertion of new actions to satisfy an achievable condition. This domain when run on O-Plan shows that O-Plan correctly identifies solutions requiring this capability.

Summary of Result:
Adding this capability to O-Plan has some serious consequences for comparative trials of O-Plan versus other planning systems. As far as we are aware, O-Plan is the only planner to have identified and remedied this problem. The search spaces introduced by handling it make for larger numbers of choices for a range of domains, some of the worst being the "puzzle" orientated domains used by many researchers to test their systems. Even simple problems in large plans can lead to very many more open choices if this capability is added.

Up to and including version 2.3 of O-Plan, the final release to the ARPI CPE from the O-Plan project's work in July 1995, the default handling for achieve conditions assumed that the system should allow for "spanning" solutions to be found. It is anticipated that a future release of O-Plan will alter the default handling to limit solutions to the temporal scope of the operator expansion in which the condition is introduced. However, O-Plan will continue to provide the more comprehensive "spanning" solution as necessary, and this will be able to be switched on by simply altering a default to the TF Compiler – using `achieve_after_point`.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.9 Experiment 9: Dealing with Plan State Variables

Type: Scalar

Domain Columns Addressed: 3.6.3

The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
The aim of this experiment was to investigate the use of maintaining tighter information on the possible values for different plan state variables. A Plan State Variables (PSV) can be restricted to disallow a certain value or to require that its value be different from that of another PSV. The latter case is called a "not-same" constraint in O-Plan. The problem of dealing with the co-designation/non-codesignation of constraints has been a topic of research for many years. Others researchers [1] has developed schemes which attempt to solve parts of this problem. The work of the O-Plan system aims to develop research in this area further.

Description of Experiment:
Each PSV has a "possibles-cache" that lists the values the PSV might take. Restrictions can remove values from the possibles-cache. If the PSV is left with only one possible value, it must be bound to that value; if it's left with zero, the plan is invalid.

The PSV Manager was changed to extract more information from *combinations* of not-same constraints. This was done only when a restriction is added to a PSV, leaving two or more values in PSV's possibles-cache. Then the PSV Manager looks at the PSVs listed in variables' not-sames constraints to check how many of the variables possible values they might take in combination.

The basic idea can be explained by an example. Suppose P-1, P-2, and P-3 all have A and B as their only possible values. Suppose P-2 and P-3 must have different values and that we then restrict P-1 to be different from both P-2 and P-3. Clearly, with three variables and only two possible values, there aren't enough values to go around. The aim is to detect cases of this sort and force the planner to abandon invalid plans earlier.

Summary of Result:
This change to the PSV Manager did not result in a noticeable increase in overall run-time and significantly reduced the number of O-Plan problem solving cycles required for certain tasks. For instance, the Pacifica task Blue Lagoon went down from 259 O-Plan cycles to 124. A more significant effect was that some tasks became practical (in terms of acceptable run time) for the first time.

Reference to Results:
No further details of the experiment are provided in this version of the document.

2.10 Experiment 10: Agenda Choice

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
The aim of this series of experiment was to investigate the impact on the search space of different schemes for choosing ready to run entries from the agenda. The agenda contains "issues" that must be resolved in order to construct a complete plan: actions to be expanded, conditions to be satisfied, variables to be bound, etc. The order in which the issues (agenda entries) were processed can affect how quickly a plan is found and which plan is found. The different schemes tried were aimed at significantly improving the planner's performance by paying attention to plan levels and by exploiting heuristics that had been developed in earlier Edinburgh planning work.

Description of Experiment:
These heuristics include Branch1/BranchN factors which are used to select the most constraining choice next [3]. These have also been studied recently as the "least cost flaw repair" heuristic by Joslin/Pollack [7] who showed that it does lead to search space reductions. Recent work reported, for example at IJCAI-95 in Montreal, has begun the refinement of these methods.

Summary of Result:
The outcome and results of the experiment were mixed. As anticipated, no simple fixed prioritisation scheme improves performance on all problems. An adaptive and opportunistic scheme is what we are seeking which makes use of constraints in the plan and domain knowledge. The experiments also highlighted problems with individual techniques and these were as follows:

Plan Levels

One problem with using levels is that many of the current O-Plan TF domain definitions were written without a clear hierarchical model and a certain amount of rethinking concerning their encoding is required. In addition we are aware that the O-Plan agenda choice priority mechanism is being used to ensure the planner follows a certain planning "algorithm" (e.g., expands are done before certain types of condition are satisfied). It would be better if these mechanisms were separated from other types of choice to allow improved search control. This is the subject of future O-Plan research.

Branch-1 and Branch-N Estimators

O-Plan maintains two estimators of the branching factors for agenda entries – Branch-1 and Branch-N. Branch-1 is the number of "top level" possibilities that will be considered when an issue is processed. Branch-N is an estimate of the possible final number of alternatives for this issue. When branch-1 is 1, there is only one possibility and the planner has a single committed choice. There is an intuitive and informal argument to the effect that the Planner should prefer committed choices and process them first. All issues on the agenda will have to be processed. Some will create branches in the search space, and all remaining issues will have to be processed in all branches. If committed choices are done first, they will be processed only once. Moreover, they may constrain the plan, thus making things easier (in a sense) when processing other issues. There is also some empirical evidence that it helps to prefer forced moves [7].

However, the experiment showed that preferring committed choices can make things worse. Here, it is important to distinguish between two questions:

1. Is it better overall to process forced moves first?
2. Does it always make things better, or are there some cases where it makes things worse?

The aim of the experiment was to address the second question, not the first. Joslin's work [7] addressed the first problem.

To see that things *can* get worse locally, consider the following example. Suppose the agenda includes items A and B, that A has branch-1 = 1 while B has branch-1 > 1, and that A and B are independent in the sense that processing one will not affect how we process the other. Now suppose that when we process B we will always reach a dead end, so that the Planner must backtrack, and that processing A will not reach a dead end. If we process A before B, the time spent processing A will be wasted. This lead to the consideration of the cost involved in the processing of an agenda entry and that processing some agenda entries was more "costly" (in terms of the amount of effort expanded) than others. Earlier versions of O-Plan did try to maintain knowledge source activation estimates for this reason. Further experimentation is required to resolve this question within O-Plan.

Reference to Results:
No further details of the experiment are provided.

2.11 Experiment 11: Alternative Choice

Type: Validation

Domain Columns Addressed: 3.6.3
The need for the planner to create solutions in the minimum amount of time.

Technology Rows Addressed: 4.8.1. 4.8.2

Assumptions and Background:
The aim of this experiment was to validate a number of different schemes for chosing alternative plan states to backtrack.

When the Planner cannot continue in the plan state it is currently considering, or chooses not to continue, the Agenda Manager (AM) is asked to pick one of the available alternatives so that the planner can continue from there. That is, the planner returns to some earlier decision and makes a different choice. Alternatives represent such decisions as which schema to use when expanding an action or which value to give to a variable.

Since alternatives are data structures, rather than being expressed procedurally in the Planner's code, the AM can employ a number of different search strategies when deciding which alternative to try. This is done, in part, by assigning each alternative a cost. Both the cost function and the AM's choice method can be redefined.

Description of Experiment:
Initially, a very simple cost function for an alternative was used by counting the number of actions in its associated the plan state, and the choice method was to choose the lowest-cost alternative. If more than one alternative had the lowest cost, the most recent one created was taken. (This was done implicitly by the way the list of alternatives was sorted.)

A number of different cost functions and choice methods were tried and evaluated by planning for a range of O-Plan demonstration tasks. The different schemes investigated were as follows:

- Split the cost function into two parts that would be added, one to measure the work done so far, and one to estimate the work still to be done. This is similar to algorithms such as A* [9] and had some benefits in a number of domains.
- Add a depth-first element by choosing the most recently created alternative, rather than the one with lowest cost, in certain cases. This would provide part of the "local best than global best" strategy. We are seeking to use this in O-Plan. (The rest would be provided by knowledge sources that did some local search on their own.) Of the different schemes used this was by far the most effective change of all the ones described.
- Limit the amount of work done before seeing if an alternative might be better. After a given number of O-Plan problem solving cycles, the AM would switch to a better-rated alternative even if it was still possible to continue from the current plan state. This had no significant improvement in performance in the set of test domains and in some cases made things worse.

Summary of Result:
In addition to these schemes a number of different cost functions were tried, stopping once an improvement was found.

Reference to Results:
No further details of the experiment are provided.

3 Summary and Future Experiments

This sections describes a summary of the experiments which have been carried out to date a pointers to a number of future experiments which could be conducted as part of the evaluation framework of the O-Plan project.

The project has undertaken a large number of different kinds of experiments which have been mostly targetted at resolving a short term problem e.g. I need to have four ground transports in the demonstration and the system has severe problem in using four but not three. The aim of future experiments should be gather information through the evaluation matrix and to relate these to the needs of IFDs, TIE work and releases of the O-Plan system. The first step in this process has already been undertaken through a process of relating the experiments conducted on the O-Plan projects (Phases II and III) to the evaluation matrix. The experiments conducted in these phases have been reformatted and described in this report.

The main demonstrations of the O-Plan project in Phase II of the ARPI were conducted at the end of each year of the project, i.e years 1, 2 and 3. These demonstrations were responsible for a large number of experiments but the experiments themselves were not recorded as being part of the results of the demonstration. For example, the second year demonstration concentrated on validating the approach of using a "rich" model of resources in a planner such as O-Plan. While the demonstration was successful in validating the approach i.e. it was a successful validation experiment, the individual experiments which showed which type of resources could be handled and the size of resource pool which could be handled were not recorded. These were successful scalar and comparison experiments and could have been used to populate a number of cells in the evaluation matrix. One of the aims of the future experiments should be to re-examine the three annual demonstrations and identify the additional experiments which were carried out within them. It is often the case that validation experiments can only be conducted by carrying out a number of scalar and comparison experiments.

The design of future experiments should concentrate in the following areas.

- Identifying those areas of the O-Plan system which are acting as "bottleneck" within the planning process. These include:
  - the construction, merging and updating of `or-tree`s continues to be an area of concern. Further experiments are needed to identify the best methods for accomplishing this. The current test domains do not contain many examples of where `or-tree` merging would be of use and new domains need to be identified.
  - the workflow planning aspects of O-Plan need to be examined and experiments constructed which test different control strategies.
- Developing a better understanding between the interactions of different techniques within the O-Plan system. For example, a number of search control technqiues have been identified e.g. branch1/branchN, levels, etc which work successfully in isolation but cause problems when used together.

References

[1] Choueiry, B.Y. and Faltings B., Interactive Resource Allocation by Problem Decomposition and Temporal Abstractions, in *Current Trends in AI Planning*, (eds. Backstrom C. and Sandewall, E.), (the proceedings of the Second European Workshop on Planning), Vadstana, Sweden, 1993.

[2] Collins, G. and Pryor, L. On the Misuse of Filter Conditions: A Critical Analysis, in *Current Trends in AI Planning*, (eds. Backström, C. & Sandewall, E.), IOS Press, 1993.

[3] Currie, K.W. and Tate, A., O-Plan: the Open Planning Architecture, *Artificial Intelligence Journal*, Volume 51, No 1, 1991. Also available as AIAI-TR-67.

[4] Drabble, B. and Tate, A., O-Plan Evaluation Methodology and Experiments, O-Plan Technical Paper ARPA-RL/O-Plan/TR/27 May 1996.

[5] Drabble, B., Tate, A. and Dalton, J., O-Plan Project Evaluation Experiments and Results, O-Plan Technical Paper ARPA-RL/O-Plan/TR/23 September 1995.

[6] Gil, Y., Hoffman, M. and Tate, A., Domain-Specific Criteria to Direct and Evaluate Planning Systems, Proceedings of the ARPA/Rome Laboratory Planning Initiative Workshop, (ed. M.Burstein), Morgan-Kaufmann, 1994.

[7] Joslin, D. and Pollack, M.E., Least-Cost Flaw Repair: A Plan Refinement Strategy for Partial-Order Planning, Proceedings of the National Conference on Artificial Intelligence (AAAI-94), Seattle, WA, USA, 1994.

[8] Ludlow, C., Looking at O-Plan2 and Sipe-2 Through Missionaries and Cannibals, Proceedings of the ARPA/Rome Laboratory Planning Initiative Workshop, (ed. M.Burstein), Morgan-Kaufmann, 1994.

[9] Nilsson, N.J. *Problem Solving Methods in Artificial Intelligence*, McGraw-Hill Book Company, New York, USA, 1971.

[10] Tate, A., Planning and Condition Monitoring in a FMS, Proceedings of the International Conference on Flexible Automation Systems, Institute of Electrical Enginers, London, July, 1984.

[11] Tate, A., Drabble, B. and Dalton, J., The use of Condition Types to Restrict Search in an AI Planner, Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94) Seattle, USA, August 1994.