O-Plan2: Choice Ordering Mechanisms in an AI Planning Architecture [*]

Austin Tate and Brian Drabble
Artificial Intelligence Applications Institute
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
United Kingdom

Abstract

O-Plan2 is an AI planning architecture which supports research into a number of aspects of planning, scheduling and control. It is based on earlier work on the O-Plan System which was directed towards plan generation. The paper explores the different types of choice ordering decisions which need to be made in an architecture for command and control. The mechanisms for choice ordering and selection in the original O-Plan system were found to be too general for efficient use for all purposes. The paper describes a number of choice ordering mechanisms provided in the O-Plan2 Architecture which provide specialised mechanisms more suited to the range of different ordering problems that arise in planning, scheduling and control applications.

1 Introduction

O-Plan is a continuation of earlier work on Nonlin [Tate, 1977] and was influenced by a number of other systems developed in the late '70s and early '80s. In particular it inherits features from:

- NOAH: [Sacerdoti, 1977] by using a *least commitment* search strategy based on a *hierarchical* representation of plans in which actions may be *partially ordered.*
- Nonlin: which introduced the notion of *goal structure* as a means of recording the rationale behind actions in the plan, and also the use of *typed preconditions* as an aid to search space control. A declarative Task Formalism (TF) was also used to provide a description of applications to the planner.
- Deviser: [Vere, 1981] itself derived from Nonlin but was extended to handle *time* and *events.*
- Molgen: [Stefik, 1981] notable for its ability to perform object selection using *least commitment principles.* This is supported by constraint formulation and propagation techniques.
- McDermott: [McDermott, 1978] provided the notion of defining a plan to encompass the decisions on plan structure already taken and outstanding problems still to be handled by the planner.
- OPM: [Hayes-Roth & Hayes Roth, 1979] provided an *opportunistic* planning framework in a *blackboard* architecture. It introduced the concept of *cognitive specialists* which can make certain kinds of decisions to alter the plan as it is being built and showed how measures of the worth of invoking these specialists could be utilised.

O-Plan borrows from these systems, but importantly it presents a framework, or architecture, which enables these techniques to be incorporated into a single system in a uniform way. The system is fully described in [Currie & Tate, 1989].

O-Plan2 is a more portable redesign and reimplementation of the O-Plan architecture in a Common Lisp, X-Windows and Unix environment. It improves on O-Plan in a number of ways. This paper will give an overview of the O-Plan2 architecture and describe the different mechanisms provided within the architecture to enable the planning and control system builder to select suitable implementation methods for describing choices, posting constraints which will restrict choice, postponing choice making decisions until the most opportune time to make them, and triggering choices that are ready to be acted upon.

2 O-Plan2 Architecture

O-Plan2 is a domain independent architecture to support the construction of planning, scheduling and control systems. By providing suitable versions of the Domain Description, Plan State, Knowledge Sources and Support Tools, the architecture can be tailored and made more efficient for specialised use. Three different systems are currently being explored using the same basic architecture.

- An activity based task planner: O-Plan is being retained as the name for the activity based task plan-

[*]This work is supported by the US Air Force/European Office of Aerospace Research and Development by grant number EOARD/88-0044 monitored by Dr Nort Fowler at the Rome Air Development Center. The views expressed are those of the authors only.

ning application of the architecture.

- A scheduler: TOSCA is a variant of the system specialised to manufacturing scheduling applications. Here the plan state includes information on the work capacity of the machines available and resource based representations of the schedules being constructed. Knowledge sources are specialised to resource analysis and resource based planning.
- A planner with a logical temporal representation: a project is underway to employ a temporal logic used for temporal data bases as the basis for the plan state. Specialised Domain Description, Knowledge Sources and Support Tools will allow the planner to generate plans in this representation.

The basic O-Plan2 architecture with definitions of its parts suited to activity based task planning is shown in Figure 1.

The Task Formalism or TF domain description is compiled into static data structures, to be used during the plan generation process - in particular, activities are represented as *schemas*. The left hand side of Figure 1 denotes the *plan state*, which comprises the emerging plan (based on the partial order of activities), the list of plan *flaws*, and internal detail such as the Goal Structure [Tate, 1977], the effects of activities (as in NOAH's Table of Multiple Effects or TOME [Sacerdoti, 1977]) and plan variables. The flaws are posted onto *agenda* lists, which are simply lists of outstanding tasks to be performed during the planning process, and are picked off by an overall *controller* to be processed by the *knowledge sources* in the middle of the diagram. The knowledge sources provided represent the planning knowledge of the system and are referred to as *plan modification operators*. Knowledge sources run on one or more *knowledge source platforms* which are able to run some or all of the available knowledge sources. Knowledge sources in turn may add detail to the plan state, for example by expanding actions to greater levels of detail, establishing how conditions are satisfied, adding ordering links or choosing bindings for plan variables. The knowledge sources may also post new flaws as a result of discovering constraint violations, detecting goal interactions and other problems. The knowledge sources also provide the means whereby a user can assist the planner.

There are a range of flaw types and each is matched with an appropriate knowledge source which can process the particular flaw. Recognised flaw types in the activity based task planner include `expand an action`, `satisfy a condition`, `add a link`, `bind a plan variable` and even `call the user`. This approach allows for the extension of the capabilities of the system. O-Plan knowledge sources relate to plan generation only. The early versions of O-Plan2 will replicate the plan generation features of O-Plan. However, new flaw types and knowledge sources will be provided in O-Plan2 to provide an experimental platform for planning and control in a simplified distributed environment comprising a ground based planner and a space-borne execution agent. The O-Plan2 architecture will support the reasoning of both agents and the extraction and patching in of plans fragments between the on-going control environments of both agents [Tate, 1989].

O-Plan2 is built up in a succession of layers of functionality in order to support the control requirements in a concise manner. At the lowest level is a simple fact storage and retrieval database. This is used to provide support for effect and condition maintenance in a context layered fashion. In turn the effect and condition manager maintains "clouds" of (aggregated) effects and holding periods (ranges) for effects contributing to the satisfaction of necessary conditions in the plan state being developed [Tate, 1986]. Moving up the layers, this is turn provides support for QA (Question Answering) which is the basic reasoning component within the system. QA results drive plan state alterations made by the planner's knowledge sources which in turn are maintained by the net management and time point network manager module. To facilitate re-use of support tools across a range of different specialisations of the O-Plan2 architecture, there is a clear distinction between the plan state specific description (called by us the *associated data structure*) and the underlying management of time points and temporal relationships [Drabble & Kirby, 1991].

O-Plan2 is given tasks by adding entries to its plan state flaw list (agendas). O-Plan2 maintains a *partial plan* at all stages, and makes alterations to the partial plan and the flaw list as it proceeds. The partial plan represents a complete description of a set of possible plans which are only partially specified. The controller is responsible for selecting an outstanding flaw to process whenever a knowledge source can be activated on a waiting knowledge source platform. The domain information is consulted by knowledge sources as they run. This lets knowledge sources access task descriptions, definitions of resources and other domain constraints. The domain information also gives access to the operator schemas which define higher level activities in terms of more detailed activities. There will often be more than one plan modification possible; that is, there will often be a choice of how to remove a flaw. These choices lead to *search*. Normally, the consequences of a decision are maintained by the support tools and information about the selection made is stored as dependency information within the plan state. However, there are occasions on which alternative plan states may need to be generated to explore the options. O-Plan and O-Plan2 allow for such alternative based explorations.

O-Plan searches through a space of partial plans, modifying one plan to obtain another. It seeks a complete plan that is free of flaws - though this may not be achievable in continuous command and control applications. The plans produced by the activity based task planner variant of O-Plan2 are described in networks. The nodes in the network denote actions, and the arcs signify an ordering on action execution. Each node has information associated with it which describes the action's conditions and effects. Time and resource information can also be associated with each action in a plan network node.

3 Ordering Mechanisms in O-Plan

The O-Plan system seeks to include mechanisms to allow for the implementation of an efficient planning system able to take an opportunistic approach to selecting where computational effort should be concentrated during planning. This aim was only partially achieved in the original O-Plan. The basic mechanisms are listed in the following sections.

3.1 Building up information in an Agenda Record

O-Plan included the ability to allow a knowledge source to examine a possible decision point (represented by the agenda entry it was asked to process) and to add information relating to the choice to the fields of the agenda record. If the choice did not become suitably tightly restricted as a result of the addition of this information, it is possible to put the agenda entry back onto the outstanding flaws list with improved information for deciding on the time to reselect it for processing. The ability to build up information around an agenda entry in an incremental way prior to a final knowledge source activation is an important feature that ensures that work done in accessing data bases and checking conditions can be saved as far as possible when processing is halted. There are some similarities to mechanisms within real-time responsive architectures such as RT-1 [Sridharan, 1988].

3.2 Granularity of Knowledge Sources

Each knowledge source within the O-Plan2 architecture encodes a piece of planning knowledge. For example, how to expand an action, bind a variable, check a resource, etc. From a modularity viewpoint, there is some advantage in having a very fine grain of knowledge source to implement planning knowledge. However, this can lead to tens of agenda entries and knowledge source activations with the overheads associated with such activations for even the simplest types of action expansion. In simpler planners, such as Nonlin, an expansion is efficiently handled as an atomic operation. There is a conflicting desire to have efficient large grain knowledge sources implementing planning knowledge and very fine grain knowledge sources detailing each individual step of some higher level plan modification operator.

With a finer grain of knowledge source, it was also found that ordering relationships between agenda entries left in the agenda list had to be stated to ensure efficient processing. The controller was then required to unravel the web of activation orderings that resulted. A special form of agenda entry called a *sequence* was implemented to assist the controller in this task, it would only consider the head of the sequence for activation at any time, subsequently releasing the following agenda items clustered in the sequence in the order indicated. This process is similar to the control blocks used in the Tecknowledge S.1 system [Tecknowledge, 1988].

3.3 Priority of Processing Agenda Entries

O-Plan assigns priorities to every flaw placed on the agendas at the time they are placed. The priorities are calculated from the flaw type, the degree of determinancy of the flaw and information built up in the Agenda Record as described earlier. These provide measures of choice within the flaw. Two heuristic measures are maintained in each agenda entry. One called BRANCH-1 indicates the immediate branching ratio for the choice point. An upper bound on this can be maintained quite straightforwardly. The second measure is called BRANCH-N and gives a heuristic estimate of the number of distinct alternatives that could be generated by a naive and unconstrained generation of all the choices represented by the choice point.

In O-Plan, three agendas are maintained to efficiently select between agenda entries which are ready for knowledge source activation and ones awaiting further information to bind open variables in the agenda information. This is described in [Currie & Tate, 1985]. Eventually though, the ready to run agenda entries are simply rated according to a numerical priority maintained for each agenda entry on the basis of flaw type and the BRANCH-1 and BRANCH-N estimators. This forms too simplistic a measure for allowing the controller to decide between waiting agenda entries. Consideration was given to a rule based controller with knowledge of other *measures of opportunism* but no implementation of this was done within the original O-Plan.

4 Ordering Mechanisms in O-Plan2

O-Plan2 seeks to provide a more coherent set of mechanisms to enable the planning and control system builder to select suitable implementation methods for describing choices, posting constraints which will restrict choice, postponing choice making decisions until the most opportune time to make them, and triggering choices that are ready to be acted upon.

4.1 Knowledge Source Stages

The O-Plan mechanism for building up information in an agenda entry prior to making some selection between alternatives was a very useful feature but proved difficult to use in practice. A knowledge source had to be activated to initiate processing which might simply add a little information to the agenda entries and then suspend to allow the controller to decide whether to progress. This is very inefficient.

In O-Plan2, knowledge sources are defined in a series of *stages*. There can be one or more stages, only the last may make alterations to the plan state (thus locking out other knowledge source final stages which can write to the same portion of the plan state). Any earlier stages may build up information useful to later stages. At the end of any stage, the knowledge source must be prepared to halt processing if asked to by the controller. If it is asked to halt at a stage boundary, the knowledge source may summarise the results of its computation in a field of the agenda record provided for this purpose. A controller directed support routine is called by the knowledge source at the end of each stage to identify whether it must halt or may continue. This allows the controller to dynamically re-direct computation as it considers all the information available to it, while providing a simple and efficient way for the knowledge source to continue computation without intermediate state saving while it continues to receive a go-ahead from the end of stage continuation authorisation routine.

A *Knowledge Source Formalism* for O-Plan2 is being designed to allow for stage definition and to assist with declaring the restrictions on the plan state portions affected by the final plan state modifying stage of the knowledge source - to assist in lock management.

4.2 Knowledge Source Triggers

In O-Plan2, a mechanism of setting *triggers* on agenda entries for activating knowledge sources (and an individual stage of a knowledge source if desired) is provided. The triggers may use various "items" of data available within the plan state and other global information available to the planner. These may include things such as the availability of a specific binding for a plan variable, the satisfaction of a condition at a specific action node in the plan network, the use of a specific resource, the occurrence of an external event, information from the "clock" within the planner, etc. The Knowledge Source Formalism referred to earlier will also be used to define triggers on knowledge source stages. The triggering constructs in the language will initially be quite restrictive to ensure that efficient agenda entry triggering mechanisms can be implemented. However, as we gain experience, we expect the triggering language to be quite comprehensive. A knowledge source may also dynamically create a trigger on a continuation agenda entry when halting processing at a stage boundary.

Only agenda entries which are currently triggered will be available to the controller for decisions on which entries to activate through to a knowledge source running on a knowledge source platform.

4.3 Compound Agenda Entries

Individual *simple* agenda entries can be grouped together into *compound* agenda entries. Only the head entries in the compound agenda entry are considered at any time by the controller (and possibly by the triggering mechanism considered above), thus cutting down on the amount of processing required by the controller to select the next agenda entry to execute when such pre-defined orderings can be specified. Compound agenda entries can be made by knowledge source to implement some definite planning strategy or to implement planning algorithms with finer grain knowledge sources to provide modularity and real time response improvement.

A Support Routine is to be provided in O-Plan2 to allow any knowledge source to easily and reliably build and return a compound agenda entry.

4.4 Controller Priorities

The controller is given the task of deciding which of the current set of triggered agenda entries should be run on an available knowledge source platform. It does this by considering the priority and measures of opportunism of the agenda entry. Four priority levels are available within O-Plan2 - Low, Medium, High and Emergency. The Emergency priority level is only available to handle incoming external events. The RT-1 system has similar priority based processing arrangements [Sridharan, 1988]. In certain cases, an O-Plan2 implementation will possess knowledge source platforms dedicated to processing specific real-time responsive events appearing as agenda entries - thus allowing for reliably real-time response to events categorised as Emergency priority.

A waiting knowledge source platform will be able to run one, several, many or all knowledge sources. Any restriction on a specific platform will be known to the controller. Only triggered agenda entries at the highest priority level which can be processed on a waiting knowledge source are considered by the controller on each cycle. Where there is still choice, a range of *measures of opportunism and priority* are employed to make a selection. The underlying principle is to make a selection according to a strategy given to the controller. Initially this strategy will use user selected preferences or by default will seek to reduce search to the extent it can judge this (reflecting the opportunistic generative planning nature of the early versions of O-Plan2 - like its predecessor O-Plan). Measures such as BRANCH-1 and BRANCH-N described earlier for O-Plan are relevant to this. However, the use of a utility function guided by task specifiers given to the controller will be explored later for O-Plan2 when it is used in continuous command and control applications.

5 Summary

O-Plan2 seeks to provide a more coherent set of mechanisms to enable the planning and control system builder to select suitable implementation methods for controlling the flow and ordering of making choices in an AI planner. These mechanisms are:

- the use of *stages* in knowledge sources to allow for a linear thread of computation to be defined which can be assumed to run through to completion, but provides a means for interruption at defined staging points.
- the definition of *triggers* on knowledge sources and knowledge source stages to provide a clear means to delegate a higher level of knowledge source activation checks to the controller.
- the use of *compound agenda entries* to put direct dependencies of some tasks on others that must complete earlier. This allows complex computational dependencies and strategies to be created.
- the use of *agenda manager priorities* to allow the controller to select appropriate ready-to-run agenda entries and match these to waiting knowledge source platforms.

All the mechanisms described above are part of the O-Plan2 planner now being constructed.

Acknowledgements

My thanks to our colleagues on the O-Plan and O-Plan2 projects: Ken Currie and Richard Kirby.

References

[Currie & Tate, 1985] Currie, K.W. & Tate. A. O-Plan: control in the open planning architecture. *In proc. of the* BCS *Expert Systems '85, Warwick,* UK, *Cambridge University Press, 1985.*

[Currie & Tate, 1989] Currie, K.W. & Tate, A. O-Plan: the Open Planning Architecture. *Submitted to the AI Journal. Also* AIAI-TR-*67. 1989.*

[Drabble & Kirby, 1991] Drabble, B. & Kirby, R.B. Associating AI Planner Entities with an underlying Time Point Network. *Submitted to the European Workshop on Planning, West Germany, March 1991.*

[Hayes-Roth & Hayes Roth, 1979] Hayes-Roth, B. & Hayes-Roth, F. A Cognitive Model of Planning. *Cognitive Science,pp 275 to 310, 1979.*

[McDermott, 1978] McDermott, D.V. A Temporal Logic for Reasoning about Processes and Plans In *Cognitive Science, 6, pp 101-155, 1978.*

[Sacerdoti, 1977] Sacerdoti, E. A structure for plans and behaviours. *Artificial Intelligence series, publ. North Holland, 1977.*

[Sridharan, 1988] Sridharan, N. Practical Planning Systems, *Rochester Planning Workshop,* AFOSR*, 1988.*

[Stefik, 1981] Stefik, M. Planning with constraints. *In Artificial Intelligence, Vol. 16, pp. 111-140. 1981.*

[Tate, 1977] Tate, A. Generating project networks. *In procs.* IJCAI-77*, Cambridge,* USA*, 1977.*

[Tate, 1986] Tate, A. Goal Structure, Holding Periods and "Clouds". *In Reasoning about actions and plans, Morgan-Kaufmann, 1986.*

[Tate, 1989] Tate, A. Coordinating the activities of a planner and an execution agent. *In Procs. of the Second NASA Conference on Space Telerobotics, (eds. G.Rodriguez & H.Seraji), JPL Publication 89-7 Vol. 1 pp. 385-393, Jet Propulsion Laboratory, Pasadena, CA, 1989.*

[Tate, 1990] Tate, A. Interfacing a CAD system to an AI planner. *paper to the* SERC *seminar on integrating knowledge-based and conventional systems, Edinburgh, May 1990. Also Artificial Intelligence Applications Institute* AIAI-TR-76*, 1990.*

[Tecknowledge, 1988] Tecknowledge, S.1 product Description, Tecknowledge Inc., 525 University Avenue, palo Alto, CA 94301. 1988.

[Vere, 1981] Vere, S. Planning in time: windows and durations for activities and goals. IEEE *Transactions on Pattern Analysis and Machine Intelligence Vol. 5, 1981.*

[Wilkins, 1988] Wilkins, D. Practical Planning. *Morgan Kaufman, 1988.*

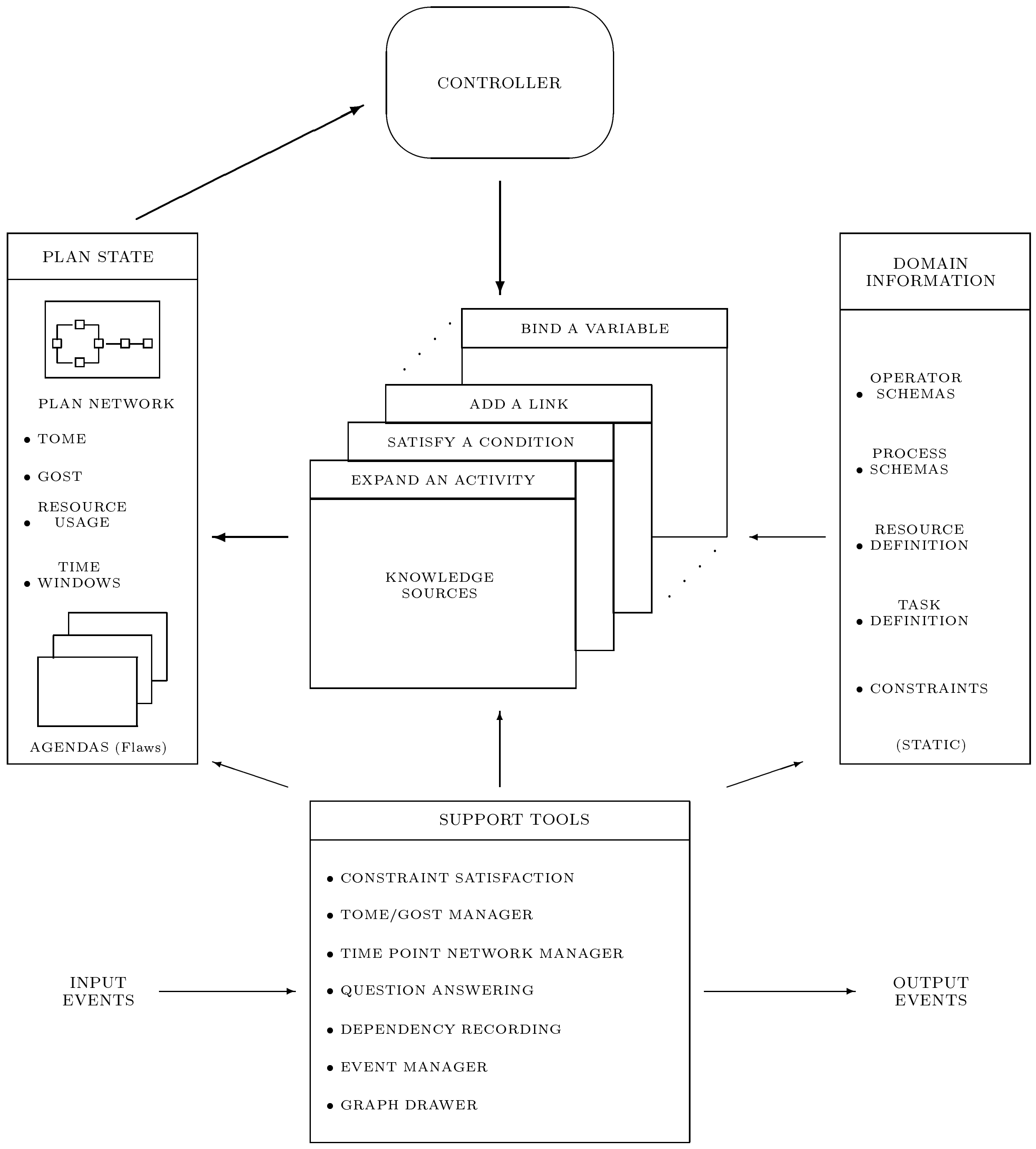

Figure 1: O-Plan2 Architecture