Maintaining Consistency Between Planning Hierarchies: Techniques and Applications

David R. Zoch

Ford Aerospace & Communications Corporation
Space Missions Division
College Park, MD 20740

ABSTRACT

In many planning and scheduling environments, it is desirable to be able to view and manipulate plans at different levels of abstraction, allowing users the option of viewing and manipulating either a very detailed representation of the plan or a high-level more abstract version of the plan. Generating a detailed plan from a more abstract plan requires domain-specific planning/scheduling knowledge; the reverse process of generating a high-level plan from a detailed plan (Reverse Plan Maintenance, or RPM) requires having the system "remember" the actions it took based on its domain-specific knowledge and its reasons for taking those actions.

This paper describes this reverse plan maintenance process as implemented in a specific planning and scheduling tool, The Mission Operations Planning Assistant (MOPA), as well as the applications of RPM to other planning and scheduling problems, emphasizing the knowledge that is needed to maintain the correspondence between the different hierarchical planning levels.

PROBLEM

In many planning and scheduling environments for space applications (e.g., scheduling instrument operations on board a satellite, Space Station Payload Scheduling, etc.) a daily schedule may consist of a complicated mixture of hundreds/thousands of payload activities and operations activities. It is difficult to detect any high level organization or overall plan from this detailed

schedule. These complicated schedules, however, are usually generated from higher level plans, which are valuable to the mission planners since they make it possible to get a more abstract view of the mission objectives and instrument operations for the day.

If schedules remained static, there would be no problem; mission planners have access to both high level plans and detailed schedules. Unfortunately, this is often not the case: schedules change due to changing mission objectives, instrument failures, and targets of opportunity (a sun-observing instrument might want to reposition itself to observe a solar flare, for instance). It is often much simpler to make these changes at the detailed schedule level instead of making the changes to the higher level plan (re-generating the detailed schedule might take hours or days). This obviously causes a difference in the two representations, invalidating the more abstract plan. This, then, is the purpose of Reverse Plan Maintenance: to rectify the difference between the high level plan and the detailed schedule so that the high level plan "agrees" with the modified detailed schedule.

Benefits of Reverse Plan Maintenance

The benefits of RPM in a planning and scheduling application are:

1. A reduction of scheduling mishaps due to discrepancies in plan representations.

2. Immediate feedback on the high-level impacts of making changes to a schedule at a very detailed level.

3. Makes "what-if" scheduling practical, since the entire schedule does not need to be regenerated.

4. Saves computer time and manpower since entire schedules do not need to be regenerated.

An Appropriate Environment for RPM

Reverse Plan Maintenance is not a "necessity" in many planning problems. For instance, if it is only necessary to make changes to a plan at the more abstract planning level and if the detailed schedule generation process is simple, then there is no reason to support RPM. The following criteria are useful in deciding if Reverse Plan Maintenance is appropriate for a specific planning and scheduling problem:

1. A high level (condensed) representation of a plan is desirable and possible, due to some organization (patterns) in the resulting detailed schedule.

2. The detailed schedule resulting from the expansion of the plan is "too complicated," i.e., it is difficult to perceive any organization or overall strategy in the schedule (for instance, it may contain hundreds or thousands of activities, some of which interact with each other).

3. It is desirable to make changes to the more detailed schedule.

4. It is desirable to view the modified schedule in its condensed version so that it can be quickly assimilated by a human planner.

The following pages describe a specific application (for mission planning on the Upper Atmospheric Research Satellite, UARS) where Reverse Plan Maintenance is beneficial and has been implemented.

AN EXAMPLE APPLICATION: MOPA

Upper Atmospheric Research Satellite Background

The Mission Operations Planning Assistant (MOPA) is a knowledge-based system developed by Ford Aerospace for NASA Goddard Space Flight Center to support mission planning for the Upper Atmospheric Research Satellite (UARS), a multi-instrument orbiting observatory scheduled to be launched by the Space Shuttle in 1991. UARS will provide experimenters at remote locations with data on the temperature, composition, and dynamics of the earth's upper atmosphere.

Mission planning for a satellite such as UARS is a complex process since the activities of the ten instruments on board must be defined and coordinated with the actions of the satellite (for instance, most instruments must be put into a safe state whenever the satellite is firing its thrusters.) Additionally, there are "cooperative constraints" between instruments (for instance, two scientists may want to cooperate on an experiment which requires that each has his instrument in a certain mode at the same time) and operational

constraints that if violated could cause damage to an instrument (for instance, aiming a solar/stellar instrument at the sun while it is in one of its star viewing modes).

If the scientists in charge of the instruments on board UARS were all located at Goddard Space Flight Center, these problems could be worked out in daily meetings, as is often currently done for other missions. Since the instrument scientists are located throughout the country, however, this was not feasible, and a more automated approach, i.e., MOPA was taken. An additional motivation for this automated planning approach is that it is applicable to other situations where it is not feasible for the payload scientists to have daily planning meetings, e.g., Space Station.

UARS Mission Planning

The first planning step that the Mission Planning Group (MPG) must do is to create a Daily Science Plan, which is a description of UARS instrument functions to be performed during a day's operation. The DSP is created using the Long Term Science Plan (which is developed by the instrument scientists) as a guideline. After the creation of the "first cut" DSP, the scientists are allowed to review and request modifications to the plan. The MPG acts as a coordinator and negotiator of plan modification requests which will arise due to variances in instrument performance, changing scientific objectives, and the occurrence of targets of opportunity. The MPG must coordinate with the scientists and with the flight operations team to ensure that the Daily Science Plan does not violate current instrument and spacecraft capabilities and constraints.

After the DSP has been approved, the MPG develops detailed operating plans (known as Activity Plans) which are lists of activities for the UARS instruments and their corresponding times of execution. The activities themselves are predefined command sequences which configure an instrument to perform a certain operation.

MOPA Planning Support

MOPA supports the UARS planning process by providing a plan representation analogous to each level of planning in the MPG planning process, and facilities to manipulate these plans. In addition to these three types of plans, the other major data structures in MOPA are ACTIVITIES and EVENTS. The following paragraphs describe each of these

structures.

EVENTS in MOPA are anything that a scientist might want to use in order to trigger his instrument to enter a certain mode. The events may be satellite events (e.g., a yaw maneuver), orbital events (e.g., acquisition of sun), special observational events (e.g., a volcano), or pseudo-events such as a specific time or the beginning of a specific orbit. Figure 1 shows a part of a formatted EVENT file, which contains a list of events for a specific day. (EVENTS are represented and manipulated internally using schema, the data structure provided with the Automated Reasoning Tool (ART) by Inference Corporation. Generic Plans, Daily Science Plans, Activity Plans, and activities are also represented using ART schema. MOPA contains formatting programs to display these to the user in a more readable form. The internal representation used by MOPA is described later in the Knowledge Representation section.)

Start Time	Name	Duration	Priority
1:08	SLEW	:11	4
1:22	SUN	00:37	2
1:36	ORBIT-2	01:36	2
1:38	ASC-NODE	00:49	2
1:53	TDRS-EAST	00:15	2
1:59	NIGHT	01:01	2

Figure 1. Some Sample Events from an EVENT-FILE

ACTIVITIES in MOPA define command sequences that can be executed by an instrument's microprocessor or other hardware in order to perform some arbitrary function. Both Activity Plans and Daily Science Plans consist of sequences of activities, but the activities in each are at different levels of abstraction: the DSP activities are "compound" activities that represent high level instrument operations in terms that the scientists themselves have defined; AP activities are primitive activities that correspond to one instrument command.

There are three types of plans in MOPA: Generic Plans (GP's), which correspond roughly to the notion of a Long Term Science Plan, Daily Science Plans (DSP's), and Activity Plans (AP's). The planning process in MOPA begins with the Generic Plan, which is a high level

event-driven specification of the ACTIVITIES that an instrument is to perform based on the specified EVENT. Figure 2 shows a part of a sample Generic Plan. The part shown is for the SUSIM (Solar Ultraviolet Spectral Irradiance Monitor) instrument. (A complete Generic Plan has a plan for each of the instruments on board.) The plans are generic in that they (1) Describe the operation of each instrument under a wide variety of circumstances, many of which may not occur on a given day, and (2) Are possibly appropriate for many days (or even months), so they can be "reused," probably with some minor modifications, which are made via Reverse Plan Maintenance.

SUSIM plan of operations:

* On every YAW perform:
  1. SAFE-FOR-YAW 2 minutes before YAW until the end of YAW

* On SUN occurrences 2 10 perform:
  1. 10A-SPEC-SCAN for 36 minutes

* On SUN occurrences 4 12 perform:
  1. 1A-SPEC-SCAN for 36 minutes

* On every SUN Perform:
  1. CONTINUOUS-MONITOR for 37 minutes

* On every NIGHT Perform:
  1. ELECTRICAL CALIBRATION for 1 hour 35 minutes.

Figure 2. A Generic Plan for the SUSIM Instrument

The Generic Plan is used in conjunction with a daily EVENT schedule to create a Daily Science Plan for that day. Figure 3 shows part of a DSP. As previously mentioned, the activities in the DSP are actually "compound activities" in that they may represent many actual instrument commands. For instance, the "1A-SPEC-SCAN" activity (1 angstrom spectral scan) might actually consist of opening a viewing door, adjusting the wavelength by rotating a filter wheel, and then closing a door at the end of the viewing period. The instrument scientists are allowed to group arbitrary combinations of activities together in this way and define the resulting "compound" activity for use in Generic Plans (Actually,

"compound" activities can be parts of other compound activities, resulting in arbitrarily deep hierarchies of activities.)

Start Time	Instrument	Name	End Time
19:20	SUSIM	10A-SPEC-SCAN	19:56
19:56	SOLSTICE	SAFE-FOR-SLEW	20:06
19:57	SUSIM	ELECTRICAL-CALI*	20:58
20:06	SOLSTICE	STAR-OBSERVATION	20:22
20:22	SOLSTICE	SAFE-FOR-SLEW	20:28

Figure 3. Some Activities from a DSP

An Activity Plan is created from a DSP using the compound activity definitions. Figure 4 shows some activities from an Activity Plan. The formats of the DSP and AP are identical, the difference is a conceptual one: the DSP contains "compound" activities; the AP contains only "primitive" activities.

Start Time	Instrument	Name	End Time
19:57	SUSIM	ELECTRICAL-CALI*	20:58
20:06	SOLSTICE	OPEN-APERTU-DOOR	20:07
20:07	SOLSTICE	ADJ-WAVE-LENGTH	20:21
20:22	SOLSTICE	CLOSE-APER-DOOR	20:23
20:23	SOLSTICE	SOLSTICE-WAIT	20:28

Figure 4. Some Activities from an Activity Plan

MOPA provides a menu driven interface which makes it easy for the user to retrieve and save Generic Plans, create DSP's from a GP, and create AP's from the DSP. Additionally, MOPA provides a graphical representation (activity time lines) of both DSP's and AP's (Figure 5 shows the graphical representation of a DSP). The activities in the time lines are mouse-sensitive, which allows the user to replace, add, and delete activities simply by clicking the mouse on the appropriate activity and choosing the appropriate operation.

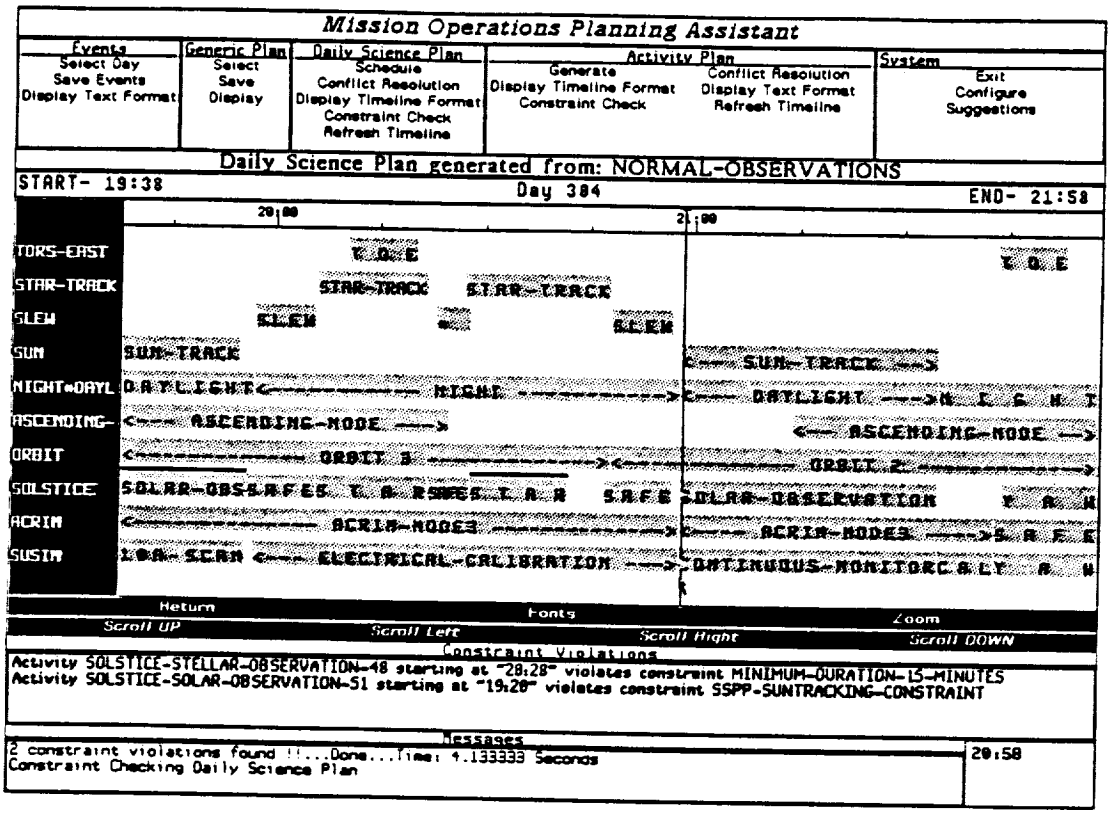

Figure 5. Graphic Representation of a DSP

KNOWLEDGE REPRESENTATION

Before tackling the internals of Reverse Plan Maintenance, it is important to understand the internal representations of events, activities, and generic plans. This section focuses on the representation of these entities and how the knowledge is used for generating Daily Science Plans and Activity Plans.

Events

As previously mentioned, the process of planning and scheduling instrument activities is driven by the nature and time of occurrence of different events. For example, the period of time during an orbit that the Sun is visible (spacecraft day) is an event which triggers the sun-viewing activities of several instruments on the UARS.

The event classes used in MOPA are represented in a classification hierarchy. At the root of this hierarchy

is the definition of the generic event as shown in Figure 6. The attributes of the generic event schema are inherited by the rest of the event classes.

Generic Event schema

```
(defschema event
  (name)
  (short-name)
  (has-activity)
  (priority)
  (start-time)
  (end-time)
  (duration))
```

Figure 6

The attributes name, short-name, priority, start-time, end-time, and duration are common to events and activities. The name and short-name attributes provide a printed representation of the event for use by the MOPA time line and textual displays. Values of these attributes may be string or symbolic types. The start-time, end-time, and duration attributes will contain the scheduled times (in numbers of seconds offset from the start of day) of the instances of the events. The has-activity relation relates an event schema to the activities which are triggered by it. This link is used during the RPM process and its values are activity schemata.

Activities

Central to the design of MOPA is the representation of UARS instrument activities. Activities are represented in MOPA as collections of ART schemata. This section describes the details of the activity representation scheme.

Basic Attributes-- All activity schemata have certain basic attributes associated with them. The generic activity schema is shown in Figure 7. The attributes name, short-name, priority, start-time, end-time, and duration have the same meaning as they do for events. There are several relations defined for activities that link them to other schema types. The activity-of relation provides a link between an activity and the instrument with which it is associated. The priority slot is used to resolve conflicts between activities that are scheduled to occur at the same time. The priority slot gets its value

from the event that triggered it. This has the effect of giving priority to activities that occur as a result of higher priority events. For example, a GP might specify for an instrument to respond to both a spacecraft yaw maneuver and the acquisition of sun. Since the instrument might be damaged if it does not put itself into a safe mode during the yaw maneuver, the YAW event has been assigned a higher priority than "acquisition of sun" and any activity triggered by the YAW event will have priority over any activity triggered by the "acquisition of sun" event.

Generic Activity Schema

```
(defschema activity
  (name)
  (short-name)
  (has-event)
  (activity-of)
  (priority 0)
  (start-time 0)
  (duration 0)
  (end-time 0)
  (has-constraint-violation)
  (has-sub-activity)
  (sub-activity-of)
  (fixed-duration)
  (has-pre-condition)
  (delta-time 0)
  (restriction))
```

Figure 7

Key Slots for RPM-- The has-event relation relates an activity to the event instance that caused its selection and scheduling. The restriction attribute value is passed down from the Generic Plan entry that specified the activity. This value is the "event restriction" value (EVERY, NUMBERS, or TIME) from the DSP entry. The value of the restriction attribute is used during the RPM process to determine the "triggering-event-specification" (refer to the BNF in Figure 8) in the Generic Plan that caused the activity to be scheduled. The has-sub-activity relation links an activity to its detailed components; the sub-activity-of relation provides a link in the opposite direction. These links are used when doing RPM from the AP to the DSP.

The delta-time attribute determines when an activity should start in relation to the event which triggered it.

This value is a negative or positive integer specifying the number of seconds before or after the event start-time that the activity should begin.

Activities also have several other slots; these are not described here since they are not important for the RPM process.

Generic Plans

The Generic Plan (GP) is an encoded description of the relationship between Events and planned instrument activities to occur based upon the Events (e.g.., during Spacecraft Night, Susim should perform Electrical Calibration). This event-driven specification of the daily operations of an instrument is encoded as an ART schemata.

Figure 9 illustrates a sample plan schema for the Susim instrument. The plan-of relation links the instrument-plan to the instrument which executes the plan. The triggering-event slots determine the activity that Susim will perform when the specified event occurs.

Triggering-event Syntax

```
<triggering-event-specification> :=
      triggering-event ( <event-name>
                         <event-restriction>
                        (<activity-specification>*)
<event-restriction> := EVERY |
                       <number-specification> |
                       <time-specification>
<number-specification> := (NUMBERS <numbers>+)
<time-specification> := (TIME <time>+)
<activity-specification> :=
          ( <activity-name> <attribute-value-pair>*
            [DURATION <duration-specification>]
<attribute-value-pair> := <activity-attribute-name>
                          <attribute-value>
<duration-specification> := <time-specification> |
                            EVENT | END-EVENT
```

Figure 8

For example, the "triggering-event (night-event ..." entry is interpreted to mean "the Susim instrument should perform a SUSIM-ELECTRICAL-CALIBRATION activity at every occurrence of the spacecraft NIGHT-EVENT". The "triggering-event (sun-track (numbers 4 12)" means that on the fourth and twelfth occurrences of the SUN-TRACK event,

Susim should perform the SUSIM-1A-SPECTRAL-SCAN activity. A "numbers" event-restriction entry overrides an "every" event-restriction entry. If both of these types of event-restriction are present (as is the case in the Susim plan) the "every" specification is interpreted to mean "on every event, except the occurrences specifically mentioned, perform ...".

The triggering-event entry for SPACECRAFT-YAW-EVENT in the Susim plan illustrates attribute/value pairs which can be used to override the default activity slot values for the activities created during the DSP generation process. In this case the delta-time attribute of the SUSIM-MAN-PREP activity is assigned the value -120 (i.e. 2 minutes before). The duration is specified to be "end-event" meaning the SUSIM-MAN-PREP activity should continue until the end of the SPACECRAFT-YAW-EVENT (a numeric duration could have been specified instead).

Susim Plan schema

```
(defschema susim-plan
  (instance-of instrument-plan)
  (plan-of susim)
  (triggering-event
    (night-event every
((susim-electrical-calibration))))
  (triggering-event
    (sun-track every
((susim-continuous-monitor))))
  (triggering-event
    (sun-track (numbers 4 12)
((susim-1a-spectral-scan))))
  (triggering-event
    (sun-track (numbers 2 10)
((susim-10a-spectral-scan))))
  (triggering-event
    (spacecraft-yaw-event every
      ((susim-man-prep delta-time -120
                       duration end-event)))))
```

Figure 9

REVERSE PLAN MAINTENANCE IN MOPA

As previously mentioned, RPM allows the user to edit the graphic displays provided by MOPA at either the activity plan level or the DSP level and "reflect" these changes back to the high level plan for the day, the

generic plan. This plan can then be saved for later use. This saves the user from having to edit the generic plan and regenerate both the DSP and the AP, allowing him to make small changes in a number of seconds and still have three consistent representations.

Currently, the MOPA prototype supports RPM only for what we think is the most common type of modification a user will make to a plan: replacing one activity with another. Activity Replacement with RPM is supported for activities at both the DSP and AP levels. Other types of RPM's such as activity deletions, additions, and modifications (such as increasing the duration of an activity) are not currently handled, but should be for an operational system.

An Example

Figure 10 shows a part of a generic plan for the ACRIM instrument. According to the plan, ACRIM should be in MODE-3 for every SUN-TRACK event except for the tenth one; in this case it will be in MODE-7.

```
(defschema acrim-plan " "
   ...
 (triggering-event
   (sun-track every ((acrim-mode3 duration 8000))))
 (triggering-event
   (sun-track (numbers 10) ((acrim-mode7 duration 8000))))
    ...)
```

Figure 10. A Part of a plan for the ACRIM instrument.

Suppose that the user has decided that he does not want ACRIM to be in MODE-3 during the second SUN-TRACK event; instead, he just wants to put the ACRIM instrument in a wait mode. To make this change, he simply brings up the graphic display of activities (either the DSP or AP), "clicks" the mouse on the ACRIM-MODE-3 that he wants to replace, and then "clicks" the mouse on the replacement activity, ACRIM-WAIT (a menu of possible replacement activities will appear when the user clicks on the "replace" option). The MODE-3 activity will be replaced in the graphics display by a WAIT activity of the same duration. As part of the replacement operation, the current plan is modified as shown in Figure 11.

```
(defschema acrim-plan
          " "
        ...
  (triggering-event
    (sun-track every ((acrim-mode3 duration 8000))))
  (triggering-event
    (sun-track (numbers 10) ((acrim-mode7 duration 8000))))
  (TRIGGERING-EVENT (SUN-TRACK (NUMBERS 2)
     ((ACRIM-WAIT DURATION 1000)))))
```

Figure 11. ACRIM plan after RPM.

This plan says to do something special, i.e., put ACRIM in the WAIT mode, on the second occurrence of SUN-TRACK (Changes made because of RPM are capitalized). If this plan is then saved, it can be used to precisely regenerate the current activities for the current day. Additionally, this might be an appropriate plan to use on other days.

Special Knowledge Needed for Reverse Translation

In order to be able to modify the generic plan based on changes made at other levels, it is necessary to know what part of the generic plan caused the creation of an activity. For instance, if an activity such as ACRIM-WAIT-1 is replaced with ACRIM-MODE-4, it is important to know that the line:

```
(triggering-event
   (sun-track (numbers 3) ((acrim-wait duration 1000))))
```

caused the generation of the WAIT activity, since this is the line that must be altered. The necessary information is the instance of the event that triggered the creation of the activity (stored on the HAS-EVENT slot), and the restriction from the generic plan (the RESTRICTION slot). These two "attributes" are assigned to each activity when it is created so that reverse translation will be possible.

Reverse Translation at the DSP level

The main parts of reverse translation when a change has been made at the DSP level are:

1. Find the entry in the generic plan that caused the replaced activity to be generated and determine whether to modify this entry, add a new entry, or both.

2. Figure out the slots of the new activity that should be included in the plan (i.e., don't explicitly include an attribute in the Generic Plan if its value is the same as the default value.

The following sections explain each of these in more detail.

Finding the Generic Plan Entry to Modify-- Since each activity knows the exact occurrence of the event that caused it to be generated (the HAS-EVENT slot) it is straightforward to find all of the entries that could have possibly caused the creation of the current activity. The three possible cases with their resolution strategies are:

1. There is only one entry with the proper triggering-event, and it has an EVERY restriction. (In this case, an extra entry is added to the generic plan; the extra entry will specify what to do in the special case and will have a restriction slot such as "(numbers 8)".)

2. An activity with a restriction slot with a value such as "(numbers 4)" is being modified; in this case, this entry is just modified to reflect the new activity. (There might also be an entry for the same triggering event with an EVERY restriction. This entry can be ignored since the more specific restriction has precedence.)

3. An activity is being replaced that has a restriction slot such as "(numbers 4 10 12)". In this case, this entry must be modified to something such as "(numbers 4 10)" (assuming that the activity that happens on occurrence "12" is being replaced), and a new entry with a restriction slot of "(numbers 12)" is added with the new activity.

Determining the Attributes of the new activity-- The actual attributes of the new activity (its duration, start time, priority, triggering event, etc.) are known, since they are identical to the attributes of the activity that it replaced; the only difference is the actual activity. The purpose of this part of the algorithm is to (1) minimize the amount of information that is copied back into the plan (it is not necessary to specify information that is the same as the default value), and (2) to specify the duration of a replacement event as "END-EVENT" or "EVENT," when appropriate, instead of just using a numeric value.

Reverse Translation at the Activity Plan level

There are several possible situations when doing replacement at the AP level. A user might replace an activity that is part of a known hierarchy with another activity, thus destroying the hierarchy; alternatively, he might replace an activity in a hierarchy with another one that causes a new hierarchy to exist. For instance, if A/B/C is a known hierarchy called A1, and A/X/C is a known hierarchy called AX, then replacing B with X causes AX to replace A1 at the Generic plan level. If A/X/C were not a known hierarchy, then A1 is replaced with the list of activities, A/X/C. A third possibility in doing replacements at the AP level is that an activity that is not a part of any hierarchy is being replaced: in this case, the processing is identical to that for processing a replacement at the DSP level.

OTHER APPLICATIONS OF RPM

Many AI systems are said to be "intelligent" systems or "expert" systems since they use the intelligence of an expert (often in the form of rules) to make the same decisions that an expert would under the same conditions. RPM is a step towards "intelligent" systems in a different sense: the system "knows" and "remembers" why it has done certain things and can use this knowledge for other purposes. Many rule-based systems can perform a back trace of rules fired in order to let the user know how it arrived at a certain conclusion. The concept in RPM is to have the system itself effectively use this knowledge to accomplish other goals; in MOPA, the knowledge is used to allow fast incremental changes to a schedule; other applications are also possible.

For example, in a scheduling system that consists of a number of activities competing for limited resources, a scheduler might remember that it could not schedule activity A2 because it conflicted with a higher priority activity, A1, which consumed the remaining resources available at that time. If A1 is later moved or deleted, the scheduler could instantly "know" that A2 can now be scheduled. Other activity-to-activity dependencies can similarly be remembered and used to accomplish quick rescheduling/replanning.

Missing Knowledge

In many cases it is just not possible to do the "inverse" of an operation as RPM does in MOPA; for instance, if an activity is added in MOPA, there is no way

for the system to "know" which activity "causes" the execution of this new activity. The system could certainly make some intelligent guesses, by looking at the events that usually trigger a certain type of activity, but it can never be sure. Prompting the user (or having the user validate the system's result) is probably the most reasonable solution in these cases.

SUMMARY AND CONCLUSIONS

The Reverse Plan Maintenance feature of MOPA is a time-saving feature that is implemented for the case of replacing one activity with another at either the DSP or AP levels. Other types of plan modifications are possible also, even though the user might have to be prompted for additional information.

The reverse plan maintenance process is an exemplary application to show what can be done with the knowledge that is typically available in a knowledge-based system. By remembering its decision paths and storing the information in a usable form, several seemingly difficult tasks can be easily accomplished.

Intelligent systems that use available knowledge can often exhibit their "intelligence" by not doing redundant re-computations; instead they can "know" what needs to be re-done and do only those calculations.

REFERENCES

1. M. Fox, "Constraint Directed Search: A Case Study of Job-Shop Scheduling". Ph.D. Thesis.,Computer Science Dept, Carnegie Mellon University, Pittsburg, PA 15213, 1983.

2. W. Gevarter, "Artificial Intelligence". Noyes Publications, 1984.

3. I. Goldstein and B. Roberts. "NUDGE, A Knowledge-based Scheduling System". The Fifth Internation Joint Conference on Artificial Intelligence, IJCAI, 1977.

4. Inference Corporation, "ART Reference Manual Version 2.0". Inference Corporation, 1986.

5. J. King, "RPMS: Resource Planning and Management System". JAI PCC 1984.

6. Symbolics Inc. "Reference Guide to Symbolics-Lisp". Symbolics Inc. 1985.

7. J. G. Schuetzle and D. R. Zoch, A Hierarchical Planning System For Mission Support. Expert Systems in Government Symposium, 1986

8. J. G. Schuetzle, The Mission Operations Planning Assistant, 1987 Goddard Conference on Space Applications of Artificial Intelligence and Robotics.