Automated Generation of Antenna Tracking Plans: A Knowledge-Based Approach

S. Chien, A. Govindjee, and F. Fisher
Information and Computing Technologies Research Section

T. Estlin
University of Texas, Austin

X. Wang
Rockwell Science Center, Palo Alto, California

R. Hill, Jr.
University of Southern California, Information Sciences Institute, Marina Del Rey, California

This article describes the Deep Space Network Antenna Operations Planner (DPLAN)—a system that automatically generates tracking plans for operations of DSN antennas. DPLAN accepts as input the current equipment configuration information and a set of requested antenna track services. The system then uses a knowledge base of antenna operations procedures to produce a plan of activities that will provide the requested services using the allocated equipment. DPLAN produces this plan using an integration of artificial intelligence (AI) techniques, specifically, hierarchical task network (HTN) and operator-based planning. In this article, we describe the antenna automation problem, the DPLAN system for automatic generation of track plans, DPLAN's current deployment status, and planned future work.

I. Introduction

Each day, at sites around the world, NASA's Deep Space Network (DSN) [5] antennas and subsystems are used to perform scores of tracks that support Earth-orbiting and deep-space missions. Due to the complexity of this equipment, the large set of communications services (in the tens), and the large number of supported equipment configurations (in the hundreds), correctly and efficiently operating this equipment to fulfill tracking goals is a daunting task. An additional requirement is that the antenna operations knowledge embodied in the system must be easily understandable and maintainable. This requirement also must be met as equipment upgrades, services, protocols, and software changes evolve.

The Deep Space Network Antenna Operations Planner (DPLAN) is an automated planning system developed by the Jet Propulsion Laboratory (JPL) to automatically generate antenna tracking plans that satisfy DSN service requests. In order to generate these antenna operations plans, DPLAN uses a number of information sources, including the project-generated service request, the spacecraft sequence of events, the track equipment allocation, and an antenna operations knowledge base. The project service request represents the basic communications services requested during the track (telemetry/downlink,

commanding/uplink, ranging (uplink and downlink), etc.). The project sequence of events indicates the relevant spacecraft mode changes (such as transmission bit-rate changes, modulation index changes, etc.). The equipment allocation dictates the antenna and subsystem configuration available for the track. The antenna operations knowledge base provides necessary information on the requirements of antenna operation actions. In particular, this information dictates how these actions can be combined to provide essential communications services.

The DPLAN system is one element of a far-reaching effort to upgrade and automate DSN operations in order to increase reliability and reduce operations costs for the DSN. A prototype of the DPLAN system was demonstrated successfully in February 1995 at the experimental DSN station, DSS 13 [10,11], on a series of Voyager tracks. Efforts currently are under way to insert the technologies used in this demonstration into the operational DSN.

This article begins by providing an overview of current DSN operations. Next we describe an architecture for automating DSN operations. Specifically, we give a functional description of each of the components, which include the Demand Access Network Scheduler (DANS) system for automated resource allocation [4], DPLAN [12],[1] an automated procedure generation system, and a plan execution and monitoring system (called NMC). In addition, we provide examples of the inputs and outputs to each of these components to illustrate what occurs at each step of DSN operations. Next, we describe the DPLAN system, including (1) the track plan generation problem, (2) an overview of artificial intelligence hierarchical task network (HTN) and operator-based planning, (3) the DPLAN system, and (4) an example of operation. Finally, we describe current efforts to deploy the DPLAN system in the operational DSN and other areas of current work.

II. How the DSN Operates

The DSN track process occurs daily for dozens of different NASA spacecraft and projects, which use the DSN to capture spacecraft data. Though the process of sending signals from a spacecraft to Earth is conceptually simple, in reality there are many earthside challenges that must be addressed before a spacecraft's signal is acquired and successfully transformed into useful information. In the remainder of this section, we outline some of the steps involved in providing tracking services.

A. Network Preparation at the Network Operations Control Center

Figure 1 provides a simplified depiction of DSN operations (see [2,5] for a more complete description of the DSN processes).[2] The first stage is called network preparation, and it occurs at the Network Operations Control Center located at JPL. The entire process is initiated when a flight project sends a request for the DSN to track a spacecraft. This request specifies the timing constraints of the track (e.g., when the spacecraft can be tracked), data rates, and frequencies required, as well as the services required (i.e., downlink of information from the spacecraft, commanding uplink to the spacecraft, etc.). The DSN responds to the request by performing a process called network preparation. The network preparation process includes attempting to schedule the resources (i.e., an antenna and other required subsystems, such as receivers, exciters, and telemetry processors) needed for the track as well as generating necessary data products required to perform the track (predictions of the spacecraft location relative to the ground station, transmission frequencies, etc.). The output of this process is a schedule of tracks to be performed by DSN ground stations, equipment allocations to tracks, and supporting data required for tracks. One key part of these supporting data is the sequence of events (SOE) describing the time-ordered

---

[1] T. Estlin, X. Wang, A. Govindjee, and S. Chien, *DPLAN Deep Space Network Antenna Operations Planner Programmers Guide Version 1.0*, JPL D-13377 (internal document), Jet Propulsion Laboratory, Pasadena, California, February 1996.

[2] See also *Final Report of the Services Fulfillment Reengineering Team*, JPL Interoffice Memorandum RJA 95-008 (internal document), Jet Propulsion Laboratory, Pasadena, California, March 14, 1995.
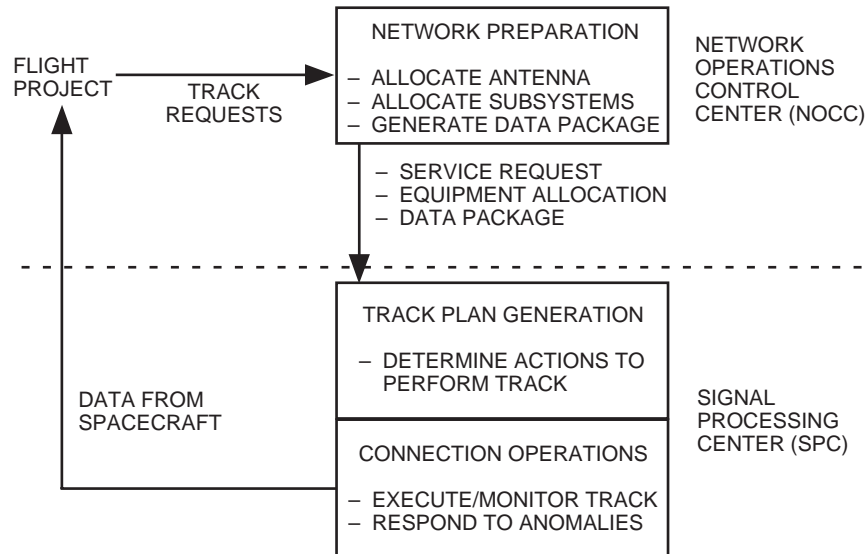
```
                        ┌──────────────────────────────┐
                        │  NETWORK PREPARATION         │   NETWORK
  FLIGHT    ───────────▶│                              │   OPERATIONS
  PROJECT      TRACK     │  – ALLOCATE ANTENNA          │   CONTROL
               REQUESTS  │  – ALLOCATE SUBSYSTEMS       │   CENTER (NOCC)
                        │  – GENERATE DATA PACKAGE     │
                        └──────────────────────────────┘

                              – SERVICE REQUEST
                              – EQUIPMENT ALLOCATION
                              – DATA PACKAGE

        ┌──────────────────────────────┐
                        │  TRACK PLAN GENERATION       │
                        │                              │
  DATA FROM             │  – DETERMINE ACTIONS TO      │   SIGNAL
  SPACECRAFT            │    PERFORM TRACK             │   PROCESSING
                        ├──────────────────────────────┤   CENTER (SPC)
                        │  CONNECTION OPERATIONS       │
                        │                              │
                        │  – EXECUTE/MONITOR TRACK     │
                        │  – RESPOND TO ANOMALIES      │
                        └──────────────────────────────┘
```

Fig 1. An overview of DSN operations.

activities that should occur during the track. The SOE includes actions that the DSN should take (e.g., begin tracking the project's spacecraft at 1200 hours), and it also includes events that will occur on the spacecraft being tracked (e.g., the spacecraft will change frequency or mode at a designated time and the DSN should anticipate the event). Additionally, the DSN must generate predict information required for the track. This is information about where in the sky the spacecraft will be relative to the antenna so that the antenna can be directed to the correct orientation to acquire the spacecraft and to maintain pointing during the track as the Earth rotates and moves and the spacecraft moves.

B. Data Capture at the Signal Processing Center

Once the schedule has been determined and the SOE and predict information generated, the DSN operations process moves to the Signal Processing Centers (SPCs),[3] where a process called data capture occurs. The data capture process is performed by operations personnel at the deep-space station. First they determine the correct operations necessary to perform the track. Then they perform these actions— configure the equipment for the track, establish the communications link, which we hereafter refer to as a "link," and then perform the track by issuing control commands to the various subsystems comprising the link. Throughout the track, the operators continually monitor the status of the link and handle exceptions as they occur. For example, the ground station may lose the spacecraft signal (this occurrence is called "the receiver breaking lock with the spacecraft"). In this case, the operations personnel must take immediate action to reacquire the spacecraft signal as quickly as possible to minimize the amount of data lost. All of these actions currently are performed by human operators, who manually issue tens or hundreds of commands via a computer keyboard to the link subsystems. The monitoring activities require

---

[3] To explain in further detail, in the operational DSN, Deep Space Stations (DSSs) are organized into complexes where several DSSs share a pool of common subsystems. These complexes are called Signal Processing Centers (SPCs). However, the prototyping work described in this article took place at the DSN's research station, DSS 13. DSS 13 does not share subsystems with other DSSs because its equipment tends to be different (e.g., experimental or test versions) and, hence, does not belong to an SPC. Thus, in the operational DSN, the track plan generation and connection operations efforts would be at an SPC, but for this work, they took place at a DSS. From an artificial intelligence (AI) standpoint, the reader can assume that SPCs and DSSs are interchangeable.

the operator to track the state of each of the subsystems in the link (usually three to five subsystems), where each subsystem has many different state variables that change over time.

C. Automation of DSN Processes and the DSN Planner

We have just described the current process for transforming a flight project service request into an executable set of DSN operations. As we have already pointed out, many of the steps of this process are intensely manual. As part of NASA's goals of reducing costs and increasing service, the DSN technology program has been working on technology development, demonstration, and deployment of a series of systems to automate portions of these tasks. The Demand Access Network Scheduler (DANS) [4] is being developed to automate the network resource scheduling process. DANS is designed to work in close coordination with the Network Planning and Preparation (NPP) system (NPP tracks resource usage but does not automatically schedule or reschedule). As already mentioned, the DSN Antenna Operations Planner (DPLAN) [3] (see also [2]) is being developed to automatically generate DSN track plans. Finally, the Network Monitor and Control (NMC) system will automate the connection operations, including intelligent task control, execution monitoring, and exception handling at the DSS sites. This article focuses on the DPLAN system; DANS [4] and NMC are described in greater detail elsewhere.

III. Track Plan Generation: The Problem

Generating a track procedure involves taking a general service request (such as telemetry—the downlink of data from a spacecraft) and an actual equipment assignment (which describes the type of antenna, receiver, telemetry processor, etc.), and then generating a partially ordered sequence of commands called a temporal dependency network (TDN). This command sequence will create/configure a communications link that enables the appropriate interaction with the spacecraft. DPLAN uses an integration of an artificial intelligence (AI) hierarchical task network (HTN) and operator-based planning techniques to represent DSN antenna operations knowledge and to generate antenna operations procedures on demand.

DPLAN uses high-level track information to determine the appropriate steps, ordering constraints on and parameters of these steps, that will achieve the high-level track goals given the equipment allocation. In generating the TDN, the planner uses information from several sources (see Fig. 2).

(1) *Service Request and Project SOE.* The service request specifies the DSN services requested by the project and corresponds to the goals or purpose of the track. The project SOE details spacecraft events occurring during the track, including the timing of the beginning and ending of the track and spacecraft data transmission bit-rate changes, modulation index changes, and carrier and subcarrier frequency changes.

(2) *Project Profile.* This file specifies project-specific information regarding frequencies and pass types. For example, the project SOE might specify the frequency as high, and the project profile would specify the exact frequency used. The project profile might also dictate other signal parameters and default track types.

(3) *Temporal Dependency Network (TDN) Knowledge Base (KB).* The TDN KB [7,8] stores information on the TDN blocks available for use by the DSN Planner and the Link Monitor and Control Operator Assistant (LMCOA). This knowledge base includes information on preconditions, postconditions, directives, and other aspects of the TDN blocks.

(4) *Equipment Configuration.* This configuration details the types of equipment available and specifies unique identifiers for each piece of equipment that is to be used in the track. Equipment includes items such as the antenna, antenna controller, receiver, etc.
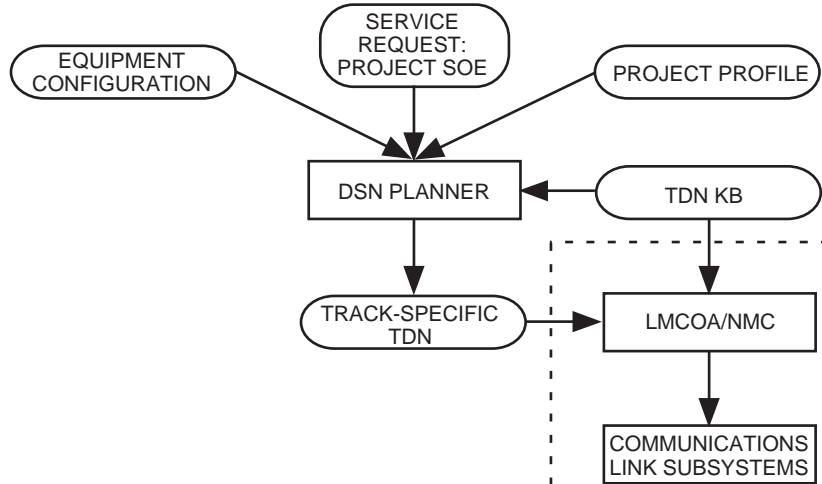
Fig. 2. DPLAN and LMCOA/NMC inputs and outputs.

IV. Artificial Intelligence Planning Techniques

AI planning researchers have developed numerous approaches to the task of correct and efficient planning. Two main planning methods are operator-based planners[4] [13] and hierarchical task network (HTN) planners [6]. DPLAN uses a combination of both these approaches, exploiting the advantages of each.

An HTN planner [6] uses task reduction rules to decompose abstract goals into lower-level tasks. HTN planners can encode many different types of information into task reductions. By defining or not defining certain reduction refinements, the designer can direct the planner towards particular search paths in certain contexts. The user also can directly influence the planner by explicitly adding an ordering constraint or goal protection that would not strictly be derived from goal interaction analyses. Search-control knowledge also can be encoded by writing predefined plan segments (i.e., action sequences) to achieve certain goals, thereby avoiding considerable search.

In contrast, an operator-based planner [13,1] reasons at a single level of abstraction—the lowest level. Actions are strictly defined in terms of preconditions and effects. Plans are produced through subgoaling and goal interaction analyses. In this framework, all plan constraints (protections, ordering, and codesignation) are a direct consequence of goal achievement and action precondition and effect analysis. Thus, an operator-based planner generally has a strict semantics grounded in explicit state representation, i.e., defining what is and is not true in a particular state (or partial state).

DPLAN combines these two planning methods, utilizing the advantages of each. For instance, an operator-based planner requires a very rigid representation, which is both a strength and a weakness. It is an advantage in that there usually is one obvious method of encoding each subproblem. However, this rigidity also can make certain aspects of a problem difficult to represent. Known ordering constraints and operator sequences can be difficult to encode if they cannot easily be represented in terms of preconditions and effects. Such constraints can and are often forced by adding "dummy" preconditions in which an operator A is made to precede an operator B by forcing A to achieve a condition C for B. However, this solution often can create a misleading representation in that other occurrences of A do not require C to be true. An HTN planner, on the other hand, allows the easy representation of known ordering constraints.

---

[4] The term operator-based planning is used to describe planning algorithms that use a set of actions (also called operators) to construct plans. These actions usually are specified in terms of preconditions and effects.

Domain information, such as constraints, is added easily to domain rules in the HTN framework. This type of representation allows the user easily to direct the planner's search by explicitly defining items such as ordering constraints and goal protections.

By using a combination of both HTN planning and operator-based planning, we easily can direct a search and can define knowledge in an understandable top-down fashion. In a hybrid representation, we also have the ability to define knowledge in the more structured operator-based fashion when appropriate.

DPLAN's algorithm is a combination of both hierarchical task network (HTN) planning techniques and operator-based planning techniques. In HTN planning, abstract actions such as "calibrate receiver" or "configure sequential ranging assembly" (SRA) are decomposed into specific directives for specific hardware types. In operator-based planning, requirements of specific actions, such as "move antenna to point," are satisfied using means–end analysis, which matches action preconditions to effects and resolves any ordering constraints that occur.

V. The DPLAN Planning Algorithm

The DPLAN planning algorithm uses a unique combination of the HTN and operator-based planning techniques, as discussed above. DPLAN operates by refining a set of input top-level goals into a set of low-level operational goals. Plans are represented by a three-tuple: $< U, C, S >$ where U is a set of nonoperational (or high-level) goals, C is a set of constraints, and S is a set of operational goals. At the end of planning, U should be empty, all constraints in C should be consistent, and the goals in S are returned as the final plan steps.

An overview of the DPLAN algorithm is shown in Fig. 3. The main inputs to DPLAN are a set of high-level goals, G; a set of decomposition rules, R; and the set of all possible operational goals, O. Search is implemented by keeping a queue of partial plans to be explored. Currently, plans are selected from the queue using a best-first heuristic; however, other search techniques easily could be employed. Steps 1 and 2 of the main loop remove the best plan off the queue, and Step 3 checks if that plan is a solution. If no solution has been found, then a new goal is selected for refinement in Step 4. Step 5 chooses a refinement strategy for that goal, and in Step 6, any new plans created through that strategy are inserted into the plan queue.

A plan is considered a solution if two conditions are true. The first is that there are no nonoperational goals left to be refined. The second condition is that all context goals have been achieved or are directly achievable in the current plan. Context goals are goals that were needed for applying a decomposition

Algorithm DPLAN (*G*,*R*,*O*)

Initialize the plan queue *Q*:= (<*G*,{}, {}>).

While *Q* is not empty and the resource bound has not been exceeded,

1. Select a promising plan *P* in *Q* using heuristics.

2. Remove *P* from *Q*.

3. If *P* contains only operational goals, then check context goals in *P*. If the context goals are achieved, return *P*. Otherwise go to 1.

4. Choose a nonoperational goal, **g**, from *U*.

5. Refine **g** by applying either a decomposition rule or simple establishment. Update the sets *U*, *C*, and *S* accordingly.

6. Insert any new plans generated by refinement into *Q*.

Fig. 3. The DPLAN search algorithm.

rule but are supposed to be accomplished by some other part of the plan. If all context goals have been achieved, then the plan is returned as a success.

DPLAN can use several different refinement strategies to handle nonoperational goals. There are two main types of goals in DPLAN: activity goals and state goals. Activity goals correspond to actions in the domain that must be performed. State goals correspond to states (or conditions) that must be achieved for an activity goal to be applied properly. Activity goals can be either operational or nonoperational activities and are usually manipulated using HTN planning techniques. Operational activity goals are considered primitive tasks that can be executed directly. Nonoperational activity goals must be decomposed further into operational ones through HTN reduction rules. State goals refer to the preconditions and effects of activity goals, and are achieved through operator-based planning. State goals that have not yet been achieved also are considered nonoperational. Figure 4 shows the procedures used for refining these two types of goals. As soon as a refinement strategy is applied to an activity goal or state goal, it is removed from the list of nonoperational goals.

DPLAN also can use additional domain information for more efficient and flexible planning. For instance, a planning problem can specify a list of static context facts. These facts represent operational goals that are always considered to be true. Such goals are easy for DPLAN to verify during planning and can help in pruning off search branches. Other possible inputs include sets of preconditions and effects for operational activities, a set of final goals that must be true in the plan solution, and a set of initial goals that is true at the beginning of planning. This information is not required for standard DPLAN operation, but can be very beneficial during planning.

> If g is an activity goal,
>
> 1. Decompose: For each decomposition rule r in R that can decompose g, apply r to produce a new plan P'. If all constraints in P' are consistent, then add P' to Q.
>
> 2. Simple Establishment: For each activity goal g' in U that can be unified with g, simple establish g using g' and produce a new plan P'. If all constraints in P' are consistent, then add P' to Q.
>
> If g is a state goal,
>
> 1. Step Addition: For each activity-goal effect that can unify with g, add that goal to P to produce a new plan P'. If the constraints in P' are consistent, then add P' to Q.
>
> 2. Simple Establishment: For each activity goal g' in U that has an effect e that can be unified with g, simple establish g using e and produce a new plan P'. If all constraints in P' are consistent, then add P' to Q.

Fig. 4. Goal refinement strategies.

A. An Example of DPLAN Representation

As mentioned in the preceding section, DPLAN uses several different types of knowledge to construct a plan. A main component of this knowledge is a set of decomposition rules. These rules specify how the planner can break down nonoperational activity goals into lower-level operational goals. A sample rule for performing a telemetry antenna track is shown in Fig. 5. This rule defines how the general telemetry operation is broken down into steps. The left-hand side (LHS) of a decomposition rule consists of a set of initial goals and, possibly, a number of other constraints that specify when the rule should be applied. All initial goals and specified constraints must be true in the current plan for the rule to be selected. The initial goal of a rule is the nonoperational goal that the rule "decompose" into lower-level goals. The rule shown in Fig. 5 has only one initial goal that checks if a telemetry track goal is present in the current plan.[5]

---

[5] Other possible LHS constraints include additional goal conditions that must be present in the plan; context goals, which the planner expects to be achieved by another rule; and codesignation constraints, which check whether two variables can or cannot be unified.

```
(decomprule default-telemetry-track
 lhs
    (initialgoals ((track-goal spacecraft-track telemetry ?track-id)))
 rhs
    (newgoals
        ((g1 (perform-antenna-controller-configuration ?track-id))
         (g2 (configure-metric-data-assembly ?track-id))
         (g3 (perform-microwave-controller-configuration ?track-id))
         (g4 (perform-receiver-configuration ?track-id))
         (g5 (perform-telemetry-configuration ?track-id))
         (g6 (move-antenna-to-point ?track-id))
         (g7 (perform-receiver-calibration ?track-id)))
     constraints
        ((before g1 g6)
         (before g7 g3)
         (before g4 g7))))
```

Fig. 5. Decomposition rule for telemetry track.

The right-hand side (RHS) of a rule contains a set of new goals and constraints over those goals. Once a rule is applied, these new goals replace the LHS initial goals in the current plan. The RHS also contains ordering constraints and protections that specify information about the new goals. An ordering constraint specifies that two goals must be placed in a certain partial order in the final TDN. A protection specifies a causal link that exists between goals. This link explains how the effect of one goal achieves the precondition of another goal. Causal links always must be preserved in order to generate a correct plan. Ordering constraints and protections are added to the current plan and always must be kept consistent during planning. For instance, if an ordering constraint is violated somehow during planning, then the current plan is discarded, and the planner selects another plan from the queue to work on.

Sometimes there may be several different rules that can be used to decompose the same initial goal. For instance, in tracks for 70-m antennas, there are several different methods for configuring a receiver depending on the type of receiver being used. To represent these different methods, there are several different rules that can be used to decompose the perform-receiver-configuration goal (which was asserted by the telemetry rule in Fig. 5). The rules listed in Fig. 6 show two possible ways to break down this goal. The first rule states that if the current goal is to configure the receiver, and the receiver assigned to the antenna track is a Block IV receiver, then the configuration method for Block IV receivers should be used. The second rule states a similar method for Block V receivers. Thus, as shown in these examples, decomposition rules can be used to represent both specific and general domain knowledge.

Another type of knowledge used by DPLAN is a set of activity-goal schemas. These schemas define the parameters, preconditions, and effects that are associated with each activity goal. As explained in

```
(decomprule configure-receiver1
 lhs
    (initialgoals ((perform-receiver-configuration ?track-id)))
     conditions ((CCN-equipment-assignment ?track-id ?equip)
                 (isa ?equip BLOCK-IV-RECEIVER)))
 rhs
    (newgoals ((configure-block-iv-receiver ?track-id ?equip))))


(decomprule configure-receiver2
 lhs
    (initialgoals ((perform-receiver-configuration ?track-id)))
     conditions ((CCN-equipment-assignment ?track-id ?equip)
                 (isa ?equip BLOCK-V-RECEIVER)))

 rhs
    (newgoals ((configure-block-v-receiver ?track-id ?equip))))
```

**Fig. 6. Two rules for decomposing the
perform-receiver-configuration goal.**

Section V, activity-goal preconditions and effects correspond to state goals and are manipulated through operator-based planning techniques. A sample of an activity-goal schema is shown in Fig. 7. This schema definition shows the associated parameters, preconditions, and effects of the calibrate-transmitter task. For instance, this schema reflects that it is necessary to configure the exciter before calibrating the transmitter. Since DPLAN employs a combination of operated-based and HTN planning techniques, a variety of knowledge types can be exploited by the planner. These different knowledge formats allow domain knowledge to be represented more naturally than if only one format were utilized. Each format allows for a different type of knowledge encoding. For instance, decomposition rules allow for the representation of abstract levels of domain objects and goals. Allowing abstract representations of these items allows the user to represent domain information in a more object-oriented form, which is easier to write and reason about. This format also contributes to a more general domain knowledge base that can be efficiently updated and maintained.

Conversely, the utilization of goal schemas and operator-based planning techniques allows certain constraint information to be expressed more easily in the domain. Ordering constraints that are due to precondition–effect interactions are deduced directly during planning, instead of having to be listed explicitly by the user. In particular, ordering constraints that apply to very specialized goals, as opposed to very general ones, can be expressed more easily through precondition–effect schemas than through decomposition rules.[6]

```
(calibrate-transmitter
        :parameters (?track-id)
        :preconditions ((exciter-configured ?track-id)
                        (microwave-controller-configured ?track-id)
                        (transmitter-configured ?track-id))
        :effects (((transmitter-calibrated ?track-id))))
```

Fig. 7. Schema for calibrate-transmitter goal.

B. An Operations Example

In order to begin the planning process, DPLAN is provided with a problem specification that contains several lists of information. Specifically, each problem contains a list of decomposition goals, along with possible lists of initial state predicates, static state predicates, and final state predicates. A sample problem for performing telemetry and ranging with a 70-m antenna is shown in Fig. 8.

The init-state field specifies a list of propositions that are true in the initial state of the planner. For instance, as shown in Fig. 8, the exciter drive is assumed to be off prior to the time of the track. The static-state field specifies a list of propositions that always are true during planning (i.e., that can never be deleted) and commonly is used to list equipment types available to the track. The decompgoals field holds the list of nonoperational goals that are to be broken down into lower-level goals through the use of decomposition rules. The final-state field is a list of propositions that must be true in the final plan. The init-state, static-state, and final-state fields are not necessary for standard planner operation and can be left empty. However, these fields are very beneficial for increasing planner efficiency by providing extra domain knowledge. Other inputs to the planner include a list of decomposition rules and a list of goal schemas, which were explained in the previous section.

---

[6] For more information on the advantages and disadvantages of employing HTN and operator-based planning techniques for this type of domain, see S. Chien, S. Wang, and T. Estlin, *Hierarchical Task Network and Operator-Based Planning: Competing or Complementary?*, JPL D-13390 (internal document), Jet Propulsion Laboratory, Pasadena, California, January 1996.

```
(decompproblem TELEM70
   (init-state ((exciter-drive-off track1)
                (range-mode-off track1)
                (test-translator-off track1)))
   (static-state
      ((CCN-equipment-assignment track1 bstring1)
       (isa bstring1 type-B-telemetry-string)
       (CCN-equipment-assignment track1
                                       APA-70m)
       (isa APA-70m APA)
       (CCN-equipment-assignment track1 bvr1)
       (isa bvr1 BVR)
       (CCN-equipment-assignment track1 rec1)
       (isa rec1 REC)
       (CCN-equipment-assignment track1 ugc1)
       (isa ugc1 UGC)))
   (decompgoals
       ((perform-pre-cal track1)
        (track-goal spacecraft-track telemetry track1)
        (track-goal spacecraft-track ranging track1)))
   (final-state () ))
```

Fig. 8. Problem specification for a
telemetry and ranging track.

DPLAN currently is started by executing the following command from the UNIX prompt:[7]

<div align="center">

dplan <problem-string>   <output-filename>   <annotation-filename>

</div>

The problem-string input is a problem name (e.g., 34 m or 70 m). When this string is given, the planner will expect to find the following files:

<div align="center">

rules-<problem-string>

goals-<problem-string>

prob-<problem-string>

</div>

The "rules" file specifies the list of decomposition rules; the "goals" file specifies the list of goal schemas, and the "prob" file specifies the particular problem specification (including initial state, decomposition goals, etc.). DPLAN will parse the information in these files (using GNU tools flex and bison) into a usable form. Then, using the algorithm introduced in Section V, DPLAN will generate a plan that successfully achieves all decomposition goals and any final-state goals listed in the problem specification.

A final plan contains a large amount of information, including a list of operational goal names (corresponding to TDN blocks), a list of ordering constraints over those goals, and a list of annotations that describes how the plan was built (i.e., what rules and operations were used). Currently, the planner outputs this information in the following way: Three output files are created—a text output file, an annotation file, and a graph-input file. The text output file contains a textual listing of blocks and parameters where blocks are listed in a correct ordering (i.e., blocks do no violate any plan ordering constraints). The annotation file contains a textual list of annotations describing the plan and how it was constructed. The graph-input file contains a list of node names and ordering constraints, which can be used to construct a graphical representation of the plan. See Fig. 9 for an example of a plan (or TDN) that was generated for a problem specification such as that shown in Fig. 8.

---

[7] Future development plans include building a more sophisticated user interface that will allow the user to interact easily with the planner.
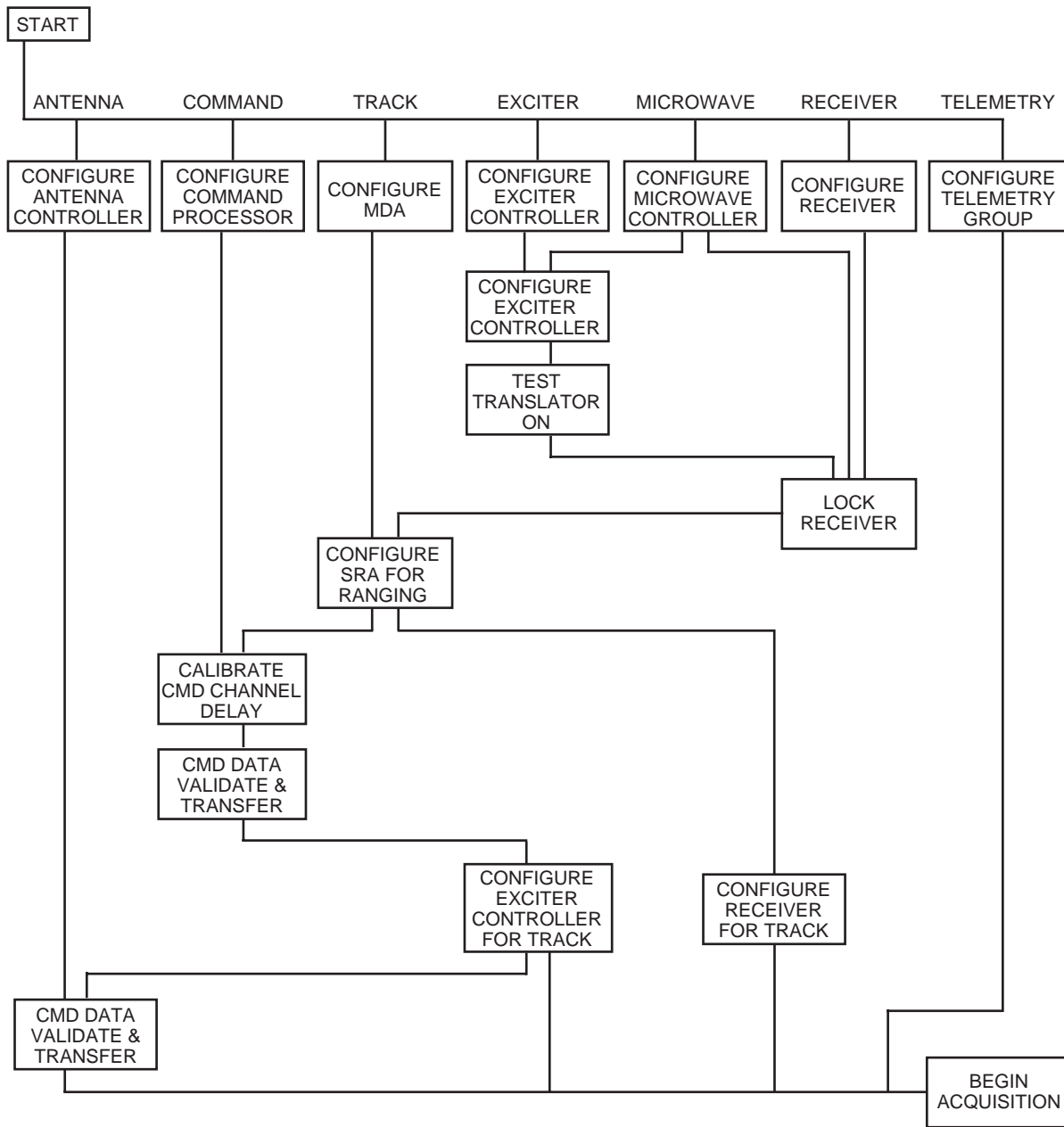
Fig. 9. Temporal dependency network for a 34-m beam wave-guide antenna pretrack for telemetry, commanding, and ranging services.

VI. Current Status

The knowledge base for the planner currently supports all the antenna types at the DSN.[8] All standard types of spacecraft passes for each antenna type are implemented in the knowledge base. Spacecraft passes include the following:

---

[8] Except for the 26-m antenna, which may be phased out in the near future.

(1) *Telemetry.* Telemetry is a downlink with the spacecraft wherein information is relayed from the spacecraft to the DSN station on Earth.

(2) *Ranging.* Ranging is a method of finding the distance between the spacecraft and the Earth, which requires both an uplink and a downlink to the spacecraft.

(3) *Commanding.* Commanding is an uplink to the spacecraft wherein commands are sent from the DSN station to the spacecraft, which instructs the spacecraft to carry out given tasks.

(4) *VLBI ΔDOR.* Very long baseline interferometry (VLBI) uses quasars—distant space objects—in order to determine the location of a spacecraft. A VLBI ΔDOR (delta differential one-way range) service provides information on the spacecraft's angular position by performing simultaneous observations from two antenna stations of the spacecraft and a quasar, followed by a second observation of the spacecraft to gather Doppler data. These data then are used to determine how to maneuver the spacecraft through space to its destination.

(5) *VLBI clock synchronization.* VLBI clock synchronization gives the instantaneous position of two stations relative to a quasar. This pass is performed in order to determine the rate of change of the clocks at the two DSN stations.

(6) *Radio science.* For radio science, the antenna station is used to gather radio frequency (RF) signal information from spacecraft transmissions or natural sources (such as a planet or star).

Not all antenna types perform all types of spacecraft passes. For example, the 34-m standard antenna is not used for any type of VLBI activity. For each of the antenna types, the DPLAN knowledge base contains all the types of spacecraft passes for which that antenna type is used, as follows:

(1) 34m BWG (34-m beam wave-guide antenna): telemetry, commanding, and ranging.

(2) 34m STD (34-m standard antenna): telemetry, commanding, and ranging.

(3) 34m HEF (34-m high-efficiency antenna): telemetry, commanding, ranging, VLBI ΔDOR, and radio science.

(4) 70m BIVR and BVR (70-m antenna with Block IV and Block V receivers): telemetry, commanding, ranging, VLBI ΔDOR, VLBI clock synchronization, and radio science.

(5) 70m BIVR (70-m antenna with Block IV receiver): telemetry, commanding, ranging, VLBI ΔDOR, VLBI clock synchronization, and radio science.

Generating a plan to make an antenna operational and ready to communicate with a given spacecraft is a complex process. The starting up and turning on of different pieces of equipment and subsystems must be coordinated carefully. DPLAN's knowledge base covers information on the full range of antenna equipment from the primitive on/off operation of the test translator to the sophisticated operation of the antenna gain controller (AGC) or the microwave controller (UGC). Also covered is information on all subsystems necessary to antenna operation, such as the receiver subsystem, the antenna subsystem, the exciter–transmitter subsystem, and others. DPLAN's knowledge of the equipment and subsystems is used to determine in what order the TDN blocks associated with each piece of equipment and subsystem must be executed.

The total number of rules in the knowledge base (covering all antenna and track types) is 197: 91 decomposition rules (an average of 23 decomposition rules per antenna type) and 106 goal schemas. The knowledge base is modular and easily extended to accommodate new antenna types and new subsystems

or equipment types. Also, as changes are made to existing antennas, equipment, and subsystems, the rules can be modified easily. For example, if a new type of antenna controller is added to the 34-m HEF antenna, then a new rule simply is added that configures the new antenna controller. Because of the decomposition structure of the knowledge base, other rules that use the antenna controller rule do not need to be changed.

All the plans generated by the planner for the different antenna types and their valid spacecraft passes (including a majority of the multiple combinations of passes) have been verified by the DSN operator experts from all three of the DSN complexes: Goldstone, California (October 1995); Madrid, Spain (January 1996); and Canberra, Australia (May 1996). For example, the 34-m STD antenna can support telemetry, ranging, and commanding spacecraft passes, and any combination of those three types of passes. DPLAN generated all of the resulting seven combinations of spacecraft passes (telemetry, ranging, telemetry and ranging, telemetry and commanding, etc.). These passes were then verified on paper by the various operator experts as being correct, executable plans in terms of the ordering of the TDN blocks and the inclusion (or exclusion) of sufficient and necessary TDN blocks.

More testing will occur during the integration phase. During integration, the plans generated by DPLAN are executed by the Automation Engine (AE), which fires scripts associated with each TDN block in the plan. The scripts execute "operator directives" that turn on and off pieces of equipment, configure subsystems, move the antenna, etc.

A preliminary demonstration that integrated the planner with the other elements that comprise the DSN automation was done successfully. The planner successfully constructed a plan, which was then executed (in simulation) by the AE. This demonstration took place in December 1995. Further testing of the planner took place in August 1996 in a computer-simulated antenna environment with simulated subsystems and equipment. During final integration, anticipated in August 1997, DPLAN will be integrated fully: the AE will call the planner to generate a given plan and then execute that plan, firing off the necessary scripts for the TDN blocks. This first will be tested in the antenna simulator environment and then tested at the Goldstone, California, DSN complex.

VII. Discussion

In this section, we discuss several issues relevant to the DSN planner, including a comparison of the DPLAN AI planning approach to alternative methods of automation, representation issues for maintainability, plan quality, and replanning.

A. Representation for Maintainability

An important aspect of the DPLAN representation is that it allows for natural encoding of abstract objects and procedures (e.g., receiver calibration). By allowing decomposition rules to refer to abstract objects, changes to DSN procedures involve fewer knowledge base updates than if the knowledge base contained a large number of very specific rules. For instance, a change relating to a specific equipment type need not affect more general domain information. If a new receiver type, called a Block VI receiver, were added to the DSN equipment list, more general rules, such as the telemetry rule shown in Fig. 5, would not need to be modified. Instead, only a few more specific rules would need to be constructed or edited. In this case, a new configure-receiver rule would be added to the set of rules shown in Fig. 6. Therefore, many such changes would cause only a few specialized rules to be created or updated instead of causing numerous rules to be modified. Even with the current DSN goal of automating all TDN generation, the planning knowledge base must be updated and verified constantly. Fewer more general rules are cheaper to update and verify and, thus, support more efficient knowledge base maintenance.

Another benefit of this type of representation is that domain information is understood more easily. By keeping domain details separate from more general knowledge, it is easier for a user to understand the

general aspects of an antenna track. For example, to understand the general steps of a telemetry operation, a user has only to view the main telemetry track decomposition rule. If more low-level knowledge is desired, such as how to operate a particular piece of equipment, the user could then search for rules that directly pertain to that equipment type.

B. Comparison to Scripts

One option considered by DSN personnel was to implement the higher level of track automation by a hierarchy of scripts. There would be scripts for general activities, such as calibrating a Block V receiver in the context of a ranging track. This scripting approach can be viewed as similar to the HTN planning approach, but with two key differences. First, there is no explicit representation of the context in which a script necessarily will achieve the goal. The set of situations in which a script S is expected to work is represented only implicitly in the set of scripts that call S. The intended coverage, conditions, etc., are not explicitly represented, as they are in HTN rules. The second difference is that the planner allows a "call by goal" usage in operator-based planning. In this way, the planner can invoke routines (or operators) based on the conditions it desires to achieve, and the planner automatically will detect and resolve any conflicting interactions with other activities. Not only does the planner representation allow for encoding of conditions and assumptions of when particular activities are appropriate (through conditions on HTN rules or preconditions on operators), it actually requires such definitions in order to operate correctly. Therefore, it encourages correct documentation of operations requirements for all activities.

C. Comparison to End-to-End TDNs

Another option considered by DSN Operations was simply to encode end-to-end TDNs for each supported combination of the cross-product between service requests and equipment allocation. Unfortunately, this option has several drawbacks. First, articulating all of the relevant knowledge in this format can be very tedious and prone to error. While generating the initial set of end-to-end TDNs, the expert operators said that they often found it difficult to keep all of the different TDNs straight. Second, this representation is not amenable to maintenance. If an equipment type is added or changed, it must be changed in every TDN that is relevant. The knowledge pertaining to the equipment type is not centralized in a set of rules or activity definitions as it is in the planning representation.

D. Representing and Reasoning About Plan Quality

Representing and reasoning about plan quality [9,14–16] is another key concern of DSN operations. Since there often is more than one correct plan for a particular antenna operation, it is important for a planning system to be able to compare a set of final plans using user-identified plan quality measures. There are a number of quality measures that can be emphasized during planning, including producing more robust, flexible, and/or efficient plans. One important quality goal is to minimize the overall plan execution time. In particular, the time to set up (precalibration) and reset (postcalibration) the communications link often can be reduced. For instance, it can take up to 2 hours to manually precalibrate a DSN 70-m antenna communications link for certain types of services. By using a plan generated by DPLAN, the time to perform the same services can be reduced from 2 hours to approximately 30 minutes, where further reductions in setup time are limited by physical constraints of the subsystems themselves.

Plan execution time often is significantly reduced by exploiting parallel path possibilities, especially where the control of multiple subsystems is involved. DPLAN currently uses the critical path length of a plan to help identify better plans. Critical path length is calculated using time information attached to a TDN block, which specifies the average time it should take to execute the block. By comparing critical path lengths of competing plans, DPLAN could choose a highly efficient final plan that will provide a minimal execution time. Minimizing plan execution time allows more data to be returned per operating time for the link.

Another important measure of plan quality is generality. Because of the considerable effort involved in generating, maintaining, and refining TDNs, a single generalized TDN is cheaper than hundreds or

thousands of experiment-specific TDNs. For example, in one radio science experiment performed in the DSN, called the Ka-band Antenna Performance (KaAP) experiment, the TDN currently produced is considered a generalized TDN since it represents the many different ways that a KaAP experiment can be executed. The support data for each particular KaAP experiment identify a particular path through the TDN. This path can change depending on the particular mission requirements. In particular, there is a data capture loop in the KaAP TDN that allows data to be captured from either a star or a planet, thus requiring different antenna modes. One experiment may specify that data be acquired from the following sources in sequence—star1, star2, star3—whereas another experiment may specify that data be acquired from star1, planet1, star1, star2. The more general TDN helps provide for more efficient knowledge maintenance since only one TDN must be maintained for this type of track.

Flexibility is another aspect of plan quality that has been a requirement in the DSN domain. The support data for a particular experiment may specify a particular path through a TDN; however, the DSN operator has the flexibility to alter this path in real time. The final plan must be flexible enough to handle these real-time changes. Some of the changes that the operator can make to the TDN are skipping blocks, deleting commands in blocks, adding commands in blocks, and editing time tags on blocks. It also may be necessary (or desirable) for an operator to reorder blocks. For example, some TDN blocks cannot execute in parallel due to resource conflicts. The ordering of such blocks often can affect plan quality by making a plan more robust or more efficient, depending on the particular antenna operation and current track status. If a better ordering is known prior to TDN generation, this information can be input to the planning system, which will incorporate it into the final TDN. However, these ordering constraints often are best determined at run time by the operator, thus requiring flexible plans that can handle last-minute constraint additions.

There also are some standard TDN blocks that may be inserted into a plan at various points (such as transmission rate changes, etc.). If such commands are executed in the middle of an inflexible plan, it may not be possible to continue execution. Depending on the steps inserted, preconditions, postconditions, and time tags of other blocks may become invalid. Flexible plans that allow for the insertion of common steps while still retaining their applicability are greatly valued since they allow execution to continue and avoid the need for replanning in these situations.

A final plan quality issue is robustness. Ideally, the final plan representation should be expressive enough to provide robustness under a variety of situations; however, an expressive representation usually increases an application's complexity and often results in a loss of generality. In the DSN application, the TDN representation used to represent the final plan initially has been kept extremely simple, although it does include parallelism. As the intricacies of particular antenna procedures become evident, more expressive representations may be required. Constructs such as loops, metric time, and actions with temporal scope could be added. Unfortunately, this may cause the creation of very specific constructs that are overly specific to a certain track. For instance, in the Ka-band Antenna Performance track, a useful planning construct to have is a "loop until time" construct, which is used when the actions in a loop need to be executed until a prespecified time occurs. So far, such a construct has been deemed necessary for only one particular kind of antenna track and, thus, may not be applicable in other tracks. By adding such a construct, the plan representation does become more flexible and may provide for more robust plans. However, such an addition also increases planning complexity. This trade-off may be necessary in order to generate plans that are robust enough to be executed correctly in such a varying and complicated domain as the DSN.

E. Replanning for Antenna Tracks

DPLAN also is required to replan during the course of typical antenna operations. Replanning occurs in two general cases. First, after a plan has been generated, the objectives sometimes change. Before, shortly prior to, or during a track, a project may submit a request to add services to the track. This request corresponds to additional goals that must be incorporated into the track plan. In the case when

goals are added before the track actually begins, DPLAN adds these unachieved goals to the current plan and restarts the planning process. Unfortunately, this method is incomplete in theory because the planner may have previously made choices that are incompatible with the new goals. However, for the specific sets of goals and domain theories (related to antenna operations) that we have examined, we have been able to use encodings in which completeness has not been a problem. This is an area of current work. Another area of current work is replanning in the case when goals are added during actual track execution. One approach to dealing with this would be to allow the planner the ability to backtrack or repair the current plan so as to adapt to the current situation. The planner might do this using a set of plan modification operators.

Another replanning issue is caused by dynamism. After a plan has been generated, a block (plan step) may fail, a piece of equipment may require resetting, or a piece of equipment may fail or be preempted by a higher priority track. In the case of a simple plan step failure, DPLAN simply calls for reexecution of the block. If a piece of equipment requires resetting, DPLAN has knowledge describing which achieved goals have been undone and require reestablishment. DPLAN then uses a replanning technique that reuses parts of the original plan to reachieve the undone goals as necessary.[9] This technique takes advantage of the fact that the original plan begins from a state that is equivalent to resetting all of the subsystems.

VIII. Conclusions

This article has described the DSN Antenna Operations Planner (DPLAN), which automatically generates communications antenna tracking plans based on requested services and equipment allocation. DPLAN uses a knowledge base of information on tracking activities and a combination of artificial intelligence planning methods to generate appropriate tracking plans. We also have described the deployment status of the DPLAN system and outlined areas of current work, including representation and reasoning about plan quality, replanning, and representation to support maintainability.

References

[1] J. G. Carbonell, J. Blythe, O. Etzioni, Y. Gil, R. Joseph, D. Kahn, C. Knoblock, S. Minton, M. A. Perez, S. Reilly, M. Veloso, and X. Wang, *Prodigy 4.0: The Manual and Tutorial*, Technical Report, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, June 1992.

[2] S. A. Chien, R. W. Hill, Jr., X. Wang, T. Estlin, K. V. Fayyad, and H. B. Mortensen, "Why Real-world Planning Is Difficult: A Tale of Two Applications," in *New Directions in AI Planning*, M. Ghallab and A. Milani, eds., Washington, DC: IOS Press, pp. 287–298, 1996.

[3] S. Chien, A. Govindjee, X. Wang, T. Estlin, and R. Hill, Jr., "Integrating Hierarchical Task Network and Operator-Based Planning Techniques to Automate Operations of Communications Antennas," *IEEE Expert*, Winter 1996.

[4] S. Chien, R. Lam, and Q. Vu, "Resource Scheduling for a Network of Communications Antennas," *Proceedings of the 1997 IEEE Aerospace Conference*, Aspen, Colorado, February 1997.

---

[9] X. Wang and S. Chien, *Replanning for the Deep Space Network (DSN) Antenna Operations Planner: Preliminary Report*, JPL D-13388 (internal document), Jet Propulsion Laboratory, Pasadena, California, January 1996.

[5] *Deep Space Network*, JPL Publication 400-517, Jet Propulsion Laboratory, Pasadena, California, April 1994.

[6] K. Erol, J. Hendler, and D. Nau, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," *Proceedings of the Second International Conference on AI Planning Systems*, Chicago, Illinois, pp. 249–254, June 1994.

[7] K. E. Fayyad and L. P. Cooper, "Representing Operations Procedures Using Temporal Dependency Networks," *SpaceOps '92*, Pasadena, California, November 1992.

[8] K. Fayyad, R. W. Hill, Jr., and E. J. Wyatt, "Knowledge Engineering for Temporal Dependency Networks as Operations Procedures," *Proceedings of AIAA Computing in Aerospace 9 Conference*, San Diego, California, 1993.

[9] J. M. Gratch, S. A. Chien, and G. F. DeJong, "Learning Search Control Knowledge to Improve Schedule Quality," *Proceedings of the 1993 Workshop on Knowledge-Based Production Planning, Scheduling, and Control*, Chamberry, France, pp. 159–168, August 1993.

[10] R. W. Hill, Jr., S. A. Chien, and K. V. Fayyad, "Automating Operations for a Network of Communications Antennas," *Proceedings of the 1996 IASTED International Conference on Artificial Intelligence, Expert Systems, and Neural Networks*, Honolulu, Hawaii, August 1996.

[11] R. Hill, Jr., K. Fayyad, C. Smyth, T. Santos, R. Chen, S. Chien, and R. Bevan, "Sequence-of-Events-Driven Automation of the Deep Space Network," *The Telecommunications and Data Acquisition Progress Report 42-124, October-December 1995*, Jet Propulsion Laboratory, Pasadena, California, pp. 153–173, February 15, 1996.
http://tda.jpl.nasa.gov/tda/progress_report/42-124/124H.pdf

[12] R. W. Hill, Jr., S. Chien, C. Smyth and K. Fayyad, "Planning for Deep Space Network Operations," *Proceedings of the 1995 AAAI Spring Symposium on Integrated Planning Applications*, Palo Alto, California: AAAI Press, 1995.

[13] J. S. Pemberthy and D. S. Weld, "UCPOP: A Sound Complete, Partial Order Planner for ADL," *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, Cambridge, Massachusetts, October 1992.

[14] M. Williamson and S. Hanks, "Optimal Planning With a Goal-Directed Utility Model," *Proceedings of the Second International Conference on AI Planning Systems*, Chicago, Illinois, pp. 176–180, June 1994.

[15] M. Williamson and S. Hanks, "Flaw Selection Strategies for Value-Directed Planning," *Proceedings of the Third International Conference on AI Planning Systems*, Edinburgh, United Kingdom, pp. 237–244, May 1996.

[16] A. Perez and J. Carbonell, "Control Knowledge to Improve Plan Quality," *Proceedings of the Second International Conference on AI Planning Systems*, Chicago, Illinois, pp. 323–328, June 1994.