N88-16428

Prototype Resupply Scheduler

Steve Tanner and **Angi Hughes**
General Research Corporation
635 Discovery Drive
Huntsville, AL 35806

Jim Byrd
United Space Boosters Incorporated
188 Sparkman Drive
Huntsville, AL 35805

Abstract

Resupply scheduling for the Space Station presents some formidable logistics problems. One of the most basic problems is assigning supplies to a series of shuttle resupply missions. Some supplies relate to life-support, others are required by critical experiments, and still others are necessary for routine maintenance. If an emergency occurs or the space station inventories are depleted unexpectedly, resupply plans must be quickly adapted. The speed at which a logistics expert can replan schedules is a critical factor in the successful operation of the space station. The logistics expert requires a great deal of knowledge to construct a resupply schedule which satisfies the life-support, experiment-support, and maintenance constraints.

The Artificial Intelligence Department of General Research Corporation (Huntsville) with the logistics expertise of United Space Boosters Incorporated constructed a prototype logistics expert system which constructs resupply schedules. This prototype is able to reconstruct feasible resupply plans and, in addition, analysts can use the system to evaluate the impact of adding, deleting or modifying launches, cargo space, experiments, etc.

1 Introduction

In this paper the Heuristic Assistance in Tactical Scheduling (HATS) prototype system is described. This system was built by Steve Tanner and Angi Hughes of GRC with domain expertise provided by Jim Byrd of USBI. It was implemented on a Symbolics 3670 lisp machine with the use of the KEE software tool. A more advanced implementation will be undertaken using tools developed in house and which will work in VAX and PC environments. The system, while still very much in its infancy has proved viable for small scheduling tasks, and with time will work with larger more complex problems.

The system is general enough in nature to help with many types of scheduling and logistic problems, however, the main problem domain addressed by HATS is the supply and resupply payload schedules for shuttle launches necessary to maintain the space station.

Some of the techniques that HATS incorporates are: object oriented structures that take full advantage of inheritance facilities, hypothetical reasoning features that allow the system to try different solution paths without data corruption, heuristic methods to allow for some level of control and constraint of the immense search space that scheduling problems often generate, and rule based reasoning to help with the capture of domain expert knowledge.

2 Structure of the Knowledge Base

By using a rich object oriented environment in which to implement this system, a great deal of reusable and interrelated information can be stored in a concise, consistent and understandable manner. The two main categories of objects dealt with are the supplies that must be scheduled on launches and the launches themselves. The supplies that are to be scheduled include a broad range of supplies necessary to operate and maintain the space station, support experiments and deploy, retrieve and maintain communications and observation satellites. This means that criticalities I, II, and III items are scheduled as well as the experiment containers, platforms and any equipment that the experiments themselves need. Extra crew and scientists that a particular experiment needs are simply represented as another necessary supply.

Thanks to the use of inheritance, much generic information about each of these categories is defined in their high level object abstraction. The specific information about each individual launch and supply is stored in each individual object instantiation. Stored information for launches includes such data as projected launch date, maximum payload size and weight, crew size (excluding experiment scientists), and other pertinent launch information. Supply information is considerably more complex. The supply objects have basic size and weight information, but in addition, they have date needed by, and date needed after information. Also, because of the interrelated nature of some supplies, relationships between supply objects are also possible. For example, as mentioned above, extra crew needed for a particular experiment are scheduled as another supply. It would do little good to schedule the extra crew for one launch and the experiment equipment for another. There must be some way for each supply to know what other supplies it requires. This type of information is easily stored in attribute slots on the objects themselves.

As another more complicated example of supply interrelationships, there can be several scheduling paths based on interrelationships of supplies. In the above case, the experiment equipment could indeed go up on a launch prior to the experiment crew. The equipment could sit dormant on the space station until the required personnel finally showed up. It is also possible, but

not likely, that the crew could go up before the experiment equipment and sit around until it arrived. This scenario happens quite frequently with field service personnel back on earth. The three different schedules mentioned here (crew and equipment on one launch, equipment first, crew first) are all three possible and even workable supply schedules. Ways to make the system favor one schedule over the others is discused a little later on, but the ability for objects to have defined relationships with one another is a great help in sorting out these problems.

Because objects in this environment allow for methods, these supply objects can be made to automatically make changes of themselves and other objects as the data changes. (Methods are like programs that can be run when certain events occur.) For example, adding an extra crew member for an experiment will mean that more food and oxygen will be consumed during a specific time period. This means that there must be a corresponding change in the food and oxygen supply objects. Either current objects must be increased, dates shifted, or new objects created. This type of automatic readjusting can be handled by methods that are keyed to run whenever supply objects are added. Also methods can be used to create supply objects to help with cyclic types of resource use. For example, the user may tell the system that 100 pounds of gas x is used per month. The methods may automatically generate the necessary gas x objects to make sure this requirement is met. Direct down links will eventually provide on-orbit inventory quantities against which the system will automatically plan replenishment payloads.

Another type of interrelationship of supplies is one supply with a fixed date of need and another supply with no known date, but with a known link to the first. Because the first supply has a fixed date, the second supply must be scheduled either before or concurrent to the first supply. This means that even though the second supply has no obvious date itself, there is a necessary scheduling date inherent in its relationship with the first supply.

The initial state of the knowledge has the basic set up of the supply and launch data. The user of HATS has a graphical user interface to help with this initial set up. The dates placed on the supplies is optional, and as the system works, any unknown dates are taken to mean that the supply can go on any launch, unless of course there are relationships with other supplies.

3 Hypothetical Worlds and Rules

Once the initial data has been set up, the system tries to find a way to place all the supplies on the limited number of launches available. This is accomplished by the use of a rule base and two techniques known as hypothetical reasoning and worlds. As rules in a rule base are fired, they generally assert new facts. In hypothetical reasoning these new facts are considered hypothesis rather than true facts. These hypothesis are asserted only in separate autonomous worlds. In this way, if a trail of asserted facts leads to a dead end, there is no need to retract all the facts. Often it is

impossible to retract a fact, and at the very least it is time consuming and difficult. With hypothetical worlds, if a path leads to a dead end, the useless worlds are ignored or thrown away, and no facts need to be retracted.

As the system runs, a trail of worlds is created as a tree. Each world leads to at least one more world until a dead end is reached. A dead end means that either a workable schedule has been found and the system can quit, or this current path is unworkable and another must be found. Each world has one supply placed on one launch. If there are 10,000 supplies to be scheduled, a branch of worlds with a depth of 10,000 nodes represents a workable solution. If the branch does not go 10,000 deep, then some supplies would not fit the current launch configuration.

These worlds can be used as a simple brute force method for finding all possible scheduling solutions. It is a simple matter to try all supplies on all launches until a complete path is found. However, this is combinatorial and even with the use of several super computers, would require an unreasonable amount of time. This is especially problematic for last minute payload changes. That is where the use of the rule base can really pay off.

Because rules are used to create the world tree, they can help to trim the tree and eliminate many of the paths that will probably lead to unworkable dead ends. As a very simple example, scheduling larger items on launches before looking at the smaller items will point out dead ends far faster than the other way around. Also Criticality I items can be scheduled before other items so that if only partial schedules are found, then the critical items are scheduled and optional equipment can be left behind.

Another advantage with the combination of worlds and rules is that new ideas can be tested quickly and effectively. It is fairly easy to change or add rules to the rule base. If a change speeds things up and finds solutions faster, then the new technique can be left in place. If not, then the changes need only be taken out of the rule base.

4 Future Work

As funding allows, work will concentrate on several areas. The rule based heuristics that help trim the search tree will be improved considerably. The current rule base takes about thirty criteria into account. As domain expertise is codified, this will improve and expand. This expansion will be aimed specifically at speeding up the system by trimming unfruitful branches off the world search tree.

The interrelationships between objects will be expanded. This area is fairly rudimentary now and should be revamped and improved. Many types of relationships have been defined and need only be incorporated into the system. The infrastructure is already in place to do this, and is flexible enough to take new relationships into account as well.

The system will be rewritten to use an Entity Attribute Relationship database and rule system that we are currently working on in house. This system will be considerably faster than the current knowledge base and rule system now being used. During this rewriting, a more generic form of HATS should emerge. This generic tool will be useful as a baseline for other types of scheduling problems.

An improved user interface will be implemented. The current interface is very useful for creating launch and supply objects and placing them in the object hierarchy, however there are several things that need to be added. The interface needs to make defining relationships between objects easier and filling in slot attributes easier.

The entire system will be ported to more mainstream types of machines. The Symbolics is a very good environment in which to develop systems like this, however at this time it is unreasonable to assume that eventual users of HATS will have such equipment on their desks.

More forward looking analysis techniques should be added to the system. For example, if no complete schedule can be found, the system could suggest other solutions. It might determine what items would cause the least trouble if left behind, or suggest the fewest number of additional launches that would alleviate the problem.

5 Conclusions

The HATS system has shown that several AI techniques are useful when applied to scheduling problems. Basic object oriented structures are good for setting up the types of data that are required and in defining the relationships between the data. Objects that represent launch and supply items are easy to manipulate and alter. A world tree is a useful way to represent the planning paths taken while determining a schedule. Rule based reasoning is very effecting in both creating the tree and constraining the search paths that the tree represents. The system is flexible and can expand as domain expertise is incorporated. Application of this system to logistics support will not only improve support response and effectiveness, but also provide a valuable tool for future program planning.