SOFIA's Choice: Automating the Scheduling of Airborne Observations *

Jeremy Frank
Caelum Research Corp.
NASA Ames Research Center
Mail Stop N269-1
Moffett Field, CA 94035-1000

Why SOFIA's Choice Matters

This paper describes the automated scheduling of observations for an airborne observatory. The problem is to construct flight plans in support of astronomical observations of very distant objects in order to optimize the science output of the observatory. The resulting problem contains a large set of prioritized observations to choose from, and a wide range of complex constraints governing legitimate choices and orderings.

This problem is quite different from scheduling problems which are routinely solved automatically in industry. For instance, the problem contains many interacting complex constraints over both discrete and continuous variables, and the consequences of making some decisions are other decisions to make later. Furthermore, new types of constraints may be added as the problem changes over time. As a result of these features, this problem cannot be solved by traditional scheduling techniques. The problem resembles other problems in NASA, from observation scheduling for rovers and other science instruments to vehicle routing; consequently, it is worthwhile to determine how to address this problem.

In this paper we describe the observatory and the problem of planning flights in support of astronomical observations. We discuss why this is a difficult problem to solve using traditional scheduling techniques, then discuss an approach based on dynamic constraint satisfaction techniques which can address this problem.

SOFIA: The Observatory

The Stratospheric Observatory for Infrared Astronomy (SOFIA) is NASA's next generation airborne astronomical observatory. The facility consists of a 747-SP modified to accommodate a 2.7 meter telescope. Employing a suite of optical, infrared, and sub-millimeter instrumentation, the observatory spans operational wavelengths of 0.3 to 1600 microns. SOFIA supersedes NASA's Kuiper Airborne Observatory (KAO) - a modified C-141 with a 0.9 meter telescope. SOFIA is expected to fly an average of 140 science flights/year over it's 20 year life time, double the previous rate of the KAO. The combination of a factor of nine in telescope collecting area and an approximate factor of two in aircraft flight rate establishes SOFIA as NASA's premier observatory for innovative astrophysical instrumentation throughout the broad wavelength range of the facility. More details on SOFIA can be found in (Becklin 1997) and (Erickson & Davidson 1997).

Building upon the KAO program, SOFIA will routinely provide astronomers with a research platform above 99% of the earth's water vapor. The SOFIA telescope is mounted aft of the wings on the port side of the aircraft and is articulated through a range of 20 to 60 degrees of elevation. Most flights will originate and terminate at Moffett Field, CA; therefore, it is necessary for the observatory flight plans close in on themselves. This typically requires an astronomical observing plan covering both Galactic and extra-galactic targets.

In this paper, we are concerned primarily with the problem of scheduling flights in support of the General Investigator (GI) program. GIs are expected to propose single observations, and many observations must be grouped together to make up single flights. The SOFIA science staff is expected to have 3 - 4 facility science instruments to support GIs. The scope of the flight planning problem for supporting GI observations with the anticipated flight rate for SOFIA makes the manual approach for flight planning daunting. There has been considerable success in automating the scheduling of jobs in a wide variety of industries with many different types of constraints. However, these problems are typified by relatively simple, homogeneous constraints, and the successful approaches depend on these simple representations. In this paper, we describe an approach to solving more complex scheduling problems which can address the difficult problem of planning flights.

Planning for a Single Flight

The basic problem for an airborne observatory like SOFIA is the *Single Flight Planning Problem* (SFP). This problem consists of constructing a good flight plan for a single flight on a given day. The problem input consists of the set of *observations* that have been requested, the constraints peculiar to the *flight environment*, and the *objective function*.

*I would like to thank Ari Jónsson, David Smith, Sean Casey, and Jesse Bregman for their help in writing this paper.

The Observation Requests

An observation request consists of the name and coordinates of the object to be observed, the amount of time requested, the relative importance of the observation, and a set of constraints on the observation. We assume that the amount of time is fixed and also that it is strictly less than the maximum duration of the flight. The importance or priority of the observation is a summary of several different factors. Some observations are naturally more interesting to the science community than others. Additionally, due to the limited duration of flights, it may be necessary to observe a target many times, and thus important to finish a sequence of observations on a target than to start a new observation. Observation requests have the following constraints:

Ordering Constraints Some operations must be performed in a pre-specified order. For example, instruments may need to be calibrated by observing particular objects before the primary observation of interest is performed. In addition, the telescope may need to be tuned at the beginning and periodically during the flight by observing objects with particular characteristics. High-precision tuning may require observing the same object at multiple elevations, for instance. These requirements impose ordering constraints on the observations that must be obeyed [1].

Astronomical Constraints Some objects may only be visible from certain positions on the earth at certain times of day, resulting in an earliest start time and a latest end time for completing a given observation request. Astronomers may also provide explicit constraints on particular observations so that the data is of high quality. For example, the astronomer may require that the object be sufficiently far away from the moon or the sun, or that airmass or atmospheric water vapor be below a certain threshold. These constraints also dictate when a target may be observed. In particular, minimizing airmass requires observing at a higher altitude, and minimizing water vapor can be accomplished by observing at higher altitudes or by observing further north (Horn & Becklin 2000).

Aircraft Constraints SOFIA has complex constraints simply because it is an airborne observatory. Most objects appear to move through the sky as time passes. Because the telescope has little horizontal flexibility, the aircraft must fly a curved trajectory in order to keep objects in view. The wind speed, aircraft speed, and time and position an observation is started dictate the trajectory and final location of the aircraft at the end of the observation. Object visibility windows are further constrained by the limits on the telescope's angle of elevation. Even though an object may be visible from the ground, it may not be visible from the aircraft, either because it is too low or too high for the telescope to view. An object may sometimes have multiple windows of visibility during a single flight. For example, it may pass above and then below the maximum telescope elevation, requiring a choice of when to observe.

The aircraft must normally return to the airport it took off from. Flight time is limited by fuel, requiring all observations to be done with enough time for the aircraft to return, from wherever it is, to the airport. Finally, the aircraft's altitude is constrained by its weight, but the weight decreases over time as fuel is consumed, so the aircraft can generally climb an additional 2000 ft every two hours. These factors can interact with constraints on airmass or water vapor to further limit possible observing times.

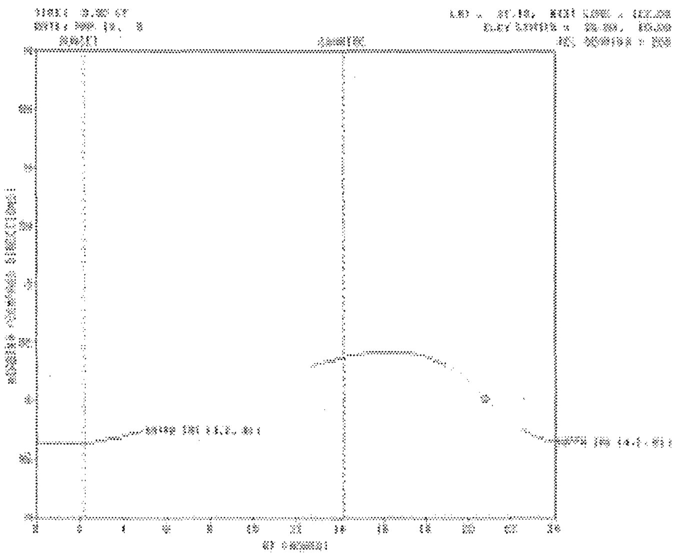

Figure 1: Visibility of an object during March 18, 2000 from Moffett Field. Note that time is given in Universal Time, and that sunset and sunrise are marked.

Many of these constraints are demonstrated in Figure 1. This figure shows visibility and heading information for an object viewed from Moffett Field during March 18, 2000. The y axis shows the heading the aircraft must fly to keep the object in view. The direction changes over time, indicating that the aircraft must constantly turn to keep the object in view. The curves on the plot indicate when the object is in view. No curve indicates the object is below the telescope's 20 degree minimum elevation, while the dotted curve indicates the object is above the 60 degree maximum elevation. Notice that the object passes below the minimum elevation and then returns to view then passes above the maximum elevation and again returns to view. During any 9 hour period, there are at most two windows of visibility for this object.

The Flight Environment

In this section we discuss constraints derived from the environment on the day of the flight. One important example of such constraints are airborne warning zones, commercial flight routes, and other administrative restrictions on where the aircraft can fly. Some flight environments may have fewer such restrictions;

[1]While calibrations and setup operations are not strictly observations, we represent them this way for convenience.

for example, if the aircraft flies out of Hawaii or New Zealand, there will be fewer such restrictions than flights over Nevada. Bad weather may constrain observations as well. While cloud cover is usually not an issue at the altitudes where observing is likely to occur, turbulence can affect the performance of the observations, and may increase observation time or have other effects on flights. Wind speed and direction can also have an effect on a particular flight. The aircraft's ground speed is directly affected by wind, and wind patterns change over time. Flight planners must take these effects into account when doing planning.

The Objective Function

The final component of the SFP is the objective function, which is used to compare two candidate flight plans. Within the confines of the single flight planning problem, the objective function can range in complexity. A good flight contains as many high-priority observations as possible; hence a good objective function might be to simply sum the priorities of the observations which are performed. Astronomers may also *prefer* rather than require that observations be done at various water vapor levels, that targets be observed when they are far from the moon or other heavenly bodies, and so on. All of these preferences can then be added to the objective function, resulting in a fairly complex measurement of the quality of a flight plan.

The Statement of the Problem

With all the components in place, we can now state the SFP: given a set of observations to perform, a date to perform them, a description of the environment on that date, and the objective function, select a take-off time, a subset of the observations, a start time for each observation, and specify any dead-legs (i.e. flying without observing). The resulting flight plan must not violate any of the constraints and should optimize the objective function. Figure 2 shows an example of a flight plan.

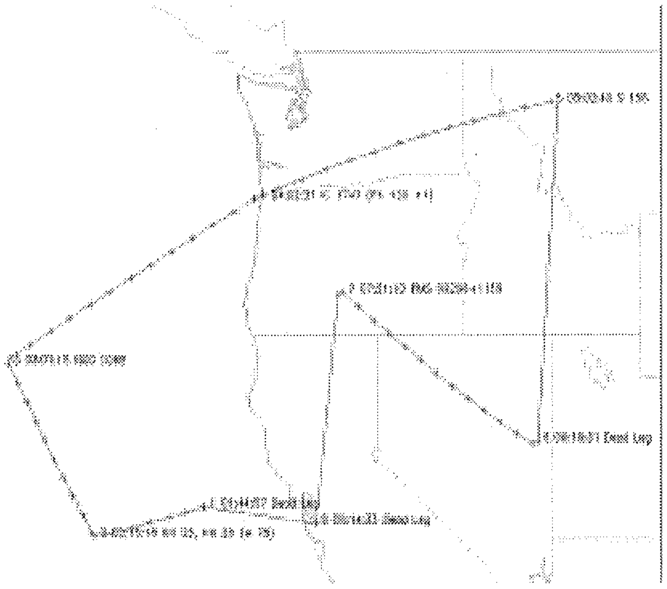

Figure 2: A flight plan. Each observation leg is marked with +s and labeled, while dead legs are simple lines. Note that no restricted areas are indicated in this figure; this flight plan is likely to cross restricted areas over Nevada.

The Complexity of Flight Planning

Many efficient scheduling techniques rely on special encodings of problems in order to deliver good computational results. These techniques work only for simple constraints, such as equality and inequality or simple combinations of resource and precedence constraints. For example, many clever encodings and algorithms exploiting them can be found in a recent text on scheduling techniques (Brucker 1998). While these techniques are very efficient, they have limited application, and a great deal of sophisticated modeling may be necessary to pose problems in the correct form.

As we have seen, any instance of the SFP is composed of a large number of complex, heterogeneous constraints over both continuous and discrete variables. Even relatively simple versions of the SFP contain geometric constraints, precedence constraints, mutual exclusion constraints and temporal constraints, all in the same problem. While it may be possible to encode parts of the problem in order to take advantage of efficient algorithms, it is unlikely that we can find a good encoding which will serve for all instances of the problem. Furthermore, over the lifetime of the observatory, other constraints may be required to represent a flight planning problem. For example, new instruments and newly discovered objects may impose new constraints. The addition of these constraints may invalidate any specialized representations and algorithms.

The SFP presents another challenge. In particular, it is not known beforehand how many observations will be performed on any given flight, nor is it known how many dead-legs will be required. Each choice made during the planning and scheduling of the flight may explicitly create other choices; for example, a decision to observe an object at a particular time and place could be preceded by a dead leg to reach that position, or observing another object beforehand. Traditional scheduling techniques require *all* of these choices to be explicitly represented, along with constraints which are triggered once the choice is made. Encoding all of this information results in large representations. The problem only becomes worse as the scope of the problem grows; thus, while it may be possible to encode a small SFP, it may not be possible to encode a larger one because the encoding will not fit into the computer's memory.

Automatic Generation of Flight Plans

In this section, we describe an approach to representing and solving the SFP. This approach is based on the Constraint-Based Interval Planning (CBIP) paradigm, which is described in (Smith, Frank, & Jónsson 2000) and which was successfully employed in the Remote Agent Planner (?).

Constraint Satisfaction and Optimization Problems

Constraint Satisfaction Problems or CSPs are an approach for representing many problems, including scheduling problems (both (Beck & Fox 1998) and (Nuijten 1994) both represent and solve scheduling problems as CSPs). A CSP consists of a set of variables, each of which has an associated domain of legal values it can take on in a solution, and a set of constraints, which restrict the legal assignments to sets of values. These constraints can be extensional, in which all the legal assignments are listed, or intentional, in which constraints are written as a mathematical relationship. A constraint is *satisfied* by an assignment if the values assigned to the variables are permitted by the constraint. A *solution* to a CSP is an assignment of each variable to a single value in its domain such that all the constraints are satisfied.

CSPs contain no mechanism to express a preference between two solutions that satisfy all of the constraints. A *Constraint Optimization Problem* or COP is a CSP which includes a mapping from a solution to the real numbers. This mapping encodes the preferences between solutions which satisfy all the constraints. It should be clear from the discussion of the SFP above that we can pose the SFP as a COP.

The representation of a CSP may be unwieldy due to the large number of constraints required to encode the conditional effects of all of the choices. An alternative representation is the *Dynamic Constraint Satisfaction Problem* or DCSP. A DCSP is a sequence of CSPs, in which each CSP is a modification of the previous CSP in the sequence. A CSP C is said to be a *relaxation* of a CSP D if C has fewer constraints, fewer variables or more combinations of assignments permitted in its constraints and a *restriction* if C has more constraints, more variables or fewer combinations of assignments permitted in it's constraints. These ideas are formalized in (Jónsson & Frank 1999). DCSPs provide a way to formally characterize how a problem changes over time, and require less space since the impact of choices need not be encoded in a single representation of the problem.

Procedural Constraints

The notion of a constraint is very general, and many real-world problems can be represented very naturally with simple constraints. However, sometimes representations of problems using simple constraints can become large and unwieldy. For instance, an enormous amount of space is required to explicitly represent all of the possible time-location pairs when an object is visible. It is often more efficient to represent a constraint with a mathematical relation such as $\leq$. *Procedural constraints* (Jónsson 1996) generalizes this concept by formalizing the notion of a procedure which enforces a relation among the variables of a constraint. A special form of procedural constraint called an *elimination procedure* is permitted to get rid of any element of a domain that is provably not part of any solution to the problem, given the information currently at hand. For example, an elimination procedure might eliminate all observations as candidates for the next observation on a flight because the aircraft is almost out of fuel. Procedures also are used to formalize the concept of *decision variables.* If a procedure is able to assign values to a set of variables $V - D$ given a consistent assignment to the variables in D, then there is no reason to search the values of variables in $V - D$. The variables in D are called the decision variables, since those are the only variables which require making decisions. Continuous variables can be handled by making sure they are not in the set of decision variables (Jónsson & Frank 1999).

Constraint-Based Interval Planning

CBIP is a general planning framework that uses a model to specify the domain in which the planning is to take place. The elementary notion in this framework is that of an *interval*, which denotes an activity or state which is maintained over a period of time. Each interval is described using a set of DCSP variables. The temporal aspects of a interval are described using variables that represent the start time, the end time, and the duration of the interval. Actions and states can have parameters that further specify the action or fluent, so the state value described by the interval is also described by a set of DCSP variables. An interval consists of the name of an action or state, and some parameters of that action or state. The model consists of *planning schemata*, which specify relations between intervals, enforcing preconditions, effects, enabling conditions, mutual exclusions, etc. These schemata then give rise to constraints between the different DCSP variables that describe the different intervals. A set of connected intervals, i.e, a candidate plan, gives rise to an instance of a constraint network. We employ procedures as described in the previous section to enforce some of the more complex constraints.

The CBIP Representation of the SFP

In this section we describe how to represent the SFP in the CBIP framework [2]. Let us assume that the problem consists of a set of observation requests, and our task is to construct a flight for a particular day such that the sum of the priorities of the observations performed exceeds a certain threshold. The durations of the observations are fixed, but we will permit dead legs in this problem. For simplicity, we ignore restricted zone constraints and artificially restrict the bearings of dead legs to the 4 cardinal directions, and restrict dead leg flight duration to 5, 10, 15 or 20 minutes.

Let us suppose that as part of the SOFIA domain we want to represent calibration operations for the instruments on board the aircraft. We would then write a planning schema for a Calibrate action. In the schema below, variables are preceded by a ?. The schema might look like this:

Calibrate(?loc,?Target,?Instrument,?NextOp):-
$CalibrateTime(?dur, ?Target, ?Instr)$
$LocationAfterCalibrate(?loc, ?endloc, ?dur)$

[2]We have considerably simplified the presentation, and the resulting flight plans would require post-processing before final approval.

contained-by	Status(?Instr,ON)	
contained-by	Calibration(?Target)	
contained-by	Observable(?Target)	
meets	Calibrated(?Instr)	
Eq(?*Next*, *Obs*) $\rightarrow$	meets	Observe(?any,?endloc,?any)
Eq(?*Next*, *DLeg*) $\rightarrow$	meets	DeadLeg(?endloc,?any)

The line *CalibrateTime*(?*dur*, ?*Target*, ?*Instr*), indicates that the instrument, duration of the operation, and target used to perform the calibration are mutually constrained. Subsequent lines specify temporal relationships between the calibration event interval and other intervals. For example, contained-by Status(?Instr,ON) indicates that the instrument must be turned on for an interval encompassing the calibration interval. Finally, the last two lines specify disjunctions over intervals that may be related to the calibration event interval. For example, the value of the ?NextOp variable indicates whether an observation interval or a dead-leg interval follows the calibration interval.

As planning decisions are made, variables and constraints arising from new intervals are added to the constraint network, and variables and constraints stemming from intervals that are no longer part of the plan are removed from the constraint network. Figure 3 shows how a partial plan translates to an induced DCSP.

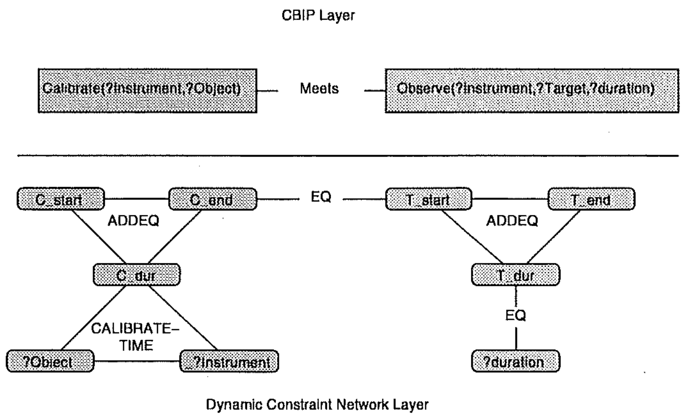

Figure 3: A partial plan and the induced DCSP. As commitments are made to steps in the plan, new intervals are generated and new variables and constraints are added to the DCSP.

At this point we mention how domain specific constraints are integrated seamlessly into the description of the planning domain. For example, *CalibrateTime*(?*dur*, ?*Target*, ?*Instr*) may be a very complex relationship between the duration, target, and instrument. However, once *CalibrateTime* is written as an elimination procedure, it can be specified as part of a plan axiom as described above. This gives modelers the flexibility to decide how sophisticated and powerful an elimination procedure they wish to provide. We also observe that it is now possible to make restrictions such as those we demand on consecutive dead legs. In this case, we would simply write into the plan schema for dead legs that they must be followed by an observation leg. As a final point, we observe that it is relatively straightforward to write a procedural constraint which handles the flight plan threshold criteria; as each new observation leg is posted, we can update the inputs to the optimization function and compute the new value of the flight plan.

Finding Plans

Finding plans in the CBIP framework involves solving the DCSP which is created by specifying the initial state of the particular planning problem. This is normally accomplished using *backtracking search*, which constructs a solution to a DCSP by selecting a variable from the remaining unassigned variables in a problem, then trying each possible value in turn. If at any point a constraint violation is detected, the procedure returns to the previous variable binding, and tries another value. If all the values of a variable are tried without success, then the procedure also returns to the previous variable and tries another value.

We pose a particular instance of the SFP by specifying an interval for each observation which could be performed on a particular night, and specifying intervals reflecting the takeoff and landing legs of the flight plan. The commitments to variable bindings will establish which observations are made, in order, after takeoff and throughout the flight. Plan axioms can specify that certain intervals exist, but this requirement can be satisfied by either creating a new interval, or making use of a pre-existing interval in the current plan; these choices are also reflected as variable binding decisions. For instance, suppose that the planner must decide what to do about an observation after a calibration. Either the planner can insert a new observation into the plan, or determine that the observation following this calibration is some observation already in the plan, thereby satisfying the axiom. To ensure that the planner does not try and make all observations, we permit the planner to "disable" an observation by eliminating it's interval. The optimization criteria will prevent the planner from creating a legal flight plan with no observations.

This conceptually simple algorithm is guaranteed to solve a DCSP or demonstrate that no solution is possible. The worst-case running time for this procedure is the product of the sizes of the domains of all the variables, which is exponential in the number of variables. Consequently, backtracking is only possible for DCSPs with discrete domains. Backtracking can be easily modified to solve DCOPs by saving the value of the best solution found, and searching over all solutions instead of halting after finding the first solution satisfying the constraints. In essence, this is like imposing a new bound on solution quality each time the old bound is improved upon. Figure 4 shows a simple backtracking algorithm for solving the SFP by complete search. Note the step in which the next CSP is generated; this is accomplished by consulting the plan schemata.

The performance of this algorithm depends dramatically on functions which select the next variable to choose, select the order in which to try values, perform fast inference to eliminate values from the domains of unbound variables, and decide which variable binding decision is responsible for a constraint violation. For instance, in the SFP, there is often

```
procedure FindSFP($P$)
  generate next CSP, $P'$
  if a constraint is violated return fail
  if problem solved return success
  select an uninstantiated variable $V$
  for all values of this variable $v \in dom(V)$
    if FindSFP($P' \cup V = v$) = success
      return success
  end for
  replace $P'$ with $P$ if necessary
  return fail
end
```

Figure 4: Finding a simple flight plan.

a choice concerning which observation to make for a given observation leg. Since the goal is to exceed a bound on the priority, a good choice might be to select the remaining observation with the highest priority to try next. However, if this observation is too long or takes the aircraft in the wrong direction, other observations will not be possible and a poor quality flight plan will result. Consequently, modifying this choice by checking the direction the aircraft must fly may lead to better flight plans in less time. It should also be clear that using procedural constraints to eliminate bad choices for variables can save time, since the algorithm does not need to guess these values. These modifications are critical to good algorithm performance; for results on traditional scheduling problems, see (Beck & Fox 1998) and (Nuijten 1994).

It should be clear from the discussion of the constraints that flight planners must analyze several tradeoffs when scheduling flights. For example, an obvious tradeoff concerns whether to try a high-priority observation first, or to try an observation of lower priority which may be easier to schedule. Another tradeoff concerns when to schedule a particular observation. If the observation is constrained by a minimum water vapor threshold, for instance, then this may be satisfied by flying higher or further north (Horn & Becklin 2000). However, these both require making observations later in the flight, and care must be taken to ensure that the aircraft can still return home. Tradeoffs such as these drive the construction of both variable and value ordering heuristics, which are necessary to ensure good algorithm performance.

Related Problems

The SFP is similar to other problems important for both NASA and industry. For example, the Vehicle Routing Problem (VRP) is the problem of delivering packages to various destinations in an urban area(Kilby, Prosser, & Shaw 1999). This problem does not have the complex geometric constraints that the SFP features, but ordering constraints, package size, fuel constraints, truck capacity and distances all interact to make for a complex scheduling problem. Scheduling operations for planetary rovers and interplanetary vehicles requires sequencing science observations and their enabling activities such as sample acquisition, as well as operations to aid in navigation as well as re-charging operations. Furthermore, the total number of operations must be within the available power budget of the vehicle. Finally, path planning and astronomical navigation feature complex geometric constraints which may require simulation.

Conclusions

The flight planning problem motivated by the SOFIA GI program features many heterogeneous constraints over a mixture of continuous and discrete variables. The resulting problem is further complicated by the fact that the problem includes an uncertain number of steps and other conditional constraints that can be very expensive to encode in a single problem instance. These factors lead us to the conclusion that traditional scheduling techniques which solve simple scheduling problems are, by themselves, likely to be inadequate. We show how to represent the problem as a DCSP with procedural constraints. This problem can then be solved by a complete search algorithm using the procedural constraints to efficiently eliminate assignments. Only by doing so can we ensure that SOFIA's choice will be made correctly.

References

Beck, J. C., and Fox, M. 1998. A generic framework for constraint-directed search and scheduling. *AI Magazine* 19(4):101–130.

Becklin, E. E. 1997. Stratospheric observatory for infrared astronomy. *Proc. of ESA Synmposium* The Far Infrared and Submillimetre Universe, *ESA SP-401* 201–206.

Brucker, P. 1998. *Scheduling Algorithms.* Springer-Verlag.

Erickson, E. F., and Davidson, J. A. 1997. Sofia: The future of airborne astronomy. *Proc. of the Airborne Astronomy Symposium on the Galactic Ecosystem: From Gas to Stars to Dust* 73:707–732.

Horn, J. M., and Becklin, E. 2000. Optimized flight planning for sofia. In *Proceedings of the SPIE*, volume 4014.

Jónsson, A., and Frank, J. 1999. A framework for dynamic constraint reasoning using procedural constraints. In *Proceedings of the Workshop on Constraints and Control, held in conjunction with the 5th International Conference on Principles and Practices of Constraint Programming.*

Jónsson, A. K.; Morris, P. H.; Muscettola, N.; and Rajan, K. 1999. Next generation remote agent planner. In *Proceedings of the Fifth International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS99).*

Jónsson, A. 1996. *Procedural Reasoning in Constraint Satisfaction.* Ph.D. Dissertation, Stanford University, Stanford, CA.

Kilby, P.; Prosser, P.; and Shaw, P. 1999. A comparison of traditional and constraint-based heuristic methods on vehicle routing problems with side constraints. *Journal of Constraints, to appear.*

Nuijten, W. 1994. *Time and Resource Constrained Scheduling: A Constraint Satisfaction Approach.* Ph.D. Dissertation, Eindhoven University of Technology.

Smith, D.; Frank, J.; and Jónsson, A. 2000. Bridging the gap between planning and scheduling. *Knowledge Engineering Review* 15(1).