REFERENCES

[1] Adorf, H.-M., and di Serego Alighieri, S. 1989: "An Expert Assistant Supporting HST Proposal Preparation," in Data Analysis in Astronomy III, ed. V. Di Gesu, Plenum Press.

[2] Adorf, H.-M., M. Johnston "A Discrete Stochastic 'Neural Network' Algorithm for Constraint Satisfaction Problems", H.-M. Adorf and M.D. Johnston, in "Proceedings of the International Joint Conference on Neural Networks (IJCNN 90)", (San Diego, June 17-21 1990), (Piscataway, NJ: IEEE), Vol. III, pp. 917-924.

[3] Dechter, R., J. Pearl (1988). Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence*, 34:1-38.

[4] Gerb, A. "Transformation Reborn: A New Generation Expert System for Planning HST Operations", A. Gerb, to appear in the Proceedings of the 1991 *Goddard Conference on Space Applications of Artificial Intelligence*.

[5] Hall, D., ed. (1982), The Space Telescope Observatory, NASA CP2244.

[6] Johnston, M., and G. Miller, 1990. "Artificial Intelligence Scheduling for NASA's Hubble Space Telescope", "Proceedings of the Fifth Annual Expert Systems in Government Conference" (Washington, May 7-11, 1990), ed. B. Silverman, V. Huang and S. Post (Los Alamitos, IEEE Computer Society Press), pp. 33-39.

[7] Johnston, M., 1990. SPIKE: AI Scheduling for NASA's Hubble Space Telescope," in Proceedings of the Sixth IEEE Conference on AI Applications (Santa Barbara, March 5-9, 1990), (Los Alamitos, CA: IEEE Computer Society Press), pp 184-190.

[8] Keene, S (1989). Object-Oriented Programming in Common Lisp, Reading, MA: Addison-Wesley Publishing.

[9] Lucks, M., 1990. “A Knowledge-based Framework for the Selection of Mathematical Software”, Ph. D. dissertation, Dept. of Computer Science and Engineering, Southern Methodist University, 1990.

[10] Miller, G. and Johnston, M., 1991. "A Case Study of Hubble Space Telescope Proposal Processing, Planning, and Long-Range Scheduling," to appear in the Proceedings of the AIAA Computing in Aerospace 8 Conference (October 22-24, 1991, Baltimore, MD).

[11] Minton, S., M. Johnston, A. Philips, and P. Laird, 1990. "Solving Large-Scale Constraint Satisfaction and Scheduling Problems Using a Heuristic Repair Method" in "Proceedings of the Eighth National Conference on Artificial Intelligence" (Boston July 29-August 3, 1990), (Menlo Park, CA: AAAI Press), pp. 17-24.

[12] Sponsler, J. (1989). Genetic Algorithms Applied to the Scheduling of the Hubble Space Telescope, in "Proceedings of the 1989 Goddard Conference on Space Applications of Artificial Intelligence" (Greenbelt, May 16-17, 1989), ed. J. Rash, NASA Conference Publication 3033, (Greenbelt: NASA), pp. 391-402, reprinted in Telematics and Informatics, 6, pp. 181-190.

[13] Steel, G. (1990). Common lisp the language. Digital Press.

[14] Texas Instruments 1990. The Explorer System Software Manuals; Window System Reference. Publishing Center (Texas Instruments Incorporated, 1987), pp 15-1.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

on a 6 month planning interval. Batch pre-processing (TRANSFORMATION, etc.) of the 176 proposals was done and this required approximately 70 hours of wall clock time. The time required by SPIKE to generate a single schedule with no conflicts was on the order of two hours. Such a schedule is believed to satisfy all temporal and spacecraft constraints; however, the schedule is not necessarily the best schedule possible.

Planned Projects

The SPIKE system will be augmented in the near future by software that will constrain the scheduling of moving targets (e.g., a planet). Explanation tools are planned also. Currently, the GUI provides the suitability functions for visual analysis but often there are very many of these and their interactions are complex. A facility is invisioned where one would select a region of time and ask SPIKE "Why?" in order to gain an understanding of the information (such as zero suitability) displayed.

The current version of SPIKE includes an internal SU tracking database and supports queries that can indicate where an SU lies in the pipeline (e.g., "SU-1 has been scheduled by SPSS"). This tracking ability will be enhanced within the next year and will be used to provide such information to other groups within the STScI and to the observers.

AI technology has facilitated the design and development of the Spike system. Symbolic processing, object-oriented programming, and constraint satisfaction algorithms have served to make a seemingly intractable problem manageable. The goal of increasing the scientific efficiency of the Space Telescope will continue to guide the development of the SPIKE system as the mission progresses.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

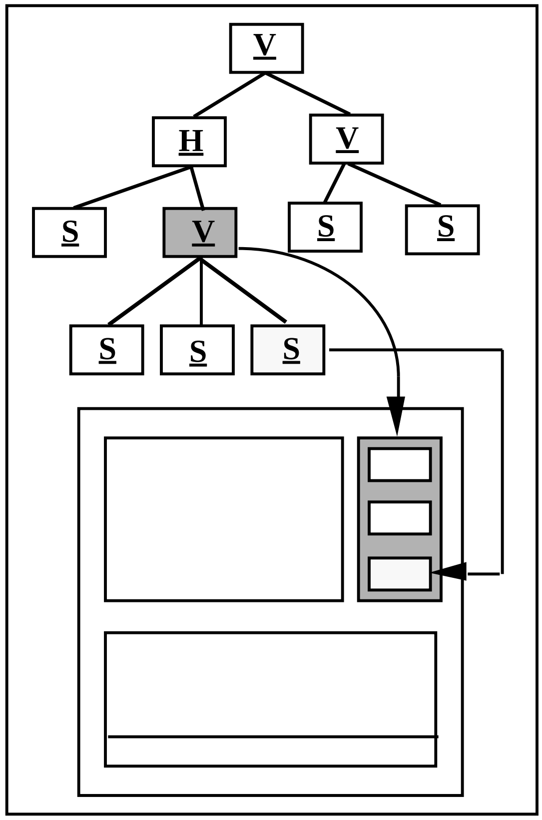

Figure 8. A complex window can be represented by a hierarchy of frame objects in memory. Vertical frames (V's) and horizontal frames (H's) by definition govern the geometry of their children. Simple frames (S's) are terminal in the tree. Arrows illustrate the correspondence between nodes and windows.

Support is included in the design for various operations that may be applied to frames. The operations include mouse operations (enter-window, leave-window, button-press, button-release), frame operations (expose, deexpose), graphics (draw-line, draw-rectangle, draw-circle), and text processing (input and output).

At a very low level in the frame management software a frame maps to a specified window system. Currently, two such systems have been integrated into the LAWS format: Explorer Windows (native to that machine) and CLOS-X. The latter system is an object oriented system that hides the details of CLX (Common Lisp X Windows) which in turn conform to the X11M network windowing protocols.

7.CONCLUSIONS

The SPIKE long-term scheduling system has been described in this report. SPIKE is used to create schedules for the Hubble Space Telescope. It analyzes the constraints on exposures and provide a manual and automatic facility for scheduling. Since SPIKE does not schedule with second by second precision, it can within reasonable limits of accuracy place SUs on a months long calendar. Week-long time segments of schedule are passed to SPSS which schedules the SUs to precise times within the week. This data transfer only occurs a few weeks in advance of actual execution time and so accuracy is guaranteed in the short term schedule. Since SPSS need only work with a subset of SUs, its processing time is minimized as well.

Performance Test Results

Timing tests have been done to assess the throughput of STScI scheduling systems. The general observer pool (the first group of proposals that have been submitted by astronomers who are outside of the Space Telescope project) were scheduled

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

individual criteria, or *features*. Some of these features are qualitative in nature, i.e. they are either completely satisfied or completely unsatisfied by a primary-parallel pair. An example is the feature *instrument-compatibility*, which is satisfied if the primary and parallel require instruments that may be utilized simultaneously, and unsatisfied otherwise. Other features are quantitative, i.e. they assume values on a graded, continuous scale. For instance, the *required-roll* feature describes how much the spacecraft must be rotated from the orientation needed for the primary, in order to place the aperture of the parallel instrument on its target. (The lesser the roll, the better the match.) The presence of such features complicates the ranking problem, because it introduces quantitative tradeoffs into the assessment.

A *knowledge-based multi-criteria decision network* is utilized in SPIKE to estimate the aggregate compatibility of a primary SU with a generic parallel SU. The network is based on a model described in [9], in which qualitative and quantitative features are represented uniformly by expert-supplied functions that express the contribution of each feature to the overall assessment. These functions, as well as the formula used to aggregate the individual features, are explicit units of knowledge that may be modified incrementally, and which support explanation facilities.

6.2.LAYERED ABSTRACT WINDOW SYSTEM

In the operating environment of the STScI, a number of machine types are available and it is the intention of the SPIKE development team to support the software on many platforms. Included are Explorers, SPARCStations, and MacIntoshes. To fully support our window oriented graphical user interface (GUI), the Layered Abstract Window System (LAWS) is been designed and prototyped. The important attributes of this system include the following:

1. Independence from specific window systems
2. Independence from specific machine types
3. Event Processing (e.g., mouse clicks) is supported
4. Object Oriented Design
5. Automatic geometry management

The design is based on the *frame* class. An instance of a frame records the details of a single abstract (machine-independent) window. The term frame is borrowed from Lisp Machine software technology[14]. The attributes of a frame include x, y, width, height, title, and so forth. The subclass *complex frame* is a frame that may have other frames as children. This supports the notion of complex windows (such as a menu that contains buttons). To support the hierarchical breakdown of a complex window in a rectilinear scheme, the subclasses vertical frame and horizontal frame have been defined (Figure 8). A *vertical frame* is intended to have its child frames stacked vertically. The children of a *horizontal frame* are positioned side by side. From these basic concepts, one may specify arbitrarily complex window arrangements. Methods to automatically compute the geometry (x, y, width, and height) of a frame hierarchy have been defined and can be extended in order to program specialized arrangements of frames.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

5.1.Spike Automated Testing Tool

The Spike Automated Testing Tool (SPATT) is a structured environment for testing LISP software. In SPATT, each *test* is a formal CLOS object with various attributes including: a piece of Lisp code called a *test-form* that exercises some functionality of SPIKE; a Lisp form called a *test-output* that is generated by the test; and a set of Lisp-forms called *analysis functions* that check the validity of the test-output. Analysis functions provide a uniform mechanism for various forms of testing, including regression testing, aliveness of the code, datatype correctness and value correctness. For instance, an analysis function might invoke the SPATT system function “compare-to-benchmark” which looks up the benchmark from a previous test and compares it with the new result. Alternatively, an analysis function might invoke a user-defined check on the integrity of memory-resident data-structures.

Tests are grouped into *test-suites* that exercise a particular SPIKE subsystem, such as the astronomical events calculation code or the orientation constraint routines. A *test-session* is a set of test-suites executed sequentially in the same Lisp environment. At run-time, the analysis of each test (including diagnostic information regarding any erroneous results) is summarized in a file which may be inspected by the system developers or testers.

Some of the novel features of SPATT are:

1. Test results for each test, as well as benchmark results, are stored in a Lisp-readable format, thereby permitting the structure of the results to be recreated in a Lisp environment. Hence, when a discrepancy is detected between a new result and its corresponding benchmark, the full power of Lisp may be used to identify the exact location of the discrepancy and to determine whether the difference is significant. The analysis may be performed automatically via user-defined extensions to the system benchmark-comparison facility. This approach is a useful enhancement of conventional regression testing, where benchmark results are stored in semantic-free ASCII files and where comparisons are performed manually.

2. Execution errors that would ordinarily cause the system to crash are trapped using the facilities of the Common Lisp Condition System[13]. When such an error occurs during testing, the location of the error is noted, relevant information (including the function call stack at the time of the crash) is recorded, and SPATT proceeds with the remainder of the test-session.

(3) Multiple tests that repeat the same test form with variations only in the input arguments may be specified generically and then generated automatically by SPATT. The test author must specify the Lisp function name that executes the test-form, a range of permitted values for the input arguments, and the maximum number N of tests to be generated. From this template, SPATT creates N tests that satisfy the specification. Random generation of argument data is also supported.

The SPATT environment is integrated into our delivery process to verify our software. A side effect of writing structured test suites is that SPIKE has been more carefully scrutinized and flaws revealed.

6.WORK IN PROGRESS

In the following sections, subsystems of SPIKE currently under development are described.

6.1.PARALLEL SCIENCE MATCHING SYSTEM

The presence of multiple instruments on the HST creates opportunities for simultaneous use of different instruments. In some cases, observers desire to execute an exposure in parallel with some other specified task. Such a request is called a *coordinated parallel exposure.* Transformation will include the parallel exposure in a primary SU and SPIKE will process the primary as usual. No special action is required of SPIKE related to the coordinated parallel. Proposers are also permitted to request *pure parallel exposures*, which ask to perform a task at any opportune time when some (unspecified) primary SU is executing. SPIKE cannot actually schedule pure parallels because certain data, such as tape readout conflicts, are not available at the long-term planning stage. SPIKE can perform an important advisory function, however, by finding and ranking (to the best of its knowledge), primary SUs that are candidates for simultaneous execution with particular SUs. These rankings are then used to advise the short-term planning team of the best potential pairings.

Evaluating the compatibility between a particular primary and a particular pure parallel may be viewed as a multiple-criteria decision problem, in which the overall evaluation requires aggregating the compatibility of the two SUs with respect to many

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

2. RCMC (random choice/minimum conflicts) Initial Guess Algorithm. This algorithm selects a variable randomly and then assigns it to the value from its domain with the fewest conflicts.

As a result of each such assignment, constraints are analyzed (via the Constraint Processing subsystem) and conflict counts are updated. These greedy methods may produce a reasonably good schedule although generally there will be assignments that possess conflicts. Since these conflicts represent violations of HST constraints, it is necessary to eliminate them if possible. This is the purpose of the repair step.

Heuristic Repair Step

Two of the possible algorithms employed to repair schedules operate in the following manner.

1. MCMC (maximum conflicts/minimum conflicts) Repair Algorithm. This algorithm selects the variable that has the most conflicts and re-assigns it to the value from its domain that has the fewest conflicts.

2. RCMC (random choice/min conflicts) Repair Algorithm. This algorithm randomly selects a variable and re-assigns it to the value from its domain that the fewest conflicts.

4.7.RESOURCE CONSTRAINTS

In the CSP system, resource constraints are monitored by keeping track of how much of an available resource is consumed for each discrete segment of time (a week). When the resource for one such segment is consumed entirely a conflict is logged for all variables for that segment's associated value.

4.8.SCHEDULES DELIVERED TO SPSS

Once a complete long-term schedule is created, the scheduling decisions made are logged into a SPIKE database. When it is required that such decisions be communicated to the SPSS system, a delivery for a specific week is made. The delivery contains the following information:

1. A list of SUs committed for the week.

2. For each committed SU, the scheduling window as determined by SPIKE. It is the case that this time interval may be shorter than a week (due to constraints on the SU).

3. For each committed SU, the required orientation angle range (as determined by interpretation of relative orientation constraints). SPIKE must keep track of this long term constraint link (currently SPSS does not).

Feedback from SPSS updates the SPIKE database and includes the precise time of scheduling. If an SU is not found to be schedulable by SPSS, this information is communicated to SPIKE as well. Such an SU is placed back into the pool of SUs to be scheduled by SPIKE in other weeks.

5.SOFTWARE ENGINEERING

The SPIKE system has been implemented in Common Lisp[13] and the Common Lisp Object System (CLOS)[8]. The use of object classes, inheritance, and specialized methods facilitates the creation, maintenance, and extension of our software. The Lisp language provides symbolic processing capabilities that support rapid program development.

The development and delivery machine has been the Texas Instruments Explorer II. The software also runs on Sun Microsystems SPARCStations under Allegro Common Lisp.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

4.5.MODES OF OPERATION

SPIKE supports several operating modes, two of which are Analysis Mode and Scheduling Mode.

Analysis Mode

In this mode, the constraints on each exposure are available for viewing via a graphical user interface. Users may then see the effects of each constraint in order to find out what has caused an exposure or an SU to be unschedulable.

Scheduling Mode

In this mode, for a given exposure all absolute constraints are combined into a single suitability function and all such functions are combined for each exposure in an SU. The result is that there is now a single suitability function that represents the absolute constraints for an SU. The complexity of processing this data is reduced (relative to Analysis Mode) for the following reason:

Proposal Data Compression

The pre-scheduling processing of a proposal (transformation, absolute constraint analysis) results in compressed proposal data. Specifically, the B-Functions are stored and so can be re-used later without having to recompute them. Also, the absolute constraints for each SU are stored as a single suitability function instead of storing them by individual component exposure. Since some of these constraints (most notably, absolute orientation) are expensive to calculate, storing these intermediate results provides a time-savings later in the scheduling process.

4.6.CONSTRAINT SATISFACTION PROBLEM SCHEDULING SUBSYSTEM

The definition of a good schedule includes completeness (i.e., were all scheduling units scheduled?) and overall suitability (i.e., in order to fit all SUs onto the schedule, is the summed suitability of the schedule lower than if the completeness requirement was relaxed?). The search for a good schedule by the SPIKE system is aided by a general purpose tool called the CSP system[11].

A CSP consists of N variables. Each variable corresponds to a scheduling unit. Each has a domain of legal values. Values are elements in a set of natural numbers (zero and above) and each value represents a week (i.e., time is discretized). A set of constraints limits the allowed values for a variable based on the assignment of variables to values. A conflict occurs when a constraint is violated. For each variable/value pair a conflict count exists.

The CSP subsystem exists as a layer over the Constraint Processing System. Communication between the systems occurs as follows:

• The absolute suitability function of an SU in SPIKE is used to determine the domain of acceptable values for a corresponding variable. If an SU has zero suitability in a week, the associated CSP value is removed from the variable's domain.

• Variable/value assignments are analyzed using the constraint propagation system. When a commitment is made, the effects of that commitment on other clusters is determined by examining the suitabilities that have been calculated by the constraint processor. For any week that no longer is suitable, a conflict is added to the conflict count of the variable/value pair.

The CSP system is used in a two step process: initial guess and heuristic repair. These are discussed below.

Initial Guess Algorithms

An initial guess (exhaustive assignment of variables to values) is generated according to a selected algorithm. It is unlikely that a solution without conflicts would be found. Among the algorithms used to do this are the following:

1. MC (minimum conflicts) Initial Guess Algorithm. This algorithm processes each unassigned variable. For a variable, the algorithm selects the value from its domain with the fewest conflicts for that variable and an assignment is made.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

range of time (because all orientations are now legal for B). This sort of discontinuity in the mapping functions empirically defines orientation as time variant.

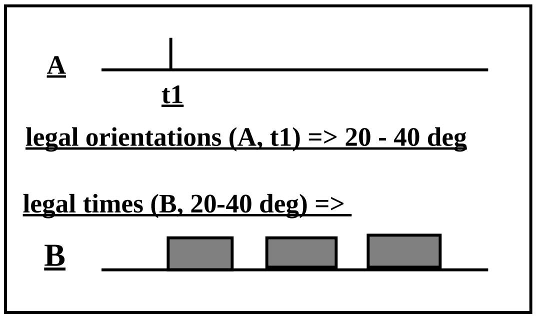

Figure 6. If exposure A is scheduled at time t1, the calculated orientation range is 20-40 degrees for example. The suitability function for exposure B for that orientation range is depicted graphically.

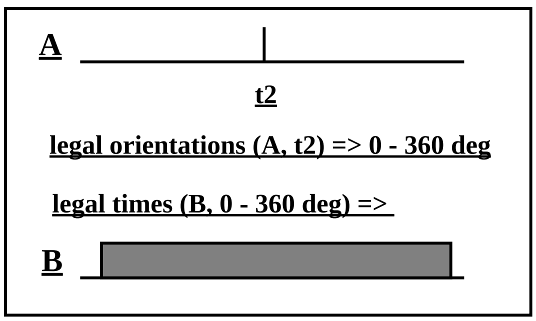

Figure 7. If exposure A is scheduled at a different time t2, the calculated orientation range is different (0-360 degrees). The suitable time range for exposure B for that orientation range is now quite different than in the scenario depicted in Figure 6.

4.4.THE SCHEDULING PROCESS

Using the SPIKE scheduling tools, one may make a scheduling decision (i.e., a *commitment*) that restricts the times when a scheduling cluster may be scheduled. A commitment is a plan to do the activity in the week.

The Spike system supports *propagation of constraints*. The underlying notion is that if a network of relative constraints exist, the effects of the constraints must be propagated through the network iteratively until all nodes have been reached and the suitability functions of all related SUs have been corrected based on that propagation. For example, let A, B, and C be SUs and let C1(A,B) and C2(B,C) be relative constraints. In order to calculate the effect of SU A on SU C via the two constraints, at least two steps must occur. The first step propagates the effect of A on B via C1. The second step propagates the effect of B on C via C2. (If the constraint links have been made path consistent, then this propagation is unnecessary.)

The overall scheduling interval (generally a year) is divided into discrete units called *segments*. The length of a segment is programmable but the operational practice has been to set segment length to be one week.

A long range schedule will consist of a number of time segments each of which will have a set of scheduling clusters that have been committed there. The commitments are to week-long segments and do not specify precise times.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

Time Invariant Constraints.

The behavior of a time invariant constraint does not change relative to absolute time. For example, the constraint "X after Y by 5 days" produces the same kind of effect for Y regardless of when X is scheduled. That is, Y can be schedulable 5 days after X. Based on this analysis, the concept of the B-function (B stands for "benefit") has been developed[7] to represent such a constraint. B-functions are similar in nature to suitability functions but are more abstract because they represent constraints without regard for absolute time. Figure 5 illustrates several B-functions and how they may be combined.

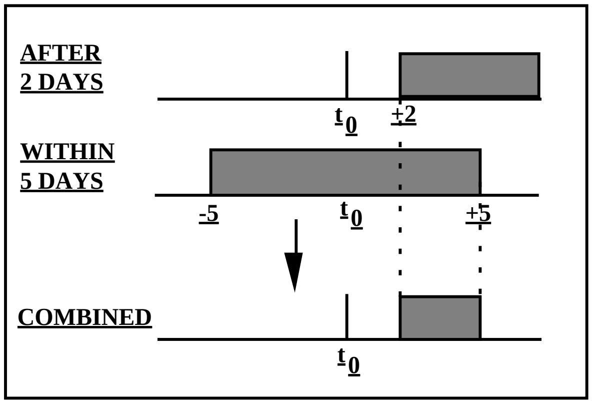

Figure 5. The graph labeled AFTER 2 DAYS illustrates a B-Function that represents the notion that it is legal to schedule the related task more than two days after the initial task. Time t_0 is a conceptual reference point that can be applied to any point in universal time. The WITHIN 5 DAYS B-Function indicates schedulability from -5 to +5 days from t_0. When combined, a new B-Function, that captures both constraints, results.

A *path consistency* algorithm[3] is executed in order to induce all possible time invariant constraint interactions. Let a_1, a_2, and a_3 be SUs, and c_1 be a constraint between a_1 and a_2 and c_2 be a constraint between a_2 and a_3. A new constraint c_3 is deduced between a_1 and a_3. This is explained in more detail in [6]. The complexity of determining all such constraints is $O(n^3)$ where n is the number of initial constraints. For certain proposals, this number is small (less than one hundred) but for others this number may be in the thousands. Although this algorithm is expensive as n becomes large, the resulting knowledge can be put to good use in the SPIKE system (where a fully path consistent network eliminates the need to propagate time invariant constraints during scheduling).

Time Variant Constraints

Time variant constraints cannot be represented as simple B-functions. Because of this, constraint networks possessing such constraints cannot be fully path consistent in the current scheme. Such networks, therefore, require some extra processing when suitabilities are being calculated. This processing involves propagating constraints (all types) until the suitabilities of all activities no longer change with successive iterations.

Time variant constraints, on the other hand, produce effects that vary as absolute time varies. The *same orientation* constraint falls into this category (see the description above). Due to spacecraft requirements on temperature and electrical power and because the orbit of HST precesses over time, this function varies over absolute time. It therefore cannot be abstracted into a B-function representation.

An informal analysis is provided here to clarify. Assume that exposures A and B must have the same orientation. To determine the suitability of scheduling B given that A has been scheduled at time t1, SPIKE calculates the times when B can achieve the orientation of A. Figure 6 illustrates the suitability that might result from such a calculation.

For some targets (ones that lie on the ecliptic plane), there may be a point in absolute time where the acceptable orientations may include a full 360 degrees (Figure 7). If such an orientation range occurs, then the suitability of B may cover a broad

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

4.3.Time Invariant and Time Variant Constraints

Relative constraints, such as *after*, fall into two categories: *time invariant* and *time variant constraints*. These two classes are handled differently by SPIKE.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

The resource constraints, in general, track the availability and usage of finite HST resources and include the following:

Data Volume: This constraint keeps track of how much data will be produced by the scientific instruments per unit time.

Scheduling Unit Duration: This resource maintains a record of the total amount of telescope time that will be consumed by the SUs that are assigned to a unit of time.

Real-Time Contacts: A certain fraction of observing time is available for astronomers to interact with the telescope in real-time. This constraint maintains of a record of this.

4.2.Knowledge Representation

SPIKE employs the *suitability function* as the fundamental knowledge representation. The suitability function provides a means for representing scheduling constraints and preferences over time[7]. The approach is numeric and provides a way to represent the concept of "goodness over time." Figure 3 illustrates a suitability function.

The suitability function for a specific constraint is determined by analyzing the particular constraint. Sun Exclusion suitability generation, for example, works by sampling over time. At each time point, the geometry of HST, target, and sun are checked and if sun exclusion is violated the suitability for that time is zero. The suitability is one if it is not.

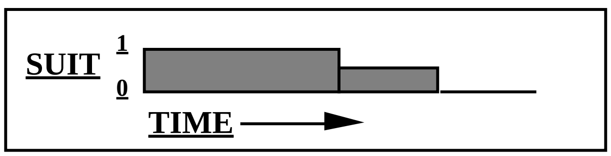

Figure 3. A suitability function describes goodness (from 0.0 to 1.0) over time. A zero value indicates an unsuitable (i.e., unschedulable) time. The height of the suitability graph indicates the suitability at each time point.

The suitability functions are combined for all constraints on an exposure by multiplying them (Figure 4). This is the fuzzy logical analog to the *and* operation and has these properties: any zero value produces a zero result; two suitability values that are less than one will amplify producing much lower suitability. Similarly, the suitabilities of component exposures are combined (using the geometric mean) to determine the suitability function of a scheduling unit. This operation preserves the zero logic result but does not allow relatively low suitabilities to pull the results down for the containing SU.

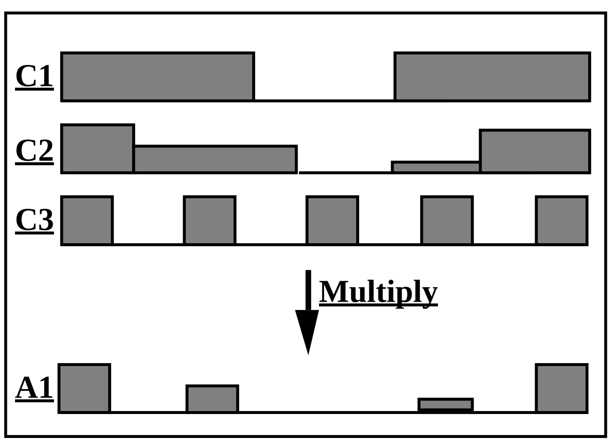

Figure 4. The constraints (C1, C2, C3) on an scheduling unit A1 are represented as suitability functions. Multiplying the suitabilities (applying logical *and*) gives the suitability of the SU.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

A Constraint Processing Subsystem processes data that describe targets, exposures, scheduling units, and constraints. It contains methods that can calculate the effects of the constraints.

4.1.HST SCHEDULING CONSTRAINTS

There are three types of constraints on HST: *absolute constraints*, *relative constraints*, and *resource constraints*.

Absolute Constraints

Absolute constraints describe some restriction or preference regarding a single exposure. A subset of these constraints is listed below.

Sun Avoidance: The angle from the V1 axis (see Figure 2) of the telescope to the sun can be no less than 50 degrees.

Moon Avoidance: The angle from the V1 axis to the moon can be no less than 15 degrees.

Zodiacal Light: Discourage scheduling exposures when there is a lot of light that is scattered from dust in space.

Orbital Viewing: A complex constraint that includes the following logic: prevent scheduling an observation when the duration of target visibility (i.e., when the target is not occulted by the earth) is less than the exposure duration.

South Atlantic Anomaly (SAA): Discourage scheduling exposures when the telescope's orbit passes through the SAA where the Van Allen radiation belts dip unusually low.

Relative Constraints

Relative constraints describe the relationship between two or more exposures. Examples include:

AFTER: Schedule exposure B after exposure A.

GROUP WITHIN: Schedule a set of exposures within some time interval. The order of the exposures is not important but no two exposures can be more than the allotted time apart.

SAME ORIENTATION: The V3 axis (see Figure 2) of the telescope can be oriented in space by an angle that is offset from a fixed coordinate system. If a proposer requests it, two exposures can be required to be taken at the same orientation.

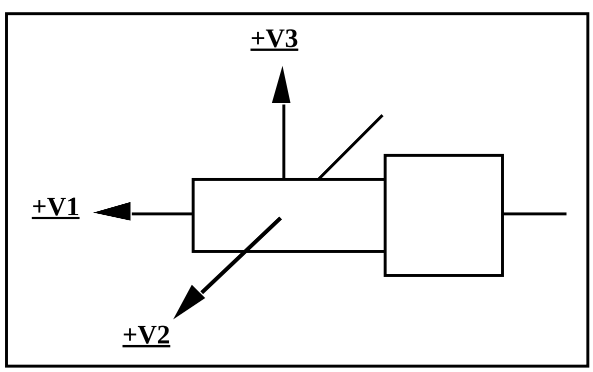

Figure 2. The Space Telescope's three axes are +V1 (points toward the astronomical target), +V2 (solar panels rotate about this axis), and +V3 (orthogonal to V1 and V2).

Resource Constraints

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

The reasons for developing SPIKE include the following:

1. The in-track position of HST is not predictable over long time scales. Thus, the finely detailed schedules (time resolution of spacecraft commands is on the order of seconds) produced by SPSS are not accurate many months into the future. SPIKE does not produce schedules with resolution in seconds but is utilized to schedule SUs to a coarse time segment (a week, conventionally).

2. The load on SPSS with respect to scheduling is reduced if a long term analysis of constraints is done initially. The number of SUs that must be processed for a week long, short term, schedule is reduced from thousands to approximately one hundred. Periodically, the information from one week of a schedule will be communicated to SPSS which will then build a more detailed schedule. The logic of this organization is based on the notion that SPIKE can attempt to optimize a year-long schedule. SPSS will then have far fewer SUs upon which to work and will operate in much less time. Scheduling the SUs in that time range should be successful (based on SPIKE calculations), and a higher quality schedule will result.

Architecture of SPIKE

The SPIKE scheduling environment consists of a core constraint processing system, a user interface, and a general purpose *constraint satisfaction problem* (CSP) system used to search for good solutions to the ST long term scheduling problem. Descriptions of these subsystems follow.

Figure 1 illustrates the data flow model of SPIKE. Proposal data is processed by TRANS producing the Spike Trans Interface File wherein is defined the SU hierarchy and other exposure data. SPIKE processes many proposals at once producing a schedule. A week's worth of schedule is written to the Scheduling Update File which is passed to SPSS. This file contains code that populates a database and specifies the SUs for the week and the SPIKE scheduling windows for each.

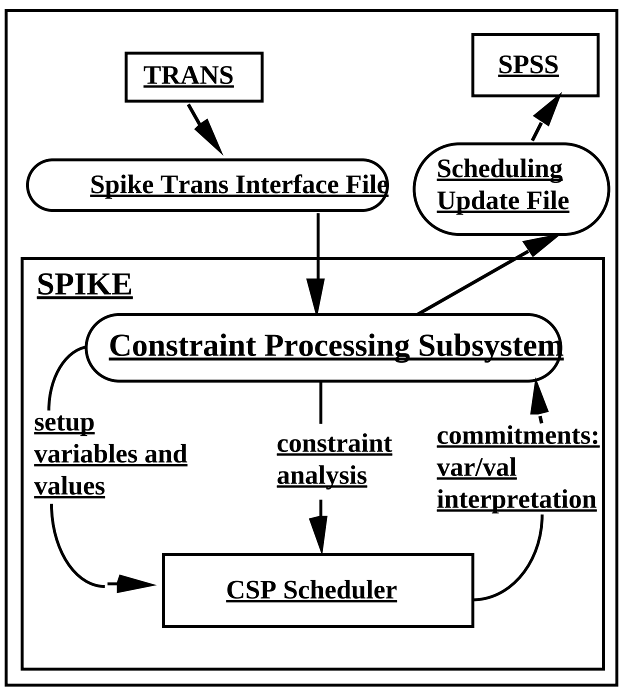

Figure 1. The input to SPIKE is a Spike Trans Interface File which contains the SU hierarchy, targets, and constraints. The Constraint Processing Subsystem is populated with scheduling units and constraint data. The variables (representing SUs) and values (representing weeks) of the CSP subsystem are created from constraint processing subsystem data. As the CSP searches for a solution (good set of variable/value assignments) it will cause the constraint processing subsystem to process these scheduling commitments and analyze the constraints. The results of these analyses are returned to the CSP and violations are logged as conflicts. Week long portions of an acceptable schedule are then passed to SPSS via the Scheduling Update File.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

Expert Assistant[1] was developed. At the Space Telescope Science Institute (STScI) an expert system called TRANSFORMATION[4] is operational (this system is discussed later). Advanced technologies prototyped for analysis include neural networks[2] and genetic algorithms[12].

2.DESCRIPTION OF THE HUBBLE SPACE TELESCOPE

NASA's Hubble Space Telescope is an orbital observatory that was launched by the Space Shuttle *Discovery* in April of 1990. It has six scientific instruments (Wide Field/Planetary Camera, Goddard High Resolution Spectrograph, High Speed Photometer, Faint Object Camera, Faint Object Spectrograph, and Fine Guidance Sensors) and provides improved resolution and sensitivity because it is above the earth's obscuring atmosphere. The Space Telescope Science Institute is responsible for managing the scientific operations of HST.

Proposals for observation of astronomical objects are submitted by astronomers and are processed by a series of software systems including SPIKE. The results of this processing is a series of spacecraft commands which cause data to be obtained from scientific instruments (SIs). This is data is returned to the STScI for analysis and archiving. For more details about HST, see [5].

3.ARCHITECTURE OF THE GROUND SYSTEM

Various systems employed at the STScI to process proposals include the Proposal Entry Processing System (PEP), TRANSFORMATION, SPIKE, and the Science Planning and Scheduling System (SPSS). Data flows through these systems as if through a pipeline.

3.1.PROPOSAL DATA DEFINITIONS

Some definitions will clarify the discussion of the ground system. Each proposal is prepared by an astronomer and contains *target* specifications and *exposure* specifications. An exposure is defined as an observation of a point in space by a scientific instrument. Each exposure specification includes the duration of the exposure and target data (e.g., position of an astronomical object).

Various constraints (called *special requirements*) may be placed upon the exposures by the proposer. The types of constraints include *temporal constraints* (e.g., after), *target constraints* (e.g., the phase of a target with periodic behavior), and *spacecraft constraints* (e.g., HST must be in the earth's shadow during observation).

Subsystems SPIKE and SPSS require that exposures be grouped into *scheduling units* (SUs). The logic of this grouping depends on knowledge obtained from operations astronomers. An example heuristic is: Grouping two exposures must not violate timing constraints.

3.2.PROPOSAL DATA FLOW

PEP performs syntax checking (the specifications conform to a formal syntax) and populates a database with information. A rule-based expert system called TRANSFORMATION processes proposal data from PEP and is responsible for grouping exposures into scheduling units. The output of TRANS is the input to SPIKE and SPSS. The SPIKE system is used to create long term schedules (for periods of one year or more). SPIKE feeds the data from one week of its long term schedule to SPSS. This system is used to create a more finely detailed schedule consisting of specific spacecraft commands.

The complete path of a proposal is discussed in detail in [9].

4.THE SPIKE SCHEDULING SYSTEM

SPIKE is a long-term scheduling system. It provides constraint analysis algorithms and a graphical user interface (GUI) that allows a user to manually or automatically place scheduling units into week long bins.

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.

AN AI SCHEDULING ENVIRONMENT FOR THE HUBBLE SPACE TELESCOPE

Jeffrey Sponsler, Mark Johnston,
Glenn Miller, Anthony Krueger,
Michael Lucks, Mark Giuliano

Space Telescope Science Institute*
3700 San Martin Drive
Baltimore MD 21218
(301) 338-4565, sponsler@stsci.edu

ABSTRACT

This report describes the SPIKE (Science Planning Intelligent Knowledge Environment) system that is currently used to create long term schedules for the Hubble Space Telescope. SPIKE employs the suitability function ("maps goodness over time") as its fundamental knowledge representation for scheduling constraints. To support the search for optimal schedules, a constraint satisfaction system is used. An initial long range schedule is generated using a greedy algorithm and a heuristic algorithm is applied in order to repair the schedule (specifically portions where constraint violations have occurred). Work in progress includes the scheduling of parallel exposures and the design of a machine independent windowing environment.

1.INTRODUCTION

This report describes a system called SPIKE. SPIKE is used by astronomers at the STScI to verify pre-defined long-term schedules and to create new schedules for the Hubble Space Telescope. The system employs techniques of artificial intelligence including heuristic repair of schedules. The input to SPIKE includes data about scientific proposals created by astronomers and the output is a complete long term schedule (where the time scale is on the order of months).

The following issues will be discussed in this report:

1. An introduction to the Space Telescope and a description of its mission.

2. The software systems for science scheduling and the flow of data through the system from proposer to telescope.

3. The architecture of SPIKE; this includes the knowledge representation, the operational scheduling process, the HST constraints that are handled, a constraint satisfaction tool-set, and heuristic techniques for searching.

4. Automated testing tools for the Lisp environment are presented.

5. Subsystems of SPIKE that are currently under development (parallel observation scheduling, portable graphical user interface).

Generating "good" schedules for the Hubble Space Telescope (HST) is an important goal for the following reasons: The spacecraft is an expensive observatory. Therefore, procuring the greatest amount of scientific data during its lifetime is desirable. Algorithms and heuristics developed for scheduling in such a complex domain can be applied to other problems (i.e., technology transfer).

A number of systems pertaining to HST that apply artificial intelligence methodologies have been implemented. At the Space Telescope European Coordinating Facility, a system to aid in the creation of proposal specifications called the HST

* Operated by the Association of Universities for Research in Astronomy for the National Aeronautics and Space Administration.