Planning and Monitoring Solar Array Operations on the ISS

Sudhakar Y. Reddy[1], Mitchell Ai-Chang[5], Michael J. Iatauro[1], Elif Kürklü[1], Matthew E. Boyce[2], Jeremy D. Frank[3], and Ari K. Jónsson[4]

[1]Perot Systems Government Services, [2]MCT, Inc., [3]NASA Ames Research Center, [4]Reykjavik University, [5]Intrinsyx Technologies Corp.

NASA Ames Research Center, Moffet Field, CA 94035

{Mitchell.Ai-Chang,Sudhakar.Y.Reddy, Michael.J.Iatauro, Elif.Kurklu, Matthew.E.Boyce, Jeremy.D.Frank}@nasa.gov, Ari@ru.is

Abstract

Managing the International Space Station (ISS) solar arrays requires flight controllers to constantly balance multiple complex constraints against power needs. The complexity not only impacts planning activities, but has an even more acute effect on real-time operations, in particular when handling unexpected events or changes in operations plans. The Solar Array Constraint Engine (SACE) has been developed to assist the flight controllers with the task of planning and executing solar array operations in a safe and effective manner. SACE is built on top of the EUROPA model-based planning system, using its constraint management and automated planning capabilities to reason about the different constraints, find optimal array modes and orientations subject to these constraints and user-configurable solution preferences, and automatically generate solar array operations plans. In addition to operations planning, SACE provides situational awareness, what-if analysis, and optimization functionality.

Introduction

As the International Space Station (ISS) nears completion, new solar arrays have been added to improve the power availability to meet the demands of the new science and crew modules, which in turn added a great deal of complexity to ISS operations planning and monitoring. The solar arrays are designed to automatically track the sun, as the station revolves around the earth, to maximize the power production. However, normal ISS operations such as water dumps, visiting spacecraft (Space Shuttle, Progress and Soyuz), and extra vehicular activities (EVA) put complex constraints that ultimately define safe array configurations, due to structural load limits, contamination concerns, and thermal impacts, which in turn limits power generation. Additional constraints are imposed on account of contingency planning done to minimize risks involved in human flight operations.

ISS Mission Operations are complex; planning and validating real-time operations require consideration of nominal and off-nominal situations, involve long preparatory lead times, upfront and real-time coordination among multiple groups, and more. Operations are controlled by experts in mission control, organized into teams that manage aspects or subsystems of the station. The Power, Heating, and Lighting Controllers (PHALCON) constantly balance multiple complex constraints against power needs and power production capability. The power system itself consists of the solar arrays, the joints that orient the arrays, the batteries, the charging systems, the power loads, etc. PHALCONs interact closely with other groups, such as the Power Resource Officers, attitude control experts, and, due to the key role that power plays, eventually with all other groups. The interaction of power control and attitude control is a good example of the complexity of these operations. Even during routine operations, the attitude (roll, pitch, and yaw) of the ISS needs to be changed for docking and undocking, debris avoidance, and re-boosting for orbit correction. PHALCONs must closely coordinate their plans with the attitude control team, because the attitude change is accomplished through firing one or more thrusters attached to the ISS or the docked vehicles. These actions exert forces on the solar arrays and subject them to contamination if they are not properly oriented. Attitude and other configuration changes also impact how arrays are best oriented for power generation, which in turn might impact power availability.

Currently, PHALCONs take about four weeks of calendar time to manually produce an ISS solar array operations plan for a typical four week planning horizon, a process requiring manual transfer of information between different teams. Furthermore, certain constraints can only be checked when a plan is fairly complete, requiring multiple revisions to the plan before a valid one that meets all the different constraints can be produced.

AI Planning in Space Operations

The first AI system to manage a spacecraft was the Remote Agent (Muscettola, et.al 1998). Part of the system was the Remote Agent planning system (Jónsson, et.al 2000), which automatically generated plans to achieve operations goals, taking a simple power model into account. The potential of automated on-board power management has been demonstrated for a satellite with signal processing payloads (Shriver, et. al., 2002). The satellite's power system is considerably simpler than that of the ISS, and the spacecraft attitude (and thus power generation) is fixed; however, the onboard system can choose data processing options of differing power consumption for different expected science return. Automated power management was used for NASA Goddard's ST-5 mission (Stanley, et.

al., 2005). In this work, the power management system ran in a fully automated mode in Mission Control Center, but its role was limited to characterizing the performance of the ST-5 power system, and notifying mission planners of future constraint violations in the mission schedule due to changed expected power availability or consumption.

The ASPEN model-based planning system from NASA's Jet Propulsion Laboratory was used in a demonstration of automated scheduling of a lower-fidelity model of the ISS power system (Rabideau, 2008). However, this model lacked the highly detailed, context-dependent models required to manage the ISS, and did not manage live updates from the ISS telemetry stream.

Mixed-initiative planning systems have been used in spacecraft operations before. The first was the MAPGEN system, used to build plans for Mars Exploration Rover mission (Bresina et al, 2005). MAPGEN was based on the EUROPA planning framework (Frank and Jónsson 2003) and assisted users with building efficient and safe rover plans each day. The tool, however, did not have a fully integrated power planning element, but rather relied on a simplified power model to do planning and then checked the plan against a more complex calculation. MSLICE and PSI are follow-on applications for upcoming Mars missions (Aghevli et al., 2006). A more recent AI tool in mission operations is the MEXAR2 mixed-initiative planning tool, which is used to plan downlink operations for the Mars Express spacecraft (Cesta et.al, 2007).

Challenges of AI Planning in ISS Mission Control

Flight planners and controllers use certified processes and procedures in determining the optimal, but safe, orientations of solar arrays. The processes define "imperative algorithms" for safe operations of the solar panels. One of the technical challenges is in mapping the implicit elements of these imperative algorithms into declarative representations used by AI planning systems. Another challenge is in translating the nonlinear continuous constraints into a representation that can be efficiently reasoned by discrete variable and value-based constraint reasoning engines. The very important, but often overlooked, challenge is in supporting a complex requirements analysis and development process for producing flight-certified software, which inevitably involves changes in the problem being solved and subsequent changes to the solution methods.

The objective of this article is to introduce a very real and interesting problem to the planning community, to describe how AI planning and constraint reasoning technology is being used to solve this real problem, and to give insight into what it takes to work with customers to develop complex applications of planning technology. The paper is organized as follows: first, we define the problem; next we present the approach taken in the tool we developed; then we describe the functionality and interface of the tool; and finally we end with some concluding remarks about lessons learned and future work.

The Problem

The ISS will soon have eight solar arrays, each of them mounted on a rotary joint called the Beta Gimbal Assembly (BGA). Each set of 4 BGAs are mounted on a truss attached to a Solar Alpha Rotary Joint (SARJ), one each on the starboard and the port sides of the ISS. Therefore, each solar array has two degrees of rotational freedom, though some degrees of freedom are constrained by the shared SARJs. Further, the rotary joints can be in different modes e.g. auto-tracking the sun, parked in a specific position, latched (for BGA) or locked (for SARJ). The objective of the Solar Array Constraint Engine (SACE) is to determine appropriate modes and orientations for the different joints for safe operation under different ISS configurations and events. This capability is used both for analysis of current or future ISS configurations, and for producing long-term array plans.

Solar Array Constraint Management

The decisions to make are the orientations and modes of the different joints. The configuration constrains legal and optimal decisions; configuration variables include, among other things, the attitude, the combination of thrusters that are firing, the specific event (docking, undocking, attitude hold, water dump, etc.), the specific ports for docking, and the solar beta angle (the elevation of the sun relative to the orbit plane of the ISS).

The power provided by any array will be the maximum if it automatically tracks the sun, but this is not always a safe mode in which to operate. Constraints that limit safe operation fall into the following categories: power, structural loads (on BGA and SARJ), contamination and structural member shadowing. If power availability drops too low, some ISS subsystems can be shut off, but a minimum power is needed for life support and other critical functions. The attitude of the station needs to be changed periodically to account for various events by firing a combination of different thrusters, which in turn imposes structural loads on the array joints. The thruster plumes and water dumps can cause contamination of the arrays, degrading their power generation capacity. Differential shading of longerons (structural elements that keep the array blankets in tension) results in uneven thermal expansion, which in turn can lead to catastrophic failure of the arrays.

The power, load, and longeron shadowing calculations all require detailed numerical analyses that are computationally expensive to run during real-time operations. However, the results of these analyses for pre-determined configurations are available in the form of tabular data. The various constraints further map array configurations into three color-coded zones – red when the constraint is violated and the orientation is considered an infeasible solution or a keep out zone, green when the operating zone is feasible, and yellow an intermediate zone where one can operate if there is no green zone solution available.

Objectives and Preferences

The objective is, for a given configuration of the ISS, to find orientations and modes for the different arrays that maximize the power availability, but at the same time keep them in a feasible space with respect to the various constraints. The ISS certified operations procedures express the objective in terms of solution preferences. For orientation determination, this can be paraphrased as:

In finding a solution, first avoid all orientations that cause red power, then avoid red loads, next avoid red longeron shadowing, then avoid yellow loads, after that avoid red environment, and then find a location that maximizes power.

Under normal operations, a feasible solution is one that produces at least yellow power, but feasibility constraints are less restrictive during contingency operations.

A simplified version of the imperative procedure for determining the mode (autotrack, park, lock a SARJ, or latch a BGA) in a given configuration can be paraphrased as follows:

In determining a mode, prefer autotrack to park, and park to latch or lock. If the array loads are in the red zone, latch the BGA, and if the SARJ loads are in the red zone, lock the SARJ. If the current orientations are safe, but if there is a possibility of the loads on any joint getting into danger zone during autotracking, avoid autotracking that joint. Further, if there is a possibility of the contamination constraints getting into the danger zone during autotracking, avoid autotracking, except during contingency operations.

The modes cannot be independently determined for each joint, because four BGAs are mounted on each SARJ. The actual procedure has about forty lines of if-then-else clauses and involves calculating intermediate values for modes that get overwritten by other values depending on the intermediate values of the modes of other joints. Also, restrictions on modes for certain joints and preferences between modes for other joints could be specified by the user at run-time, based on the health status of the different BGAs and SARJs.

Planning

The planning problem is to build a solar array plan to change array modes and orientations, given the projected evolution of configurations and constraints in time. The evolution of configurations is defined by an *attitude timeline* (ATL). Figure 1 shows a sample ATL with two main events that are broken down into sub-events, each of which is associated with a time, solar beta angle, attitude type, reference frame, and YPR (yaw, pitch and roll). Some events have fixed start and end times, but others can have flexible start and/or end times. In Figure 1, events 5 and 6 have fixed start and end times, but other events have flexible end times. A *thruster timeline* (TRTL) determines thrusters that are allowed to fire during various events, both during normal and contingency operations.

#	Start-Stop GMT	Beta Angle	Attitude Name	Ref Frame	YPR	Event
ATV_Propellant_Line_Purge (M08_170_A_04.UAF)						
1	170/08:30 --	-27	+XVV +ZLV TEA	LVLH	355.0 357.3 358.0	Handover US to RS for Prop Purge (SM)
3	170/10:00 --	-27	+XVV +ZLV TEA	LVLH	355.0 357.3 358.0	Handover RS to US Momentum Management
ATV_Reboost (M08_171_A_03.UAF)						
4	171/05:00 --	-28	+XVV +ZLV TEA	LVLH	355.0 357.3 358.0	Handover USS to RUS
5	171/05:05 171/05:10	-28	+XVV +ZLV	LVLH	5.1 357.2 0.0	Mnvr to Reboost Attitude (ATV on SM Aft)
6	171/06:56 171/07:01	-28	+XVV +ZLV TEA	LVLH	355.0 357.3 358.0	Mnvr to LVLH TEA
7	171/07:30 --	-28	+XVV +ZLV TEA	LVLH	355.0 357.3 358.0	Handover RUS to USS Momentum Management

Figure 1. Part of a sample attitude timeline (ATL) for ISS operations, showing propellant purge and reboost events.

Additional constraints that govern planning include the maximum rates at which the different joints can be slewed or turned, and the minimum durations on time intervals between switching modes or orientations to account for minimum time required for authorizing and issuing commands and monitoring their completion. Further objectives during planning include the minimization of the turns of the rotary joints, which is preferred to maximizing power availability once sufficient power is available to meet critical needs (power is in the green zone). Another consideration for slewing actions is to minimize the change in direction of rotation of the joints.

The Approach

Several guiding principles have driven the choice of the approach to address the problem. First, the PHALCONs wanted to gain confidence in the approach, so the application had to be developed in stages – first to address monitoring and optimization for a single configuration, and in the next phase planning over a time horizon. Second, we needed to follow already certified procedures as much as possible. We chose a constraint-management based approach for monitoring the arrays, during real-time execution, to ensure that they are operating in a safe manner with respect to the different context-dependent constraints. For optimization of an individual configuration, we used an exhaustive search of a discretized space along with a specially designed cost function that faithfully encodes the certified solution preferences. Finally, our planning approach uses a model-based planning system to model the domain states and actions. We chose a greedy optimizer to find a locally optimal array position at each stage of the time evolution of the plan. The sequence of locally

optimal array solutions is checked to ensure there is sufficient time to change modes and array positions; a simple "configuration merging" repair strategy is used to fix violated constraints. Even though the solution is not globally-optimal, it follows the approach currently used by the PHALCONs, and thus makes it easier to gain their confidence in the solutions produced by the tool. The overall approach is described in Figure 2.

SACE uses the Extensible Universal Remote Operations Planning Architecture (EUROPA) framework for optimization and automated planning. This model-based planning system accepts a declarative description of a class of planning problems consisting of a list of *timelines* (concurrent threads of a plan), a list of *states* that may hold on each timeline over an interval, and *compatibilities* describing the relationships that must hold between timelines in order for a plan to be valid. The EUROPA framework provides an interface that allows programmers to build customized planners that meet the needs of their applications. EUROPA incorporates special purpose modules for reasoning about time, general constraints, managing timelines, managing applicability of compatibilities, and managing search control heuristics. For a detailed description of the EUROPA framework and the underlying concepts, please refer to (Frank and Jónsson 2003).

Modeling Solar Array Constraints

Each solar array has two degrees of freedom, defined by the SARJ angle (α) and BGA angle (β). The power, load, environment and longeron constraints are represented as 2-dimensional tables, with rows representing different values of α and columns representing different values of β. The domains for α and β angles are discretized into single degree increments. Each cell in a table maps an (α, β) angle pair into a user-configurable red, yellow or green zones, as discussed earlier. As some of the constraint data is generated at a coarser granularity, SACE uses numerical interpolation to find the constraint values for the intermediate values of α and β. EUROPA's constraint representation makes it convenient to represent and reason about these tabular constraints and color mappings. As mentioned previously, the set of active constraints depends on the configuration; only one table for each constraint class is active in a configuration. These constraints are essentially logical implications, and are easily described in EUROPA. Finally, the aggregation constraints amount to taking the worst of all applicable tables to evaluate an array position (α,β) and configuration: if any table is red, the array position aggregates to red, if no table is red but any table is yellow the array position aggregates to yellow, otherwise the array position aggregates to green. Again, these constraints are easily modeled in EUROPA. The PHALCONs frequently want to restrict the domains for the various angles; for example, a SARJ malfunction forced the PHALCONs to reason about power availability achieved without moving the SARJ. While this required special care when the domain extends over the 360° to 0° boundary, again, EUROPA's representation makes it easy to manage such constraints.

One of the tougher challenges was translating the imperative representation of the mode determination procedure into a declarative representation. Especially because of the interdependence of the BGA and SARJ modes, this required splitting the imperative procedures described previously into several declarative constraints, together called the *lock-latch constraint set*. These constraints are also represented in EUROPA.

As discussed earlier, differential shadowing of the longerons causes structural stresses that can be catastrophic to the solar array. At any given point in time, one can assess the time to criticality by a complex function of the time in shadow for the four different longerons that are part of a given solar array. During real-time monitoring, these are easy to compute based on the telemetry feeds. However, during planning or optimization, the calculation of a longeron constraint requires the precise information about the starting state of the shadowing timers, and in addition relies on a detailed simulation of the shadowing for a specific configuration and specific modes and orientations of the arrays, as the station revolves around the earth. Therefore, unlike the load and contamination constraints, the longeron shadowing constraints cannot be evaluated ahead of time for use during planning.

Orientation and Mode Optimization

The finding of optimal array assignments for a single configuration is posed as an unconstrained optimization problem. Both the constraints and the solution preferences are encoded into a cost function (L) that is minimized. The optimization problems for the starboard and port sides of the station are independent. On each side, however, the shared SARJ means that the individual arrays cannot be independently optimized. In this formulation, adjacent array shadowing is not considered, so the overall cost function is split into n independent functions, where n is the number of BGAs sharing a SARJ, and is either 2 or 4 depending on the stage of assembly.

We use a two-stage approach. First, we calculate the cost of each feasible orientation, independently for each solar array. An orientation is feasible if its power table entry is not red, its load entries (SARJ and BGA) are not red, and its longeron entry is not red. (The environment table entry is allowed to be red in feasible orientations.) The cost function for each array is a function of the SARJ and BGA angles, and can be represented as a weighted sum of other cost functions:

$$L_i(\alpha,\beta_i) = w_c L_i^c(\alpha,\beta_i) + w_m L_i^m(\alpha,\beta_i) + w_d L_i^d(\alpha,\beta_i) + w_\theta L_i^\theta(\alpha,\beta_i) + w_p L_i^p(\alpha,\beta_i)$$

The component cost functions refer to the color cost, L^c (due to different constraints being in the red, yellow, or green zones), the mode cost, L^m (due to modes of the BGA and the SARJ), the distance cost, L^d (due to distance between current orientation of arrays and the solution), the direction change cost, L^θ (due to change in direction of

trajectory of arrays), and power cost, L^p (due to incremental differences in power within a constrained power zone).

The distance, direction change and power cost functions are simple algebraic expressions. The mode cost uses a simplified version of the imperative procedure described earlier. In order to find the color costs, we used a linear programming formulation to solve for these costs off-line. The linear constraints encode the solution preferences discussed in the problem formulation section. We again used an off-line linear programming approach to calculate the weights in the cost summation above, such that the user-desired preference order of first optimizing using the constraint color zones, next optimizing the modes, then minimizing the distance, etc. is maintained.

During the second stage, the overall cost for each orientation (α, β_1, β_2, β_3, β_4) is calculated as the sum of the costs for the best β_i for each α with respect to the corresponding L_i. The optimal orientation is then the orientation that minimizes the overall cost, L.

During periods when solar beta is high, exterior arrays cast shadows on their adjacent interior arrays. Therefore, the power produced depends on the orientations of the pair of arrays, and the power cost, L^p , cannot be independently estimated for each array. Therefore, the power cost computation need to be moved to this second-stage of the algorithm, and an exhaustive search over (α, β_1, β_2) and (α, β_3, β_4) is needed for finding an optimal solution.

Once an optimal orientation is determined, we then determine the modes for the different joints. The mode costs used in the cost function of the first stage are optimistic lower-bound estimates. Therefore, the full mode determination preference described in the problem formulation section is used to determine the modes corresponding to the optimal orientations.

Automated Solar Array Plan Generation

The ATL, shown in Figure 1, along with the TRTL, maps the events on the timeline into one or more nominal configurations (C). Further, each nominal configuration could be associated with a thruster contingency configuration (C_A). The main and contingency ISS configurations together define the context when the different sets of load, contamination, etc. constraints (X) are active. The ATL also defines the flexible time interval over which a configuration is active.

The problem state (S) is composed of the array orientations (α, β), joint modes (m), and the (sequence of) station configurations (C, C_A). The possible set of actions (A) include the actions to change the mode (park, lock, latch, autotrack) of the different BGAs and SARJs, and the turn or slew actions to change the orientation of the arrays. Both the state and the action have an extent in time, defined by the start and end times (t_s, t_e); one or both these times could be flexible. The goal of the planning problem is to find a set of actions, states, and their extents that are consistent, do not violate the constraints (X), and optimal with respect to the solution preferences.

SACE represents the states and actions on timelines in EUROPA. In EUROPA, there is no representative difference between states and actions, but for this discussion, the configurations, array orientations, and joint modes are treated as states. The actions either change the mode (Park, Latch, Lock, Autotrack actions) or change the orientation (Turn/Slew action). The basic planning constraints are: (a) array orientation does not violate feasibility constraints on load, contamination, etc.; (b) array modes are feasible for chosen orientations; (c) joints are in position before parking, latching, or locking; and (d) actions meet minimum duration requirements for commanding and execution. SACE uses the optimizer discussed in the previous section to address constraints (a) and (b), and uses the EUROPA planner to address constraints (c) and (d).

The planning problem is NP complete, and given the size of the problem at hand, it is impractical to obtain a globally optimal solution through exhaustive search. On each of the starboard and port sides, the search space for finding a solution for each configuration is governed the variable set, (α, β_1, β_2, β_3, β_4), each of whose domain is integers 0-359. Even with the independence assumptions, discussed in the previous section, and the use of local propagation of feasibility and maximum turn rate constraints to restrict search space to 1% per configuration, the overall search space when planning over 40 configurations on a timeline is of the order of 10^{120}. More importantly, the memory required to hold the evolving feasible search space will be prohibitive.

The solar array planning algorithm works as follows:

For each configuration C_i in increasing start time order
  Optimize the array orientation
For each configuration C_i in increasing start time order
  While C_i conflicts with the previous configuration C_{i-1}
    Merge C_i and C_{i-1}
    Optimize the new configuration
For each configuration C_i in order of start times
  Insert turn and mode change actions

In the initial phase of the three-phase planning process, information from the ATL and the TRTL are represented as *tokens* on respective *timelines*. The tokens on these timelines are then translated into one or more configuration tokens C on a different timeline, based on rules in the planning domain model. The timeline is completely filled by these tokens, i.e. the end of configuration C_{i-1} is the start of configuration C_i. The SACE planner then iterates over the configuration tokens in order of their start times, finding the optimal orientation and mode using the local optimizer. Each BGA and SARJ is represented by a timeline as well, and the optimal solution is translated into state tokens (to represent orientation and mode) and action tokens (to change mode or orientation) on these timelines. The constraint management system in EUROPA restricts the domain of orientations for the down-stream configurations (based on turn rate constraints, for example)

as the solution for a given configuration is specified on the timeline.

After each of the configurations are optimized, the planner validates the resulting orientations and modes generated thus far to detect any temporal constraint violations (constraint classes (c) and (d) above). Problems may arise due to insufficient duration to turn an array from one orientation to the next in between two configurations, insufficient duration for lock/unlock a SARJ array and/or latch/unlatch a BGA, insufficient duration for commanding a turn action, etc. In case an inconsistency is detected, the planner retracts the solution for the previous configuration, merges the optimization problem for these two configurations, and repeats the process.

When two configurations are merged, the union of the constraints that are active in each of these configurations are activated in the merged context. A single solution for the orientations and modes is obtained to cover both configurations. For example, suppose that a configuration C_i defined by a thruster firing requires a BGA position optimized for loads and contamination. The optimal array positions in this configuration may require slewing the arrays given configuration C_{i-1}. If there is insufficient time during C_{i-1} to slew the arrays, the configurations C_i and C_{i-1} are merged, and a new array configuration is found. Now the constraints are checked assuming the slew starts at the end of C_{i-2}. (Note this does not mean the thrust actually starts earlier, SACE only plans the array configurations.) Though this process is not guaranteed to always find a feasible solution, in our experience, most conflicts are due to insufficient duration for actions and can be resolved in this manner.

Once the entire configuration timeline is processed and all temporal and precondition constraints are provably satisfied, the resulting BGA and SARJ timelines are processed by the EUROPA planner to add the necessary actions (turns and mode changes). The final plan, represented by the BGA and SARJ timelines, is post-processed first to determine feasibility with respect to the longeron shadowing constraint (because calculating the timers requires knowing the complete event sequence), and next to calculate the detailed power availability along the timeline.

The Tool

The SACE tool addresses the three major functions of the PHALCONs: understanding the current situation, looking at the implications of a fixed future configuration, and planning. SACE provides a "telemetry view" for situational awareness, a "sandbox view" for what-if analysis and optimization, and a "plan view" for automated planning.

Both the telemetry and the sandbox use the same interface components with a few operational differences. In the telemetry view, SACE receives the state information (configuration, orientation, and mode) from the ISS telemetry stream, whereas in the sandbox mode, user can input the configuration and other state information. The main window, which has a common form for the telemetry and sandbox, is shown in Figure 3, and has three different areas. The top left area shows the orientations, modes, and over-all summary of the constraint status for the BGAs and SARJs. The top right area shows the configuration variables, which determine the context for the different constraints. SACE reads the constraint information given in the constraint definition files, uses the configuration information in this region to determine the set of applicable constraints, and then evaluates each constraint, providing an indication of whether the state of that constraint is unknown, safe, caution or danger. EUROPA represents conditional constraints naturally, whether the configuration is determined from telemetry or specified by the operator.

Per-constraint category state information is displayed at the bottom of the main window. SACE lets the flight controller to activate or deactivate individual constraints or groups of constraints using the check boxes. The aggregate safety status of array orientations and modes is displayed on the top left hand side of the main window. The aggregation is based on a "least-safe" rule over all applicable constraints. These rules are naturally expressed

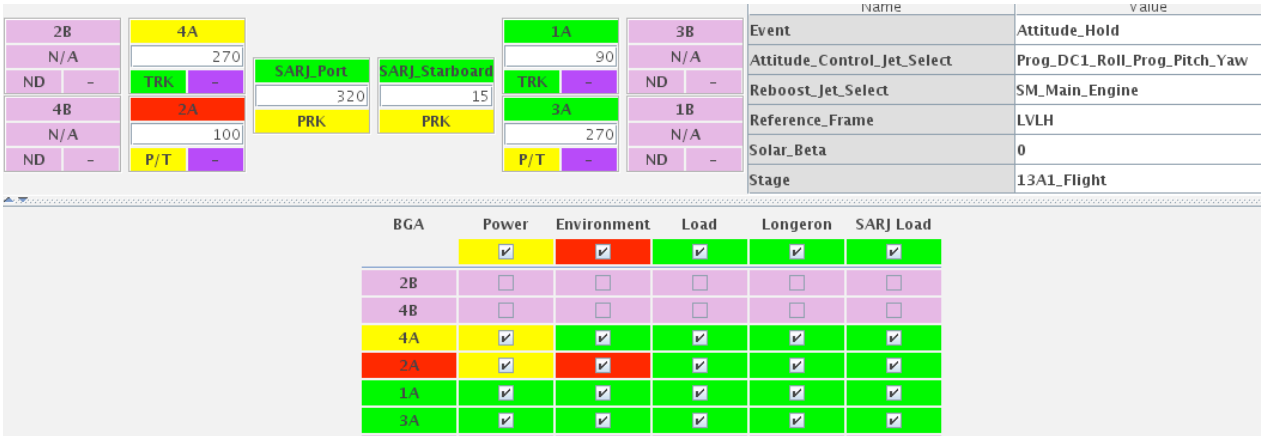

Figure 3. The main window of SACE shows the configuration variables on the top right, the orientations and modes on the top left and a summary view of the constraint status at the bottom.

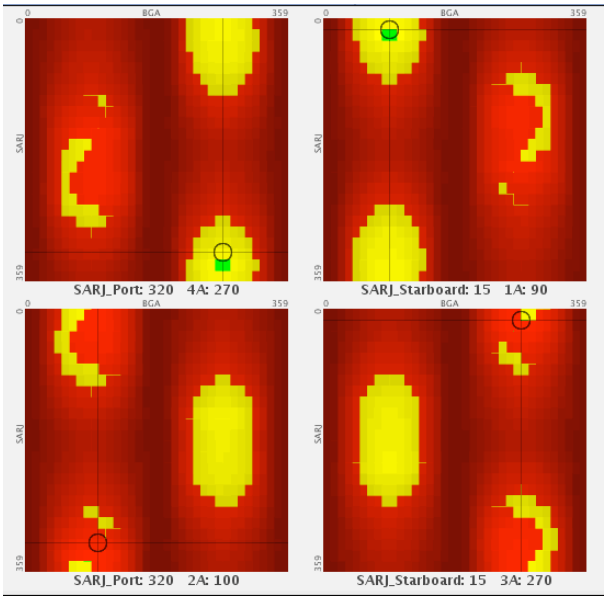

Figure 4. The map view in SACE shows the status of the constraints as a color-coded map, where the Y-axis is α and X-axis is β. The four maps correspond to the four active solar arrays in this configuration.

using constraints to capture the implications.

Contextual information, providing an indication of constraint states for ranges of angles other than the currently chosen ones, can be displayed in two primary forms in both the telemetry and sandbox: a map view displays a two dimensional map, showing the aggregate constraint state for any combination of α and β; a ring view displays a set of concentric rings, showing the constraint state of each individual constraint as well as the aggregate state for all β values given an α and all α values given a β value for each BGA. The map view is shown in Figure 4, and uses color-coding to display the status of the constraints and uses intensity to show power availability.

The sandbox allows the user to evaluate arbitrary orientations in any given configuration in terms of the various constraints and power availability. The user can also invoke the optimizer from the sandbox to automatically find orientations and associated modes that maximize power availability, subject to feasibility constraints with respect to the loads, power, contamination and longeron shadowing. The user can arbitrarily restrict the search space for the array orientations and modes, as well as change the preferences on whether include, during optimization, a specific cost metric such as the distance cost.

The planning component of SACE allows the user to load an ATL and a TRTL from files and automatically determines a solar array plan. The resulting plan is displayed as interactive timelines, as shown in Figure 5. The user can review the plan in this window, or dig deeper by loading each configuration into the sandbox by clicking on the desired configuration. The user can also edit the plan. Possible editing options include moving ATL elements along the timeline or adjusting their duration and restricting the orientations and modes for the different joints.

Among the elements in the planning window are the SARJ and BGA timelines, showing the modes (including orientations) and turns. Other timelines show the status of the longeron shadowing constraint and the power availability. SACE interacts with a trajectory and power modeling tool called SOLAR for calculating the power availability, the direction of the sun, and the paths taken by the arrays while tracking the sun.

Concluding Remarks

We have described the challenging task of automated planning and optimization of ISS solar array operations. SACE will reduce the time it takes to produce solar array plans from weeks to hours by providing an end-to-end solution that starts with a timeline of ISS attitudes and events and automatically produces a solar array plan and a timeline of power availability. The tool also supports real-time monitoring and responses to unexpected events or changes. SACE is currently undergoing testing for flight certification by the Mission Operations Directorate. Once deployed, SACE will be the first optimization and automated planning tool to be used for human spaceflight.

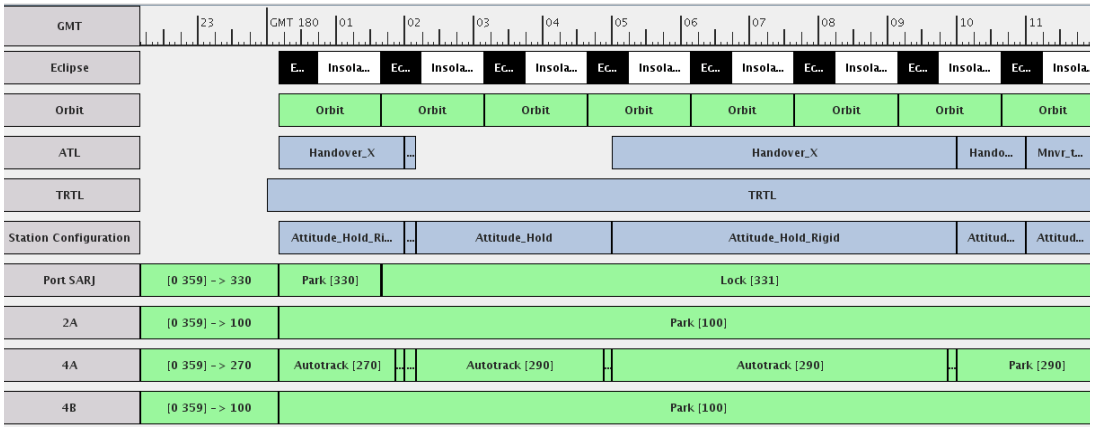

Figure 5. The SACE planning window shows the timelines for the ATL, TRTL, the configurations, and the different SARJs and BGAs, among other things.

Lessons Learned

The challenge in getting automated optimization and planning technology into human spaceflight mission operations is to provide significant value that makes up for the technical risk of bringing in new software, especially one based on AI. We first built a prototype to quickly demonstrate the capabilities offered by the technology. This led to a commitment to develop the tool, followed by an initial phase of requirements specification and refinements. To rapidly demonstrate value, we used a spiral development process, to ensure that the SACE tool was usable by customers at all stages for evaluation and feedback. The modularity of EUROPA's code base and API allowed incremental expansion of the application.

Throughout this process, as is common in software development, requirements were continually refined, and new requirements were added. For example, constraints that initially were assumed to be simple table constraints turned into complex calculations. The major reason for the success of this effort is the close coordination between the PHALCONs and the development team through twice-weekly teleconferences in addition to periodic visits to Mission Control to observe the PHALCONs in action.

Over the two years the application has been developed, the set of states and rules has changed considerably. The model-based nature of EUROPA has led to less development of new code compared to application-specific methodologies, since EUROPA's rules language is flexible enough to incorporate many of the desired changes. For example, metal shavings were discovered in one of the SARJs, requiring that the SARJ be turned as little as possible and be parked or locked otherwise. Due to the constraint- and model-based framework used in SACE, this was fairly straight forward to model, and required minor changes to the user interface.

The strictest challenge in getting software into mission control is the certification process. For all mission control applications, this boils down to development, testing documentation, and a formal approval by responsible parties. As exhaustive testing is impossible for automated planning applications, careful design of test cases is important to assure correctness. The formal approval process is based on the customer team evaluating the tool on these test cases, as well as being part of development process to ensure adherence to rules about coding.

Future Work

The optimizer in SACE was carefully crafted to take advantage of independence of parts of the problem. During the course of the development of SACE and station construction, some of these independence assumptions were invalidated, for example due to the impact of adjacent array shadowing on longeron constraint and power output. This means that the BGA angles of pairs of adjacent arrays cannot be independently optimized. This has dramatically increased the time for optimization of a single configuration, which in turn has increased the planning time to a several hours from about half-an-hour. We plan to explore alternative approaches for improving the optimization speed.

During planning, SACE solves for a locally optimal solution in each configuration along the timeline. It is possible that a locally optimal solution for a configuration can restrict the space for future configurations so much that a feasible solution cannot be found. Currently, SACE merges the optimization problem for neighboring configurations to expand the feasible space, and find a locally-optimal solution in this merged space. However, such a solution will likely be sub-optimal. Standard backtracking approaches are not very effective, because solutions in the neighborhood of an infeasible solution tend to be infeasible in this application. We plan on exploring heuristic-guided variable and value ordering approaches to address this potential problem.

References

Rabideau, G. 2008. Space Habitat Power Management with an Autonomous Scheduling System, In *Proc. 9th International Symposium on Artificial Intelligence, Robotics and Automation for Space (i-SAIRAS 2008)*.

Shriver, P., Gokhale, M., Briles, S., Kang, D., Cai, M., McCabe, K., Crago, S., and Suh, J. 2002. A Power Aware, Satellite-Based Parallel Signal Processing Scheme. *Power Aware Computing*, Kluwer Academic Publishers, Norwell, MA, 243-259.

Stanley, J., Shendock, R., Witt, K., and Mandl, D. 2005. A Model-Based Approach to Controlling the ST-5 Constellation Lights Out Using the GMSEC Message Bus and Simulink. In *Proc. Intl. Conf. Software Engineering Research and Practice*.

Frank, J., and Jónsson, A. 2003. Constraint-based Attribute and Interval Planning. *Constraints* 8(4): 339-364.

Muscettola, N., Nayak, P., Pell, B., and Williams, B. Remote agent: To boldly go where no AI system has gone before. Artificial Intelligence, 103(1/2), August 1998

Jónsson, A., Morris, P., Muscettola, N., Rajan, K., and Smith, B. Planning in interplanetary space: Theory and practice. In Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling, 2000.

Cesta, A., Cortellessa, G., Denis, M., Donati, A., Fratini, S., Oddi, A., Policella, N., Rabenau, E. and Schulster, J. MEXAR2: AI Solves Mission Planner Problems. In IEEE Intelligent Systems, 22(4):12-19, 2007

Bresina, J., Jónsson, A., Morris, P., and Rajan, K. “Activity Planning for Mars Exploration Rovers", in Proceedings of 15th International Conference on Automated Planning and Scheduling (ICAPS), 2005.

A. Aghevli, A. Bachmann, J. Bresina, K. Greene, B. Kanefsky, J. Kurien, M. McCurdy, P. Morris, G. Pyrzak, C. Ratterman, A. Vera, S. Wragg Planning Applications for Three Mars Missions with Ensemble. Proceedings of the 5th International Workshop on Planning and Scheduling for Space, 2006.