Evaluating Multi-Objective Evolutionary Scheduling Strategies for the James Webb Space Telescope

Mark E. Giuliano
Space Telescope Science Institute
3700 San Martin Drive, Baltimore MD 21218
giuliano @stsci.edu

Mark D. Johnston
Jet Propulsion Laboratory/California Inst. Of Technology
4800 Oak Grove Drive, Pasadena CA 91109
mark.d.johnston @jpl.nasa.gov

Abstract

Effective scheduling of the James Webb Space Telescope (JWST) requires careful management of the trade-offs among multiple scheduling criteria. Minimizing unscheduled time, angular momentum build-up, and the number of observations that miss their last opportunity to schedule are important objectives that are in competition with each other. Previous studies evaluated alternative multi-objective algorithms for scheduling JWST, including algorithms for breaking down the search into separately assigning and optimizing schedule times and spacecraft roll. This study builds on these results by enriching the JWST scheduling model to include a high fidelity maneuver model between observations, along with realistic science restrictions on how spacecraft roll can be assigned to observations. Additional multi-objective search strategies are evaluated using a parallelized multi-objective algorithm that can exploit available computing resources to reduce overall runtime. We have also incorporated constraints into the problem formulation, to drive the candidate solution set to better sample the Pareto frontier in regions of greater interest. The results not only demonstrate techniques useful for scheduling JWST but also show how multi-objective algorithm features such as parallelism, constraints, and the choice of population size impacts the quality of the schedules generated. Each of the algorithms explored produces a Pareto frontier as output. As a result there is no canonical way of comparing which algorithm performs better. This paper presents tools and techniques for comparing multi-objective algorithms and presents the analogy that this evaluation is essentially a recursive application of multi-objective optimization.

Introduction

The James Webb Space Telescope (JWST) will be the premier astronomical facility of the next decade, replacing two of the current Great Observatories, Hubble Space Telescope (HST) and Spitzer Space Telescope (SST) as a uniquely capable space-based observatory with highly ambitious scientific objectives. Scheduled for launch in 2013, JWST will have a 6.5m primary mirror diameter (compared to 0.85m for SST, and 2.4m for HST), and will primarily observe in the infrared (like SST, and in contrast to HST's primarily optical and UV sensitivity). JWST will study an enormous range of scientific questions, ranging from the early universe to planets within and outside the solar system.

Scheduling a mission such as JWST requires the balancing of many factors. Clearly, such an expensive and unique facility must be utilized as efficiently as possible, and minimizing any wasted time is a primary objective. At the same time, the lifetime of the observatory is limited by consumables such as propellant for reducing momentum buildup in the spacecraft's reaction wheels. Thus, optimization of the JWST schedule is determined by multiple simultaneous objectives, for which there is no well-defined trade-off mechanism that would permit definition of a single combined objective. Multi-objective techniques that keep the objectives separate permit explicit visibility and management of the multiple tradeoffs that are necessary to generate a balanced overall schedule for JWST.

For JWST, two of the primary objectives are minimizing schedule gaps, and minimizing the number of late observations, i.e. that miss their last scheduling opportunity. The more unusual objective is that of reducing angular momentum buildup in the spacecraft reaction wheels, caused by a complex interaction of pointing direction, roll angle, and solar radiation pressure on the tennis court-sized sunshade. Angular momentum buildup must be compensated by firing the spacecraft thrusters, which consumes propellant and thus is potentially a limiting factor on mission lifetime. The angular momentum resource constraint has several important features:

- it is intrinsically 3-dimensional
- resource consumption is vector additive, so that scheduling an observation at a particular time can either increase or decrease the overall accumulation in a nonlinear manner
- angular momentum represents both a hard constraint as an overall limit, as well as a preference to minimize buildup to maximally conserve propellant

These features are different from the types of resources covered in the planning and scheduling literature (Laborie 2003; Policella *et al*. 2004) where activities consume and release a constant capacity. In particular, the non-linearity

of the domain prevents us from employing techniques commonly used to handle resource constraints.

Previous studies on JWST scheduling (Rager and Giuliano 2006, Giuliano *et al*. 2007) demonstrated effective heuristics for scheduling JWST within a simplified spacecraft model. (Giuliano and Johnston, 2008) enriched the scheduling problem by adding the ability to control momentum usage through assigning a spacecraft roll for observations as well as assigning schedule times. Alternative search algorithms were empirically evaluated exploring how the search for schedule times and spacecraft roll could best be performed. The best algorithm decomposed the search into a two step processes. In the first step, observations are assigned times that minimize gaps and missed observations. In the second step, observations are assigned spacecraft rolls that minimize momentum build up.

In this paper we further enrich the scheduling model for JWST by adding:

- A slew model that calculates the overhead to move between observations based on the targets being observed and the selected roll of observations;
- A model of observations that need to be executed at the same telescope roll due to astronomical science considerations.

These new domain features impact the best scheduling decomposition in (Giuliano and Johnston, 2008). The decomposition first determines a schedule and then independently adjusts the roll of observations to minimize momentum buildup. However, the new slew model considers roll as a factor in determining the required separation between pairs of observations. This paper describes modifications to the previous algorithms to account for slews and shows that the same decomposition still results in the best schedules.

In addition to enriching the JWST problem domain this paper explores factors that influence the behavior of multi-objective algorithms. We examine the use of constraints in multi-objective algorithms to guide the search towards required solutions. Also, the paper explores how the shape of the multi-objective search impacts the results. Multi-Objective algorithms have parameters for the population size of each generation and the number of generations evaluated. The paper examines whether it is better to have a deep but narrow search (i.e. high number of generations but a small population size) or a shallow but wide search (i.e. a large population size but a small number of generations). Finally, the multi-objective problem solver was augmented to support parallel execution. This approach allows almost perfect speed-up up to the population size. We examine how this speed-up can be used if the primary execution concern is the algorithm runtime and not the combined processing time of all execution threads.

Each of the algorithms explored in this paper produce a set of optimal solutions, called the Pareto frontier, where no solution in the set is strictly dominated by another solution in the set. Although there has been much research in this area (See Zitzler 2003 for a summary) there is no canonical way of comparing Pareto frontiers to determine if one algorithm works better than another. We utilize multiple techniques to evaluate our algorithms and argue that selecting a multi-objective algorithm out of multiple possible algorithms is analogous to the selection of a single schedule out of multiple schedules on a Pareto frontier. In both cases if there is a dominating solution then the choice is clear. If there is no dominating solution then we can at best provide numeric metrics, and visualization tools.

JWST Mission Operations

JWST will provide time to general observers through a time allocation board. Approved observers will prepare their programs using an automated tool. Programs will be submitted to the JWST Science Operations Center (SOC) and will be scheduled by SOC staff using a two phase scheduling process similar to the process used for the Hubble Space Telescope (Giuliano 1998). In the first phase, a long range plan assigns observations to overlapping least commitment plan windows that are nominally 60 days long. Plan windows are a subset of an observation's schedulable windows and represent a best effort commitment to schedule within the window. In the second phase, successive short-term schedules are created for 22 day upload periods. The short-term scheduler uses plan windows to drive the creation of efficient telescope schedules. This two phase process allows a separation of concerns in the scheduling process: Plan windows globally balance resources, are stable with respect to schedule changes, and provide observers with a time window so they can plan data reduction activities. Short-term schedules provide efficient fine grained schedules to the telescope, handle slews between observations, and provide schedules robust to execution failure.

JWST Scheduling Constraints

A scheduling system for JWST has to satisfy several types of constraints on observations. First, an observation has to satisfy all requirements defined by the user. These include the ability to specify time windows for observations, to link observations via precedence or grouping relationships with offsets, and to link observations via roll constraints.

Secondly, an astronomical target can be observed by JWST only at certain times of the year determined by the location of JWST relative to the sun and the target. We call such time intervals *visibility windows*. The celestial position of the target being observed defines the visibility windows. Ecliptic poles are visible throughout the year, while a target on the ecliptic equator (i.e. on the same

plane as the Earth's orbit) has two visibility windows of about 49 days each.

Thirdly, schedules must satisfy limits on momentum accumulation. The current assumption is that stored angular momentum will be dumped every 22 days during regularly scheduled station-keeping activities. Momentum buildup during a 22-day period over a 24 Newton Meter Squared (NMS) limit will require an extra momentum dump. As momentum dumps require burning scarce fuel, extra dumps will shorten the lifespan of the telescope. It is also preferable to minimize the amount of momentum dumped during the regularly scheduled dumps. Momentum accumulation can be controlled by adjusting the spacecraft roll angle. At any pointing within the field of regard, JWST can roll ±5° from the normal angle without violating spacecraft constraints. Rolling the telescope for an observation impacts the angle that the solar pressure asserts on the sunshield, thus affecting the momentum buildup for the observation.

Schedule qualities

The JWST schedule qualities we desire are the following:

1. Minimize schedule gaps. The JWST contract mandates 97.5% scheduling efficiency. The input set (described below) of 1.2 years worth of observations provides 20% oversubscription to fill gaps in a one year schedule. We expect this level of oversubscription in operations and expect that operations will be able to utilize special gap filling observations.

2. Minimize momentum accumulation. The current operational plan is to dump momentum every 22 days during station keeping maintenance. The goal for the scheduler is to have no or very few 22-day periods that require additional momentum dumps. In addition to the 24 NMS momentum limit, it is preferable to lower the amount of momentum dumped during scheduled station keeping maintenance as that reduces the amount of non-renewable fuel to be used.

3. Minimize dropped observations. The JWST scheduling process first assigns *plan windows* to observations during long range planning. Plan windows are a subset of an observation's visibility window and are created to balance global resources while informing the astronomer when to plan data reduction activities. Missing a plan window can disrupt resource balancing, break the handling of linked observations and disrupt the plans of the astronomer end user.

Input Observation Set

The JWST project has created a Science Operations Design Reference Mission (SODRM), which is a set of observations that closely match the expected mission duration, target distribution, instrument configuration, and constraint selection. It contains the specifications for both astronomical observations as well as calibration observations. The entire SODRM amounts to approximately 1.64 years of observations, including time for slews and other support activities. To allow us to compare with the previous studies we use the same subset of the SODRM, totaling 1.2 years worth of observations, as input to this study. It consists of 2907 observations, including 1822 observations that are linked to at least one other observation. Observation duration varies from 70 minutes to 12 days with a median of 2.08 hours.

The SODRM was updated to include sets of observations that must be executed at the same spacecraft roll in order to achieve the desired science outcomes. These are sets of observations from a single program, called *same-orient sets*, that observe the same target using different wavelengths. Previous versions of the SODRM included these observations but did not link them into same orient sets. The new SODRM adds the scheduling restriction that the observations must be assigned the same roll.

Evolutionary Algorithms

A multi-objective optimization problem to minimize M objectives subject to K constraints can be stated as follows:

$$\text{minimize: } \left\{ f_i(\vec{x}) \right\},\ i = 1 \ldots M$$

$$\text{subject to: } \left\{ g_j(\vec{x}) \right\}^T \leq 0,\quad j = 1 \ldots K$$

Here $\vec{x}$ represents a vector in decision space of dimension D. A solution is called *Pareto optimal* when no improvement can be made to one objective that does not make worse at least one other objective. The set of Pareto optimal solutions is called the *Pareto frontier*. What we seek as a solution to the multi-objective optimization problem is a good approximation to the Pareto frontier. Two important characteristics of a good solution technique are convergence to the Pareto frontier, and diversity so as to sample the frontier as fully as possible.

We have adopted an evolutionary algorithm approach (Deb 2001, Abraham, Jain, and Goldberg 2005) to JWST scheduling. Among techniques developed to solve multi-objective optimization problems, evolutionary algorithms have become popular for a variety of reasons. They have been shown effective on a wide range of problems and are capable of dealing with objectives that are not mathematically well behaved (e.g. discontinuous, non-differentiable). By maintaining a population of solutions they are capable of representing the entire Pareto frontier at any stage.

For this study we have utilized one particular variant called *Generalized Differential Evolution 3*, or GDE3 (Kukkonen and Lampinen 2005) which has been previously used in multi-objective scheduling in a space network application (Johnston 2006, 2008). This technique is based on *Differential Evolution*, a single objective evolutionary algorithm for real-valued decision spaces (Price, Storn, and Lampinen 2005). GDE3 makes use of concepts pioneered in the algorithm NSGA II (Deb et al. 2002), including:

- *non-dominated* sorting of the population into ranks, such that members of rank *n* dominate members of all ranks >*n*, where rank 1 members constitute the non-dominated set, i.e. the current approximation to the Pareto frontier
- *crowding distance* is used as a secondary discriminator on members of the same rank: members in crowded regions of the population are scored lower, so the surviving members after selection have greater diversity. This helps prevent premature convergence of the population to a small portion of the Pareto frontier
- population members are compared with a *domination* or *constraint-domination* relation — the latter allows for domination comparisons even when constraints are violated

The purpose of the crowding distance calculation is to encourage a spread of solutions that sample the Pareto frontier with as much uniformity as possible. However, one side effect of this is that unacceptable candidates may be preferentially included in the population primarily because they have extreme values in some objectives, and thus remain in an uncrowded region. To alleviate this problem, we have made use of constraints g_j to define acceptability limits on the objectives, thus focusing the evolutionary algorithm on a portion of the Pareto frontier that includes more potentially viable candidates.

With constraints included, the constraint-domination relation is used to compare the parent and trial vectors. This modifies the comparison as follows:

1. If both parent and trial have no constraint violations, then they are evaluated with the domination relation in *objective* space.
2. If one of the parent or trial vector has no constraint violations, then that one is selected.
3. If both parent and trial have constraint violations, then they are evaluated with the domination relation in *constraint violation* space.

Note that one effect of this comparison technique is that candidates with constraint violations are compared without regard for objective values, i.e. finding feasible solutions is regarded as primary, and if none can be found, the population will settle to a Pareto frontier in constraint violation space.

System Architecture

The system architecture for the experiments integrates existing components (see Figure 2). The Java-based GDE3 component is the multi-objective evolutionary algorithm driver (Johnston 2006). The Lisp based SPIKE system has a model of the JWST scheduling domain. The evolver component sends SPIKE decision vectors that are used to create schedules and to return objective function values. The systems communicate via socket connections.

SPIKE (Johnston and Miller, 1994) is a planning and scheduling tool kit that was created for use on the Hubble Space Telescope. SPIKE has several built-in scheduling strategies and provides templates for creating new strategies. The system supports iterative repair search algorithms. The scheduler first makes an initial guess that assigns a start time to all selected observation, possibly assigning observations to conflicting times. In the repair stage, SPIKE tries to reduce the number of conflicts by re-assigning the start time of conflicted observations. At the end of the repair stage, SPIKE removes the assignments for observations with existing conflicts to produce a conflict-free schedule. A simple set of gap filling routines were designed for the experiments below.

One of the innovations introduced in the architecture is the capability to parallelize schedule evaluations, in order to take advantage of available computing resources to speed the elapsed runtime of the algorithm. This is indicated in Fig. 1, where the GDE3 driver can invoke multiple instances of SPIKE on different hosts to perform schedule evaluations, which is the dominating component of the overall runtime.

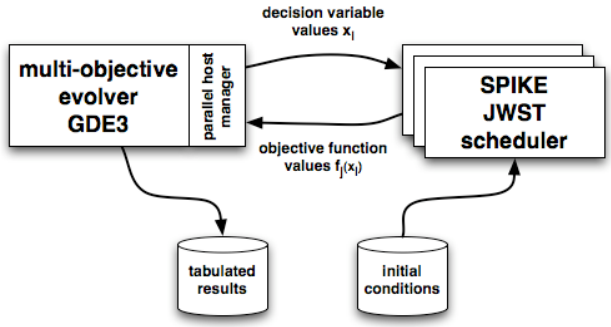


Figure 1: System Architecture

Evaluating Multi-Objective Algorithms

In this paper we evaluate how variants of evolutionary algorithms perform when scheduling JWST. For a given problem each algorithm outputs a Pareto surface. This led us to investigate how Pareto surfaces can be compared in order to determine which run of an algorithm is best. (Zitzler 2003) contrasts unary Pareto surface evaluation functions that measure the quality of a single Pareto surface with binary evaluation functions that compare the quality of two Pareto surfaces. He shows that no unary function or combination of unary functions can tell whether or not a Pareto surface P_1 is strictly better than a Pareto surface P_2. In contrast binary evaluation functions can be designed to tell if one surface is better than another. For example, using the comparison operator from (Giuliano and Johnston 2008) first construct the combined Pareto frontier of the two surfaces Combined(P_1,P_2). If Intersect(P_1,Combined(P_1,P_2)) == P_1 and Intersect(P_2,Combined(P_1,P_2)) == null then surface P_1 dominates surface P_2.

If there is no strict domination between surfaces then we cannot determine if one surface is better than another. The purpose of a multi-objective algorithm is to produce a

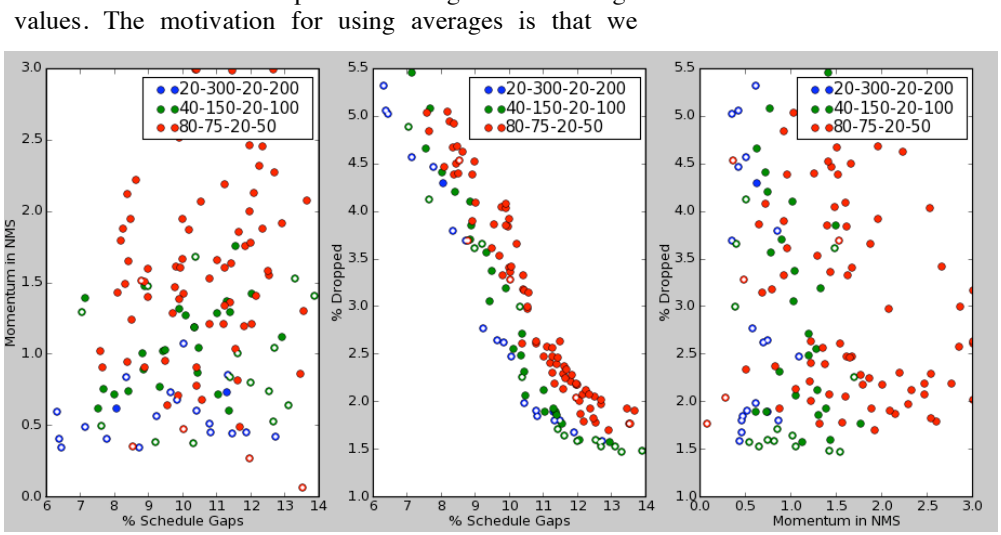
surface of Pareto optimal solutions so that a user can explore the space and ultimately select one of the solutions for execution. If there is no strict domination between a pair of surfaces then the combined surface will contain elements from both sets and the end user may prefer an element from either set. Without additional knowledge no formulation can distinguish between pairs of non-dominating surfaces. In this way the selection of a multi-objective algorithm is like the selection of a solution out of a Pareto optimal frontier. Since there are multiple objectives there is no formal way to make the selection if there is no dominating solution.

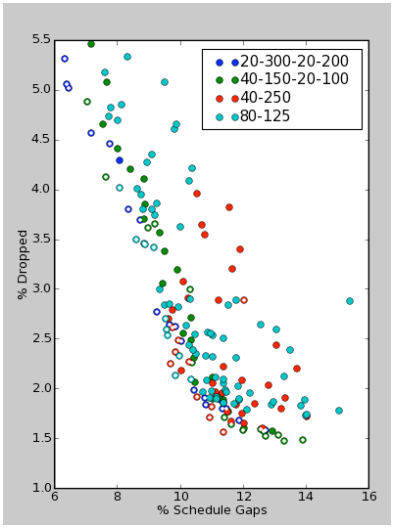
Despite these pessimistic truths analysts who code and set parameters for multi-objective algorithms need tools for distinguishing between algorithms beyond the notion of strict dominance. In this paper we explore some novel formulations that measure the quality of a Pareto surface and show how the formulations do and do not match the intuitions obtained from plots of schedule values. Suppose we want to evaluate N Pareto surfaces P_1,, P_n. Binary evaluation functions allow the determination of domination between surfaces but are cumbersome because they require examining a metric value for each ordered pair of surfaces. In contrast, unary metrics are easy to understand as they give an absolute value per surface but do not provide the ability to determine domination relationships. Using the notion of a combined Pareto surface we construct metrics that make binary metrics appear as unary metrics and that make unary metrics relative to a baseline ideal solution set. The idea is to construct the combined Pareto surface of the N solutions P_1, .. P_n. This combined surface represents the ideal best obtainable solutions given the algorithms being compared. Unary metrics can then be compared to the corresponding metric of the combined surface. Likewise, binary evaluation functions for an individual P_i can be run against the combined surface giving a single value indicating how the surface compares against the best possible.

We utilize the following unary metrics. For each of the three criteria we report the average value of all solutions for the criteria and the minimum obtainable value for the criteria. In addition we report the average of the average values. The motivation for using averages is that we prefer solutions that minimize the overall criteria scores. We use minimum values as this shows the best obtainable solution for the criteria. Each of these unary metrics is then compared with the corresponding value in the combined surface. In this paper two binary evaluation functions are modified to give a unary value based on the combined surface. One is based on the E-indicator (Zitzler 2003) which gives the factor by which one Pareto surface is worse than another with respect to all objectives. In other words the value is the minimum factor *e* such that for any solution in P_2 there exists a solution in P_1 that is not worse by a factor of *e* in all objectives. The second binary indicator is based on the naïve comparison operator in (Giuliano and Johnston, 2008). This comparison is called the P-indicator and is the fraction of P_1 that occurs in the combined surface. In general smaller $E(P_1,P_2)$ means that P_1 is preferable to P_2 and larger $P(P_1,P_2)$ values mean that P_1 is preferable. Both of these binary metrics can be used to determine if one surface dominates another.

Experimental Setup

A series of experiments are presented which evaluate different approaches for JWST scheduling and explore how multi-objective algorithm features such as the use of constraints, population size, and the number of generations impact the solutions. The results will be presented using graphs and the evaluation techniques given in the previous section. The initial experiments make use of the "all at once" search strategy, which optimizes both observation timing and spacecraft roll angle in a single step. Although this approach did not fare well in previous studies, it is included in this study, since schedule and roll determination are more tightly coupled in the enriched scheduling model. We also utilize the "delayed roll" approach from previous studies which first generates a Pareto frontier considering scheduled times only (with a nominal time delay to account for roll changes), followed by a separate optimization step which adjusts roll only. We expect this approach to fare better when schedule and roll assignments can be decoupled.





Figures 2 & 3. Compare alternative search approaches by plotting the Pareto optimal surface generated for a 22-day schedule for each pair of metrics. The hollow data points are those in the Combined Pareto Optimal surface.

There is a significant difference in computational cost for schedule time evaluations as compared to roll evaluations: the former take on the order of 20 seconds per schedule, while roll evaluations alone take only a fraction of a second. This difference led us to devise two different approaches to parallelizing the algorithm. The main loop of a multi-objective algorithm can be coded as:

```
For (gen = 1; gen <= Search Depth; gen++)
    For (i = 1;  I <= N;   i++)
        Create and evaluate a candidate vectors
    Evolve the population based on the new evaluations
```

The inner loop of this algorithm performs the time consuming schedule evaluations and is parallelized in both the all at once and delayed roll search approaches by distributing the evaluations evenly to available processors. Since all evaluations must be done before evolving the next generation this allows parallelism up to the population size N. In the delayed-roll approach after first solving the 2-objective scheduling problem the system distributes the roll search for each of the N members of the resulting population to the available processors. For a roll search the candidate evaluation cost is too small to parallelize the inner loop. In both cases, the parallelism is effective in reducing runtime, up to the point that the number of processors is larger than the population size *N*.

All of the experiments create a schedule for a 22-day station-keeping bin using a set of observations selected from a simulated long range planning process. Unless otherwise noted the experiments were sized to run in an approximately 5 hour window using 10 processors. This duration was selected such that an analyst could set-up, run, and analyze results during an 8 hour work day

Experimental Results

An initial set of experiments ran 3 different combinations of population size versus number of generations using an all at once search. The combinations ranged from a narrow but deep search with a population size of 20 and 500 generations to a broad but shallow search with a population size of 80 and 125 generations. All of the approaches evaluate 10,000 candidate solutions. After completion of the experiments we found that many of the solutions were not acceptable in that individual criteria were too high to ever be selected for execution. We then defined conditions for an acceptable solution as follows: Gaps <= 3.5, Momentum <= 24, Dropped <= 5.5. Considering these acceptability conditions we found that only a small fraction of the solutions generated were acceptable. For the 20-500 run no solutions are acceptable and for others the best is 40% acceptability. The experiments were re-executed this time using the acceptability criteria as constraints in the multi-objective algorithm. Metrics evaluating Pareto surfaces are given in Table 1 for all of the experiments described in the paper. The table is divided into sections based on experiments. We report the raw unary metrics to the left of the /, the unary metrics compared to the same metric from the combined Pareto surface to the right of the /, and the binary metrics normalized to the combined Pareto surface. The results for the runs with constraints are shown in the **All at Once** section of Table 1. The results show that a broad but shallow search is preferred over a narrow but deep search. Although the data is not shown, the same result applies to the all at once search without constraints. All of the metrics improve as the population size increases except to the e-indicator. The next section of Table 1 contrasts the all at once search with and without constraints. As can be seen the search with constraints performed uniformly better than the search without constraints. With constraints all of the solutions were acceptable and all metrics favor the run with constraints. Constraints focus the search making all solutions acceptable and reducing metric values.

Delayed Roll Search A series of experiments were then run to determine the impact of constraints, the impact of depth versus breadth on the search tree, and whether or not the delayed roll search approach would perform better than the all at once search. Since the updated scheduling model includes slews between observations, that are in part based upon the roll, the JWST schedule evaluation engine was augmented to assume a worst case roll between observations that are not forced to be at the same roll due to same-orient constraints. The delayed roll search was run with three different parameter settings that can be described by four numbers N-D-M-P indicating respectively the base population size N, the schedule search depth D, the roll search population size M (always set to 20), and the roll search depth P. For example 20-300-20-200 indicates using a population size of 20 to perform a 300 generation schedule assignment search resulting in a population of 20 schedule solutions. After the schedule search completes the system performs a 200 generation roll search with population size 20 for each of the 20 schedule solutions. This search generates 20x300 = 6000 schedule candidates and 20 X 20 X 200 = 80,000 roll candidates. The three experiments executed all generate 6000 schedule candidates and 80,000 roll candidates and can be described as: 20-300-20-200, 40-150-20-100, and 80-75-20-50.

The results for these runs are displayed in the **Delayed Roll Search** section of Table 1 and are shown pictorially in Figure 2. First we note that in this search the narrow but deep search produces the best results. This can be seen in the table in terms of the metric trends and pictorially in Figure 2 where the two lower population size searches dominate in just about all the metrics. Again the exception to the metrics is the e-indicator. The value of the e-indicator favors the high population size 80-75 - 20-50 search. The e-indicator is based on a worst case factor and is determined by a single point on the combined surface (it is the point in the bottom right corner of the leftmost panel in Figure 2). Removing this point makes the e-indicator even among the surfaces. Although this indicator may be useful in determining domination it does not agree with intuitions on the

qualities of one surface versus another. We compare the two best all at once searches with the two best delayed roll searches in the next section of Table 1. The delayed roll search has lower averages throughout and performs better on all of the metrics. The quality of the all at once search is especially bad with respect to momentum. Except for one point all of the points contributed to the Combined Pareto surface from the all at once searches are worse with respect to momentum than the points contributed by the delayed roll search. If we just consider a two criteria search without momentum, we see that the all at once search is competitive. Figure 3 plots schedule gaps versus dropped observations showing that the all at once search contributes to a large portion of the combined Pareto surface. What is surprising is that the delayed roll search performed so well with respect to gaps. In these experiments the schedule and roll search steps are decomposed by ensuring that there is always enough time to allow for the largest roll slew. This might be expected to create gaps as the actual roll slew can be smaller than the maximum of 6 degrees. The results can be explained as follows. Slew distance is a combination of the roll distance and the target slew distance, The roll component is zero for same orient sets. In general a schedule that reduces gaps will have observations from a same orient set contiguous in the schedule as all of the observations have the same target. This reduces gap time independent of roll considerations. Observations in different same orient sets generally have different targets and therefore will have a non zero target slew component. The target slew will generally be larger than six degrees and will dominate the total slew time.

Impact of Parallelism The final experiment measures the impact that parallel execution can have on the quality of the solutions generated. The experiments so far have measured the cost of an experimental run in terms of the total number of schedule and roll evaluations. With this model a search with a population size of 20 and a depth of 200 costs the same as a search with a population size of 80 and a depth of 50. Another cost to consider is the runtime of the search. With parallelism we get almost perfect speedup up to the population size. By increasing the population size and keeping a constant search depth we can keep the runtime the same but get more schedule evaluations. We ran an experiment measuring this impact for the delayed roll search. As a baseline we ran a search with a population size of 20 and a depth of 200 with a 20 X 100 roll search. We then ran two additional searches using population sizes of 40 and 80. The data for the runs are presented in the **Impact of Parallelism Section** of Table 1 and visually in Figure 4. The table data shows that the runs with the increased population sizes improve the gaps and momentum criteria. Looking at figure 5 additional structure can be seen in the results. The middle pane shows that the schedule gaps for the two larger generation searches diverge from the frontier for the lower population size search. Although the 80-200 search falls on the same line as the 40-200 search it is better sampled at the extreme minimum. These observations show the importance of using both numeric metrics and graphical data to evaluate algorithm performance.

Conclusions

Our investigations show that, for the JWST scheduling problem, the delayed roll search still remains the best overall approach, in spite of our higher fidelity spacecraft maneuvering model that makes the problem less decomposable than before. The other conclusions to be drawn from the experiments reported here include:

- the use of constraints to "focus" the Pareto frontier to a region containing a higher density of less extreme solutions is an effective technique that could be applied to other problems
- for the delayed roll search, increasing the population size for a fixed number of generations gives better results in the same elapsed time

On the other hand, there is, in general, no clear advantage either to larger populations sizes or to running more generations, for a fixed overall investment in computational resources. Further research will be required to clarify the advantages of one tactic over the other.

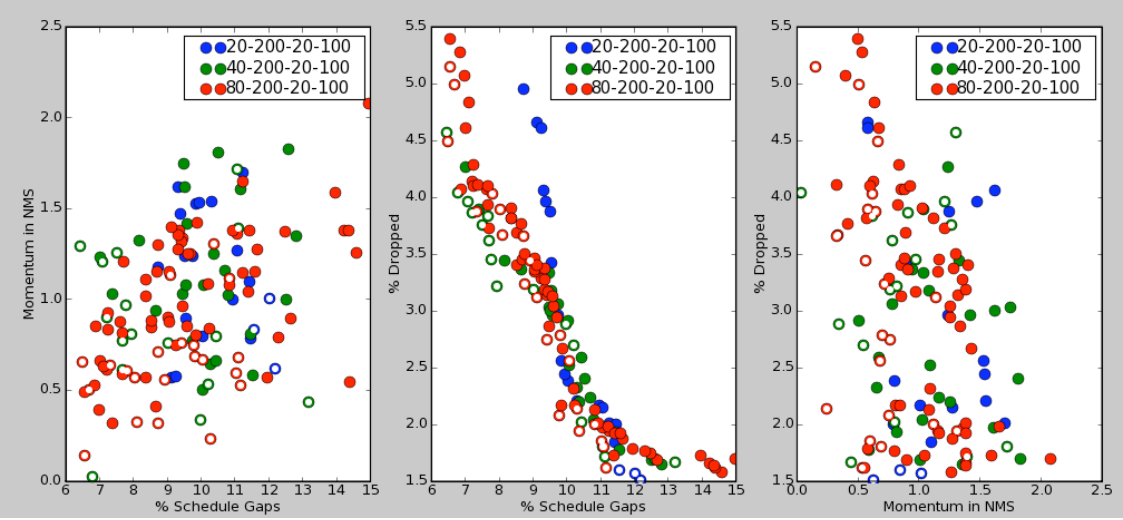


Figure 4. Compares alternative search approaches by plotting the Pareto optimal surface generated for a 22-day schedule for each

		Avg	Avg-Gap	Avg-Mom	Avg-Dropped	Min-Gap	Min-Mom	Min-Dropped	E-val	P-Val
All At Once With Constraints	***COMBINED***	4.19	2.33	7.57	2.69	1.67	0.27	1.54	-	-
	20-500	5.41/1.21	2.46/0.13	10.49/2.93	3.27/0.59	1.78/0.11	0.54/0.27	1.54/0.00	2.01	0.04
	40-250	4.21/0.02	2.49/0.16	7.78/0.21	2.37/-0.32	2.11/0.44	0.27/0.00	1.57/0.03	1.28	0.32
	80-125	4.33/0.13	2.34/0.01	7.76/0.20	2.88/0.20	1.67/0.00	0.77/0.50	1.74/0.20	2.85	0.64
All At Once With and without constraints	***COMBINED***	4.29/0.00	2.22/0.00	7.57/0.00	3.09/0.00	1.43/0.00	0.77/0.00	1.70/0.00	1	1
	80-125	5.20/0.91	2.26/0.04	10.04/2.47	3.31/0.22	1.43/0.00	0.81/0.04	1.70/0.00	1.51	0.24
	80-125	4.33/0.04	2.24/0.02	7.76/0.20	2.88/-.20	1.67/0.24	0.77/0.00	1.74/0.04	1.23	0.76
Delayed Roll Search	***COMBINED***	1.92	2.24	0.72	2.8	1.38	0.07	1.48	-	-
	20-300-20-200	1.91/-0.01	2.07/-0.18	0.58/-0.14	3.07/0.27	1.38/0.00	0.35/0.27	1.59/0.11	4.78	0.47
	40-150-20-100	2.03/0.11	2.27/0.03	1.03/0.31	2.79/-0.01	1.54/0.16	0.38/0.31	1.48/0.00	5.26	0.39
	80-75-20-50	2.35/0.43	2.34/0.09	1.64/0.92	3.07/0.27	1.66/0.28	0.07/0.00	1.71/0.23	1.36	0.13
Delayed Roll search Versus All at Once	***COMBINED***	2.84	2.25	3.66	2.62	1.38	0.27	1.48	-	-
	20-300-20-200	1.91/-0.93	2.07/-0.18	0.58/-3.08	3.07/0.45	1.38/0.00	0.35/0.08	1.59/0.11	1.28	0.4
	40-150-20-100	2.03/-0.81	2.27/0.03	1.03/-2.63	2.79/0.17	1.54/0.16	0.38/0.11	1.48/0.00	1.43	0.33
	20-500-0-0	5.41/2.57	2.46/0.21	10.49/6.84	3.27/0.65	1.78/0.40	0.54/0.27	1.54/0.06	2.07	0
	40-250-0-0	4.21/1.37	2.49/0.24	7.78/4.12	2.37/-0.25	2.11/0.73	0.27/0.00	1.57/0.09	1.9	0.27
Impact of parallelism	***COMBINED***	1.92	2.24	0.72	2.8	1.38	0.07	1.48	-	-
	20-300-20-200	1.91/-0.01	2.07/-0.18	0.58/-0.14	3.07/0.27	1.38/0.00	0.35/0.27	1.59/0.11	4.78	0.47
	40-150-20-100	2.03/0.11	2.27/0.03	1.03/0.31	2.79/-0.01	1.54/0.16	0.38/0.31	1.48/0.00	5.26	0.39
	80-75-20-50	2.35/0.43	2.34/0.09	1.64/0.92	3.07/0.27	1.66/0.28	0.07/0.00	1.71/0.23	1.36	0.13

Table 1: Gives metric evaluation values for five different experiments. The metrics are described in the text above

We have noted that the selection of a "best" multi-objective algorithm can itself be viewed as a multi-objective optimization problem, and that comparing different algorithms in terms of the Pareto frontiers that they generate is in general quite difficult. In the fortunate but rare case that one approach dominates the others, then the conclusions are straightforward. However, in the more common case that there are different regimes where different approaches do better, we have investigated several metrics for assessing their relative performance. While these metrics can provide useful insight into algorithm performance, we have found that it is frequently the case that a graphical view of the population in objective space reveals patterns and trends that are not captured by the current metrics. These graphical views may be of more help to end users than most of the metrics we have explored.

Future work includes additional investigations of the population size versus number of generations tradeoff, and of visualization techniques that will prove essential for more than 3 objectives. One additional objective we are considering is schedule robustness which can be used to manage the risk of schedule disruptions.

Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration, and at the Space Telescope Science Institute under the NASA Applied Information Systems Research Program grant number NNX07AV67G.

References

Abraham, A., L. Jain and R. Goldberg (2005). *Evolutionary Multiobjective Optimization*. Berlin, Springer.

Deb, K., Multi-Objective Optimization Using Evolutionary Algorithms. (2001), New York: John Wiley & Sons.Deb, K., A. Pratap, S. Agrawal and T. Meyarivan (2002). "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II." IEEE Transactions on Evolutionary Computation **6**(2): 182-197.

Giuliano, M. 1998. Achieving Stable Observing Schedules in an Unstable World. In *Astronomical Data Analysis Software and Systems VII*. 271-274.

Giuliano, M., Rager, R., Ferdous, N. (2007) Towards a Heuristic for Scheduling the James Webb Space Telescope. In ICAPS. Providence, Rhode Island. 160-167.

Johnston, M. and Miller, G. 1994. Spike: Intelligent Scheduling of Hubble Space Telescope Observations. In Zweben M. and Fox M. eds. *Intelligent Scheduling*, 391-422. Morgan-Kaufmann.

Giuliano, M,, and Johnston M. D., (2008). "Multi-Objective Evolutionary Algorithms for Scheduling the James Webb Space Telescope", In ICAPS, Sydney Australia. 107-115.

Johnston, M. D. (2006). "Multi-Objective Scheduling for NASA's Deep Space Network Array." In IWPSS-06. Baltimore, MD, Space Telescope Science Institute.

Johnston, M. D. (2008). "An Evolutionary Algorithm Approach to Multi-Objective Scheduling of Space Network Communications." *International Journal of Intelligent Automation and Soft Computing*: in press.

Kukkonen, S. and J. Lampinen (2005). "GDE3: The Third Evolution Step of Generalized Differential Evolution." The 2005 Congress on Evolutionary Computation.

Laborie, P. 2003. Algorithms for Propagating Resource Constraints in AI Planning & Scheduling: Existing Approaches and New Results. *Artif. Intell*. 143(2):151-188.

Policella, N., Oddi, A., Smith S.F., and Cesta, A. Generating Robust Partial Order Schedules. In *Proc of CP 2004*,Lecture Notes on Computer Science (LNCS) Vol. 3258, pp. 496-511, M. Wallace (Ed.), Springer, 2004.

Price, K., R. Storn and J. Lampinen (2005). *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Springer.

Rager, R. and Giuliano, M. 2006. Evaluating Scheduling Strategies for JWST Momentum Management. In *Proceedings of the 5th International Workshop on Planning and Scheduling for Space, 235-243*.

E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. "Performance Assessment of Multiobjective Optimizers: An Analysis and Review. "*IEEE Transactions on Evolutionary Computation*, 7(2):117–132.