Journal of Intelligent Manufacturing (1999) **10,** 387–404

Generating short-term observation schedules for space mission projects

M. SELIM AKTURK and KEMAL KILIÇ

Department of Industrial Engineering, Bilkent University, 06533 Bilkent, Ankara, Turkey

Received October 1997 and accepted June 1998

In this paper, we propose a new dispatching rule and a set of local search algorithms based on the filtered beam search, GRASP and simulated annealing methodologies to construct short-term observation schedules of space mission projects, mainly for NASA's Hubble Space Telescope (HST). The main features of generating short-term observations of HST are state dependent set up times, user specified deadlines, visibility windows of the targets and the priorities assigned to the observations. The objective of HST scheduling is to maximize the scientific return. We have tested the relative performances of the proposed algorithms including the nearest neighbor rule both in objective function value and computational time aspects by utilizing a full-factorial experimental design.

Keywords: Space mission projects, scheduling, local search algorithms

1. Introduction

Space mission scheduling (SMS) has been an important research area for several years. SMS has a wide area of applications such as scheduling space observatories, coordinating the activities aboard the space station, space shuttle ground processing systems, generating detailed commands for planetary probes and scheduling satellite activities. Space mission projects are costly to build and operate in great demand by researchers in the international astronomical community. It is essential that these facilities be operated as efficiently as possible to maximize their scientific return. Since the total number of scientific requests far exceeds the capabilities of these projects, the goal for the scheduler is to minimize the number of tasks not accomplished by the schedule. The problem we are addressing here is that of constructing short-term observation schedules for the NASA/ESA's Hubble Space Telescope (HST). HST is a unique, \$1.4-billion international space observatory launched in April 1990. As a result of lack of interference by the Earth's atmosphere, the resolution, sensitivity and ultraviolet wavelength coverage of HST are considerably greater than those obtainable with ground-based telescopes. During its nominal lifetime of 15 years HST is expected to significantly increase our understanding of a wide range of astronomical objects and phenomena.

The overall objective of HST scheduling is to efficiently allocate viewing time to competing candidate observation requests in the presence of complex operational constraints, i.e. to maximize the scientific return. Since HST is in low earth orbit, with an altitude of 500 km and an orbital period of 95 minute, most targets are periodically occulted by the earth, and thus they are visible only for a portion of each orbit. Over longer periods targets may be similarly occulted by the moon and the sun. Thus, execution possibilities are limited by target ''visibility windows'', which are known with certainty over short term horizons. Depending on the prior state of the telescope, each of the remaining requirements variably affects when the observation can be executed. It takes time to repoint the telescope toward a different target, an activity referred as slewing. Similarly, it takes time to reconfigure

0956-5515 © 1999 Kluwer Academic Publishers

viewing instruments. There are six viewing instruments onboard, and each is capable of being used in a variety of different configurations. Finally, Space Telescope Science Institute (STScI) assigns different priorities to the observation requests with respect to their relative importances.

There are two different approaches to the SMS problem in the literature. The first one makes a ''single/parallel machine scheduling'' approximation of the problem and uses the traditional operations research tools in the solution methodology, whereas the second approach considers the overall domain and formulates the problem as a constraint satisfaction problem. Fisher and Jaikumar (1978) provide an algorithm for the scheduling of the NASA space shuttle program to determine mission launch times that minimize the number of late missions, where each mission has an earliest and latest start times. A single machine approximation is provided for the problem and the proposed algorithm is inspired by Moore's (Moore, 1968) algorithm for minimizing the number of tardy jobs on a single machine. Hall and Magazine (1994) develop a dynamic programming algorithm for the single machine scheduling problem where each activity has a weight and a single time window, i.e. a specific time interval that the execution of the activity is allowed. However these formulations do not consider the sequence and state-dependent setup times required for the execution of each activity. Furthermore, there is usually a single time window for each activity, which may not be a realistic assumption for the SMS problem in many cases.

It is possible to visualize HST scheduling problem as $1||\sum w_jU_j$ problem with state-dependent setup times. $1||\sum w_jU_j$ problem is NP-hard in the ordinary sense, and Lawler and Moore (1969) present a pseudopolynomial dynamic programing algorithm to solve it. Potts and Wassenhove (1988) propose a branch-and-bound algorithm that reduces the size of the search tree with dominance reductions. Hochbaum and Landy (1994) show that the weighted number of tardy jobs with batch setup problem is NP-complete and propose a pseudopolynomial algorithm. Furthermore, the visibility windows can be viewed as due-windows of the jobs. Lann and Mosheiov (1996) study due-windows with the objective of minimizing the number of early and tardy jobs.

Vehicle routing scheduling and planning with time windows (VRSPTW) problems share many common features with the SMS problems. A typical vehicle routing planning (VRP) problem is to find the minimum costing routes for a fleet of vehicles that serves to a set of customers with fixed demand. Desrochers *et al.* (1990) provide a comprehensive classification scheme for vehicle routing planning and scheduling problems. Laporte (1992) represents an overview of the main exact and approximate algorithms to VRP, and Desrosiers *et al.* (1994) provide an extensive overview on time constrained routing and scheduling. In the VRSPTW problem there is an additional time constraint associated with each customer such that each customer has a time window which starts with an allowable earliest start time and ends with an allowable latest start time. For one vehicle and multiple time windows associated with the customers, the problem turns out to be a good approximation of the HST domain. One vehicle represents the HST, each customer of the vehicle can be viewed as an observation request and multiple time windows associated with the customer can be viewed as the visibility windows of the observations. Furthermore, the time required between two customers is sequence dependent. However the VRSPTW literature considers a single time window and mostly tries to find the minimum costing route for a set of customers with equal priorities, whereas our objective is to minimize the number of unvisited customers within a time horizon.

The second approach to the SMS is to construct software architectures, which use constraint-directed search scheduling methodology. A Fortran-based software, science operations ground system (SOGS), is developed by TRW in order to support the astronomers when planning and scheduling HST. Science planning and scheduling system (SPSS) is the major tool of SOGS, which is designed to produce executable and detailed schedules from the approved viewing proposals. SOGS has had several problems due in part to the complexity of the scheduling problem and the constraints that must be taken into account, and in part to the programming methods and computational infeasibility that emerges from the non-hierarchical nature of the solution approach as discussed by Waldrop (1989). These important shortcomings of SOGS lead STScI to find out new solution procedures to the planning and scheduling problem of HST.

An artificial neural network, called SPIKE, is developed by Johnston and Adorf (1992) to overcome these shortcomings and augmented to the system. The major contribution of the SPIKE was partitioning the problem into two parts; long-term schedule and short-term schedule. This is a quite logical approach for the domains that are complex and have highly interactive nature such as HST scheduling. Moreover, orbital constraints loose certainty on longer horizons. SPIKE is currently used as a long-term scheduling tool that partitions the approved observations into weekly or smaller buckets which in turn becomes the input to SPSS for generating a detailed short-term schedule. Minton *et al.* (1992) propose a heuristic repair-based min-conflict algorithm that can be used with the SPIKE system to minimize the search effort, and show that it is very easy to implement and at least an order of magnitude faster than the guarded discrete stochastic network of Johnston and Adorf.

The second major research direction is the heuristics scheduling testbed system (HSTS) discussed in Muscettola *et al.* (1992). HSTS is a software architecture indicating how to model the structure and dynamics of a system, and how to represent schedules at two levels of abstraction, namely abstract and detailed levels, in the temporal data base. The abstract level is responsible for the generation of initial observation sequences by taking into account the telescope availability, overall telescope reconfiguration and target visibility windows, whereas the detailed level is responsible for determining the executable and detailed schedules of HST. Even though the detailed level generates the schedules that are executable for the HST, abstract level is the main stage that guides the detailed schedules. Therefore, it is important to have a good sequencing methodology at the abstract level. Smith and Pathak (1991) proposes three strategies for the abstract level of HSTS. The first strategy is a dispatch-based methodology, namely the nearest neighbor (NN) algorithm. The second strategy, the most-constrained first (MCF), focuses on maximizing the number of scheduled observation programs and tries to add the observation with the fewest number of allowable start times. Moreover a third strategy is suggested to balance both of the objectives of maximizing the utilization of HST and maximizing the number of scheduled observation programs, namely the MCF/NN. However, these methodologies are relatively simple and myopic. In fact, developing a more sophisticated scheduling methodology that considers the different priorities of the observation programs is mentioned as a future research direction.

2. Problem statement

The HST scheduling problem is to efficiently allocate viewing time to competing candidate observation requests in the presence of complex operational constraints and priorities associated with the observation requests. Our objective is to maximize the scientific return, that can be stated as to maximize the number of requests that can be viewed during the planning horizon, or equivalently to minimize the number of rejected requests. In brief, a candidate observation represents a user request for an exposure of a certain duration of a particular celestial object using a particular viewing instrument in a particular operational configuration as discussed in Johnston (1987) and STScI (1986). Consider the following example.

Example: Take a picture of Global Cluster NGC 7078, angular distances of DEC = 12.167 and RA = 322.492, before June 14, 1998. The picture has to be taken with the wide field camera in normal configuration. The exposure must have a duration of 1 minute.

STScI receives these requests and assigns priorities to the approved observation requests considering their scientific value and operational efficiency as follows:

(i) ''high-priority'' observations (nearly 20% of the estimated available time)

(ii) ''medium-priority'' observations (nearly 70% of the estimated available time)

(iii) ''supplemental-pool'' observations (nearly 30–50% of the estimated available time).

There are six different viewing instruments on HST, namely wide-field/planetary camera (WF/PC), faint object camera (FOC), faint object spectrograph (FOS), high resolution spectrograph (HRS), and high speed photometer (HSP). The fine guidance system (FGS) of the telescope is also used for astronomic observations. Each instrument can be used with several different configurations as summarized in Table 1. The last column of Table 1 presents the expected percentages of the observations that require the specific instrument. Spacecraft power and thermal balance constraints limit the number of instruments that can be operational at any point, which requires execution of complex power-up/ power-down sequences as changeovers are made

Table 1. The viewing instruments of HST and their possible configurations

Instrument	*Configuration*	*Mode*	*Percent*
WF/PC	WF	N	35%
	WF	UV	
	PC	N	
	PC	UV	
FOC	/48		25%
	/96		
	/288		
FOS	BL		10%
	RD		
HRS			10%
HSP	PHOT		15%
	PMT		
	PRISM		
	POL		
FGS			5%

from one instrument to another. These changeovers can be between instruments, referred as the major reconfiguration, or can be mode changes of the instruments, referred as the minor reconfiguration.

In order to start an execution, HST must be pointing at the target. This can be achieved by slewing the telescope from its previous direction to its new direction. The slewing duration mainly depends on the slewing angle between two locations and the angular slewing velocity. Two angular distances namely the declination (DEC) and right ascension (RA), both in radians, are used to specify the position of the celestial object on the coordinate system. In order to calculate the slewing time between two celestial objects the well known cosine formula is used. Suppose that HST was previously picturing the target f and next will take the pictures of the target t, then the slewing time of the telescope is calculated as follows. Let angular slewing velocity $= 0.0017453294$, $rel-RA = |RA_f - RA_t|$, $b = \frac{\pi}{2} - |DEC_f|$, and if $signum\ DEC_f = signum\ DEC_t$ then $c = \frac{\pi}{2} - |DEC_t|$, else $c = \frac{\pi}{2} + |DEC_t|$.

$$\text{Slew time from } f \text{ to } t = \frac{\arccos[\cos(b) \times \cos(c) + \sin(b) \times \sin(c) \times \cos(rel-RA)]}{\text{angular slewing velocity}}$$

Since HST is in low earth orbit, the execution possibilities are limited by target visibility windows. There are also high-radiation regions over the South Atlantic where the instruments cannot be operated. The visibility windows of two celestial targets A and B are presented in Fig. 1 as an example. Periodically each target has an interval that it is visible and consecutively an interval that it is not visible. As we can see from Fig. 1, the exposure of the celestial target B can only start if it is in its visibility window, the required instrument is active and the telescope is repointed. We will refer to the time that is spent after maximum of reconfiguration and slewing times until the next scheduled observation is visible again as the idle time. Finally, an astronomer may specify a specific deadline for an observation request.

The following assumptions are made to define the scope of this paper. The reconfiguration time is implicitly accounted as temporal time delay rather than explicitly modeled it as complex power-up/power-down sequences. Only one instrument can be operational at any time because of the limited power on board. Reconfiguration of the instruments can be managed simultaneously with the slewing of the telescope, hence the maximum time of both is used as the overall set-up of HST. All of the mentioned factors that limit the size of visibility windows will be implicitly handled at once in a single visibility window for each target that specifies the available time for an exposure of that particular target at each orbit. These assumptions are very similar to the ones that are available in the literature on the HST scheduling problem, such as Muscettola *et al.* (1992) and Smith and Pathak (1991).

The notation used in the proposed mathematical model is as follows:

Parameters:

b_{vj}	=	Beginning time of a visibility window v for an observation j
D_j	=	Specific deadline for an observation request j (i.e. $D_j \leq T$)
e_{vj}	=	Ending time of a visibility window v for an observation j
M	=	Very large positive number
N	=	Total number of observation requests
P_j	=	Requested viewing time of an observation j
SC_{ji}	=	Instrument reconfiguration time from observation request j to i
SL_{ji}	=	Slewing time from observation request j to i
T	=	Length of the planning horizon
w_j	=	Relative priority of an observation request j

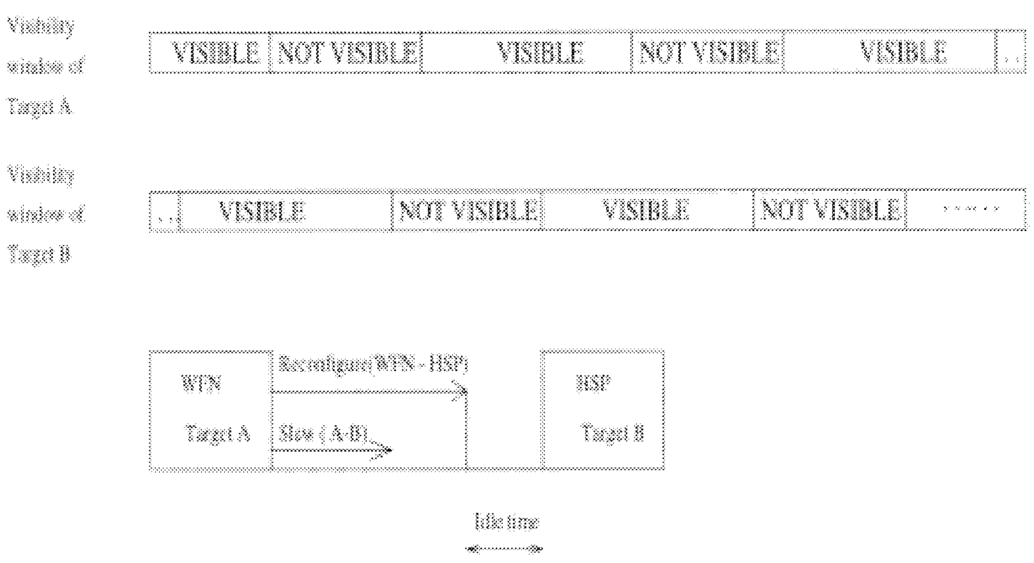

Fig. 1. Visibility windows.

Decision variables:

S_{jk}	=	Starting time of an observation request j at sequence k
x_j	=	0–1 binary variable which is equal to 1 if an observation request j is rejected
y_{jk}	=	0–1 binary variable which is equal to 1 if an observation request j is scheduled at sequence k
z_{jv}	=	0–1 binary variable which is equal to 1 if an observation request j is scheduled during the time window v

A mathematical formulation of the problem can be given as follows:

$$\text{Minimize } \sum_{j=1}^{N} w_j \cdot x_j$$

$$\text{Subject to } S_{jk} \geq b_{jv} \cdot z_{jv} \qquad \forall j,k,v \quad (1)$$

$$S_{jk} + P_j \leq e_{jv} + M(1 - z_{jv}) \qquad \forall j,k,v \quad (2)$$

$$\sum_{v=1}^{V} z_{jv} = 1 \qquad \forall j \quad (3)$$

$$S_{ik} - (S_{j(k-1)} + P_j) \geq SC_{ji} \qquad \forall i,j,k \quad (4)$$

$$S_{ik} - (S_{j(k-1)} + P_j) \geq SL_{ji} \qquad \forall i,j,k \quad (5)$$

$$S_{jk} + P_j - D_j \leq M \cdot x_j \qquad \forall j,k \quad (6)$$

$$S_{jk} \leq M \cdot y_{jk} \qquad \forall j,k \quad (7)$$

$$\sum_{j=1}^{N} y_{jk} = 1 \qquad \forall k \quad (8)$$

$$\sum_{k=1}^{N} y_{jk} = 1 \qquad \forall j \quad (9)$$

$$S_{jk} \geq 0, \text{ and } x_j, y_{jk}, z_{jv} = 0, 1 \qquad \forall j,k,v \quad (10)$$

In this formulation, objective function corresponds to maximizing the scientific return, or equivalently minimizing the number of rejected observation requests. Constraint sets (1), (2), and (3) ensure that the observation requests can only be started and completed when they are visible. Constraint sets (4) and (5) calculate the time required to go from one observation request to another which is the maximum of the instrument reconfiguration time and the slewing time. Constraint set (6) finds the number of rejected observation requests. Constraint sets (7), (8), and (9) ensure that two observation requests are not scheduled to use the HST at the same time, and no observation requests can be scheduled during either slewing or instrument reconfiguration time. Constraint set (10) gives the nonnegativity and integrality requirements for the decision variables.

3. Algorithms

Most of the real-life scheduling problems, as well as SMS problem, cannot be solved with exact algorithms in a reasonable computational time because of their complex nature. Therefore heuristics are developed to find not necessarily the optimal but a good solution to such problems. Several features that demonstrate the effectiveness of these search heuristics are their ability to adapt to a particular realization, avoid entrapment at local optima and exploit the basic structure of the problem. In this paper, we will propose a new dispatching rule and local search algorithms utilizing filtered beam search, simulated annealing and GRASP methodologies for generating short term observation schedules of SM projects along with the nearest neighbor algorithm that we have used while testing the efficiencies of the proposed algorithms.

The additional notation used in the proposed algorithms is as follows:

ct_{l_t}	=	Completion time of the last scheduled observation l at iteration t
ct_{jt}	=	Completion time of observation j if scheduled at iteration t
$FAST_{jt}$	=	First available start time of observation j at iteration t
l_t	=	Last scheduled observation at iteration t
$Score_c$	=	Objective function value of a given schedule c
$slack_{jt}$	=	The remaining time available to schedule observation j after iteration t
st_{jt}	=	End of setup time of observation j if scheduled at iteration t
S_t	=	Set of observations that are scheduled until the iteration t
TW_{vj}	=	Visibility window interval v for an observation j
U_t	=	Set of observations that are not scheduled until the iteration t
GF_{jt}	=	Global ranking index of observation j at iteration t
L_{jb}	=	Local ranking index of observation j for the partial schedule of beam b
CNS_b	=	Set of observations that cannot be scheduled at partial schedule of beam b because of deadline restrictions
GES_{jb}	=	Global evaluation function value of the augmented partial schedule obtained by adding observation j to the end of the partial schedule b
BS	=	Best schedule
CN	=	Candidate neighbor obtained after each exchange
CS_t	=	Current schedule at iteration t
frozen	=	Boolean variable, and if it is false then repeat the search, else end
mp	=	Probability of mutating the current schedule
T_t	=	Temperature at iteration t, i.e. $T_t = \alpha * T_{t-1}$ and α = rate of decrease
β	=	The rate used when constructing restricted candidate list (RCL)
RCL_t	=	Restricted candidate list at iteration t
MT	=	Maximum number of iterations

3.1. *Nearest neighbor algorithm*

The nearest neighbor (NN) rule is used by Smith and Pathak (1991) for scheduling the over-subscribed systems such as SMS problems to obtain a high resource utilization. The NN rule is a well-known, simple, computationally fast, dispatch based algorithm widely used for solving traveling salesperson problems. The basic idea is to select the first available candidate for the next step. An outline of the NN rule can be given as follows:

1. Calculate $st_{jt} = \max\{SC_{l_t j}, SL_{l_t j}\} + ct_{l_t},\ \forall j \in U_t$
2. For any observation $j \in U_t$ calculate $FAST_{jt}$. There are two possible cases,
Either $FAST_{jt} = st_{jt}$ if $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$
Or $FAST_{jt} = b_{v^*j}$ if $st_{jt} \notin TW_{vj}\ \forall v \in V$ and $v^* = \min\{v | b_{vj} > st_{jt}\}$
3. Select the observation j^* that has the earliest $FAST_{jt}$ and schedule it. Set $l_t = j^*$, $ct_{l_t} = FAST_{j^*t} + P_{j^*}, t = t + 1, U_t = U_{t-1} - \{j^*\}$ and $S_t = S_{t-1} + \{j^*\}$. Goto step 1 until $ct_{l_t} \geq T$

In step 2, either the end of setup time of the observation j is in the visibility window of observation j or not. In the first case, $FAST_{jt}$ is equal to the end of required setup time, however in the second case to start the observation j we must wait until it is visible again.

3.2. *New dispatching rule*

We propose a new composite dispatching rule that combines the weighted shortest processing time (WSPT), nearest neighbor, and min-slack rules. Under the proposed rule, observations are scheduled one at a time; that is, every time the telescope becomes free, a ranking index is computed for each remaining observation. The observation with the highest ranking index is then selected to be scheduled next. This ranking index is a function of the time t at which the telescope became free as well as the set up time, visibility windows, P_j, w_j and D_j of the remaining observations. This index is defined as

$$\pi_j(t) = \frac{w_j}{P_j} * \exp\left(\frac{-(FAST_{jt} - ct_{l_t})}{k_1}\right) * \exp\left(\frac{-slack_{jt}}{k_2}\right)$$

where $k_1 = c/(2 * \sqrt{\bar{s}/\bar{p}})$, c and k_2 are constants, and $\bar{s}$ and $\bar{p}$ are the average of setup times and viewing times of the remaining observations to be scheduled. The remaining time available to schedule observation j at time t, $slack_{jt}$, depends on the relative positions of st_{jt}, D_j and visibility windows as follows:

(a) If $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$ and $\exists v' \in V$ such that $D_j \in TW_{v'j}$ then $slack_{jt} = (e_{v^*j} - st_{jt}) + \sum_{v^*+1}^{v'-1} (e_{vj} - b_{vj}) + (D_j - b_{v'j})$

(b) If $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$ and $D_j \notin TW_{vj}\ \forall v \in V$ in this case $v' = \max\{v | e_{v'j} < D_j\}$ then $slack_{jt} = (e_{v^*j} - st_{jt}) + \sum_{v^*+1}^{v'} (e_{vj} - b_{vj})$

(c) If $st_{jt} \notin TW_{vj}\ \forall v \in V$ in this case $v^* = \min\{v | b_{vj} > st_{jt}\}$ and $\exists v' \in V$ such that $D_j \in TW_{v'j}$ then $slack_{jt} = \sum_{v^*}^{v'-1} (e_{vj} - b_{vj}) + (D_j - b_{v'j})$

(d) If $st_{jt} \notin TW_{vj}\ \forall v \in V$ in this case $v^* = \min\{v | b_{vj} > st_{jt}\}$ and $D_j \notin TW_{vj}\ \forall v \in V$ in this case $v' = \max\{v | e_{v'j} < D_j\}$ then $slack_{jt} = \sum_{v^*}^{v'} (e_{vj} - b_{vj})$

The proposed rule is a generalization of the apparent tardiness cost (ATC) rule discussed in Morton and Pentico (1993) for the $1 || \sum w_j T_j$ problem to take into account SMS constraints, such as visibility windows and state dependent set up times. The proposed rule works as the WSPT rule when the observations are away from their deadlines and state dependent set up times between the candidate observations are not too diverse. However, because of the exponential term as the t gets closer to the deadlines of the observations the third term becomes more important and higher index values are assigned to the observations with the closer deadlines. Similarly, if the difference between the set up times is large then the middle term becomes more urgent and higher index values are assigned to the observations that can be scheduled earlier because of their low set up time. An outline of the proposed rule is as follows:

1. Calculate $st_{jt} = \max\{SC_{l_t j}, SL_{l_t j}\} + ct_{l_t}\ \forall j \in U_t$
2. For every observation $j \in U_t$, calculate $FAST_{jt}, slack_{jt}$, and $\pi_j(t)$
3. Select the observation j^* that has the highest $\pi_j(t)$ value and set $ct_{j^*t} = FAST_{j^*t} + P_{j^*}$. If $ct_{j^*t} \leq D_{j^*}$ then schedule j^* and let $l_t = j^*, cl_{l_t} = t = ct_{j^*t}, U_t = U_{t-1} - \{j^*\}$ and $S_t = S_{t-1} + \{j^*\}$. Goto step 1 until $ct_{l_t} \geq T$. Else, $U_t = U_t - \{j^*\}$ and goto step 2.

3.3. *Filtered beam search*

A filtered beam search resembles the famous branch and bound (B&B) algorithm, however it differs from it by pruning the nodes that are not seemed to be the most promising ones, which can be done only after a guarantee of non optimality is provided in a B&B algorithm. A filtered beam search is a fast, approximate B&B method which uses heuristics to estimate a fixed number of best paths, permanently pruning the rest. Lowerre (1976) was the first one to use beam search for a speech recognition called HARPY. Ow and Morton (1988) present a thorough analysis of a filtered beam search methodology for different scheduling problems. Sabuncuoglu and Karabuk (1998) employed it as an off-line scheduling algorithm for a flexible manufacturing system. A conventional filtered beam search has two decision parameters, namely beamwidth (b) and filterwidth (f). Each one of b beams is a temporary partial schedule. At each step of the algorithm, for each b beams, f promising unscheduled observations are determined with respect to a local evaluation function. Then global evaluation function scores are obtained for $b \cdot f$ partial schedules that are obtained by temporary addition of these f promising unscheduled observations to the corresponding b partial schedules. The best b of the $b \cdot f$ partial schedules are selected with respect to the global evaluation function scores until no more observations can be scheduled to any of the partial schedules.

In this paper, we add another parameter, called ''childwidth'' (c), to the classical filtered beam search and test its impact on the overall solution. If we denote the initial observation scheduled at each beam as a parent then the parameter (c) determines the maximum number of the children allowed for each parent, hence we limit the number of beams that originate from the same parent. The motivation behind this modification is to prohibit the premature entrapment of local optima that is quite possible after several iterations. An outline of the filtered beam search algorithm is given below in which the parameters ct_{jl_b}, ct_{l_b}, $FAST_{jb}$, l_b, S_b, $slack_{jb}$ st_{jl_b}, and U_b are found for each partial schedule of beam b.

```
Algorithm: While not done do
  1. Procedure   Filter_with_one_step
      1.1 Calculate $st_{jl_b} = \max\{SC_{jl_b}, SL_{jl_b}\} + ct_{l_b}$
      1.2 For every observation $j \in U_b$, calculate $FAST_{jb}$ and $slack_{jb}$
      1.3 Find the $L_{jb} = \frac{w_j}{p_j} * \exp\left(\frac{-(FAST_{jb} - ct_{l_b})}{k_1}\right) * \exp(\frac{-slack_{jb}}{k_2})$ $\forall j \in U_b$
      1.4 Select $f$ observations $j_1^*, \ldots, j_f^* \in U_b$ that has the highest $L_{jb}$ values. If $ct_{j^*l_b} = FAST_{jb} + P_{j^*} \leq D_{j^*}$ then set $i = i + 1$, $U_b = U_b - \{j^*\}$, $S_b = S_b + \{j^*\}$, and $Filterset_b = Filtersel_b + \{j^*\}$. Otherwise, delete the observations $j^*$ that have $ct_{j^*l_b} > D_{j^*}$ from $U_b = U_b - \{j^*\}$ and add to $CNS_b = CNS_b + \{j^*\}$
  2. Procedure   Evaluate_the_global_evaluation_scores
      2.1 For each $j^* \in Fillerset_b$ repeat the following
      2.2 Augment $j^*$ to partial schedule of beam $b$
      2.3 Schedule the remaining unscheduled observations to the augmented partial schedule with respect to
          $GF_{jt} = \frac{w_j}{p_j} * \exp\left(\frac{-(FAST_{jt} - cl_{l_t})}{k_1}\right)$
      2.4 Set $GES_{j^*b}$ to the corresponding objective function value of the schedule obtained by this explosion
  3. Select the best $b$ of $b \cdot f$ partial schedules with respect to the $GES_{jb}$ values by considering the limitation of $c$ for the beams that are originating from the same parent
  4. For each $b$ repeat the following
      If $|W| = |S_b| + |U_b| + |CNS_b|$ then $done_b = \text{True}$
      else $done_b = \text{False}$
  5. $done = \Pi_b done_b$
```

In the procedure Filter_with_one_step, we evaluate the local ranking indexes of the unscheduled observations for each b partial schedules with respect to L_{jb} values that we have proposed in the previous section. After the f promising candidates are determined for each of the b partial schedules, $b \cdot f$ augmented partial schedules are exploded by scheduling the remaining unscheduled observations dynamically with respect to the global ranking index, GF_{jt}, to obtain the global evaluation function scores, GES_{jb}. The GES_{jb} is the total weight of the observations that are scheduled before their deadlines for each exploded partial schedule. We then select the best b of the augmented partial schedules using the GES_{jb} values. Note that while selecting the best partial schedules, the number of children that belong to a specific parent is limited with the parameter c.

3.4. *Simulated annealing*

Simulated annealing (SA) is a well known widely used iterative improvement technique for optimization problems, initially developed by Kirkpatrick *et al.* (1983). It takes an initial solution, searches the neighbors of this solution, and moves to the neighbor if it has better objective value. It is also allowed to move to the neighbors with worse objective values with a probability of p usually set to $e^{\frac{-\Delta E_{ij}}{T}}$, where ΔE_{ij} is the loss in the objective function at a transition from a configuration i to its neighbor j and T is a control parameter corresponds to temperature. Both ΔE_{ij} and T are positive numbers. The probability of accepting a transition is called as the acceptance function. In this procedure, the probability of making uphill moves is initially higher. This is provided by selecting a high value of initial temperature, denoted as T_0, to avoid from premature entrapment of local optima. As the iterations proceed, T is lowered by a mechanism, which is known as cooling, and the final state is called the frozen level. Connolly (1990) shows that a sequential construction of neighborhood search is more effective than the random search method. Moreover an intelligent neighborhood generation mechanism can be used to overcome the most important shortcomings of SA, namely the huge computational time. Zegordi *et al.* (1995) propose such an algorithm for flow-shop scheduling problem by generating a list of promising neighbors that is called the moving desirability of jobs index and find the next neighbor according to that index rather than random or sequential. By this way it is possible to construct a smaller neighborhood space from the most promising ones. There are hundreds of papers available in the literature both in theoretical aspects and applications of SA. A review of the SA literature can be found in Collins *et al.* (1988) and Johnson *et al.* (1989).

In the proposed SA algorithm, we find an initial schedule by using the new dispatching rule discussed in Section 3.2 and apply a neighborhood search mechanism. We have previously mentioned that there are six different viewing instruments of HST and each instrument has several operating modes. Each instrument mode can be viewed as a ''family'', and there are 15 different observation families. The consecutive observations that belong to same family constitute the observation groups in the current schedule. We generate the neighbors of the current schedule simply by exchanging groups of observations that belong to different families. We calculate the exchange desirability values and generate the neighbors with respect to the most promising ones. Let us explain this procedure on an example problem with 5 observations. The observations 1, 2 and 5 require the WF/N camera and called Family A, whereas observations 3 and 4 require the FOC/96 camera and called Family B. The setup times from the initial state of the telescope, called state ''0'', to the required cameras are equal to $S_{0A} = 300$ and $S_{0B} = 400$ time units, whereas the family set up times are $S_{AB} = S_{BA} = 600$ time units. Suppose that the current schedule on hand is {5-1-3-2-4}, which has the following family structure: A(2)-B(1)-A(1) -B(1). The numbers in parenthesis represent the number of consecutive observations that belong to the given family, such as ''5'' and ''1'' are from family A whereas 3 is from family B. There are 6 different ways of possible group exchanges for the given example. All of the possible exchanges, their outcomes and exchange desirability values are given in Table 2. Note that the exchanged groups for each exchange number are represented in bold letters.

In Table 2, the New Family Structure column represents the new order of families after each exchange. For example, we interchanged the groups of observations A(2) and B(1) for the exchange number 1. The order of the observations in the parenthesis, i.e. (5-1-2), in the New Schedule column does not necessarily reflect the exact order. The exact order within the groups are determined by rescheduling each group using the new dispatching rule explained in Section 3.2, that considers the visibility windows and slewing times. However it might be computationally ineffective to reschedule all of the possible exchanges. Therefore, we determine the most promising exchanges by using the heuristic desirability values given in the last column. These exchange desirability values indicate the net gain from the family setups that will be obtained by the corresponding exchange. For example, if we exchange B(1) and A(1) for the exchange number 4 then we can save three family setups and incur an additional family setup resulting in a net gain of 1200 time units. Furthermore, we only generate the neighbors that have the highest exchange desirability values determined by the parameter desirable_exchanges_list_width (DELW). We use the first wins strategy while passing to the next neighbor. The main motivation behind this neighborhood generating mechanism is to augment the groups of observations that belong to same family to save from setup times. By this way we try to overcome the myopic nature of the dispatching rule that we have used while creating the initial schedule.

Furthermore, we can also mutate the current schedule obtained after each exchange with a certain mutation probability. The basic idea of mutation is to delete one of the scheduled observations in the current schedule. We consider two criteria while selecting the observation to be deleted. The first obvious one is to select the observations that are scheduled after their deadlines, which is called type I mutation, since the neighborhood generation mechanism ignores the observation deadlines. The second one is to select the observation that consumes highest amount of time due to reconfiguration times, slewing times and visibility windows availability, which is referred as the idle time of the

Table 2. Possible exchanges and their outcomes in the example problem

Exc. #	*Exchange*	*New family structure*	*New schedule*	*Desirability*
1	**B**(1)-**A**(2)-A(1)-B(1)	B(1)-A(3)-B(1)	(3)-(5-1-2)-(4)	500
2	**A**(1)-B(1)-**A**(2)-B(1)	A(1)-B(1)-A(2)-B(1)	(2)-(3)-(5-1)-(4)	0
3	**B**(1)-B(1)-A(1)-**A**(2)	B(2)-A(3)	(3-4)-(2-5-1)	1100
4	A(2)-**A**(1)-**B**(1)-B(1)	A(3)-B(2)	(5-1-2)-(3-4)	1200
5	A(2)-**B**(1)-A(1)-**B**(1)	A(2)-B(1)-A(1)-B(1)	(5-1)-(4)-(2)-(3)	0
6	A(2)-B(1)-**B**(1)-**A**(1)	A(2)-B(1)-A(2)	(5-1)-(3-4)-(2)	600

telescope in Fig. 1. Each observation has a backward idle time that is the time needed to execute the current observation after the previous observation, and a forward idle time that is the time needed to execute the next observation after the current observation. The total idle time of each observation is the sum of backward and forward idle times. After determining the total idle times, we create the deletion_observation_list (DOL) that comprises of the observations that need type I mutation and the observations with the highest total idle times. The number of the observations in DOL is determined by the parameter deletion_observation_list_width (DOLW), and each observation in this list deleted one at a time. We reschedule the remaining observations and calculate the new objective function value. Consequently, we either generate the next neighbor or select the next mutation candidate with a given probability acceptance function. We again use the first wins strategy while mutating the current schedule so that we can both obtain a diversity of search in the decision tree and calculate the opportunity cost of deleting one observation from the current schedule.

Algorithm:

```
1. Create the initial schedule ($IS_0$) by scheduling all the observations with the new dispatching rule described in Section 3.2
2. Set $t = 1$, $frozen :=$ false, current schedule $CS_t = IS_0$ and calculate the objective function value of $CS_t(Score_{CS_t})$ then set $BS = CS_t$, $Score_{BS} = Score_{CS_t}$ and $T_t = T_0$
3. While not ($frozen$) do
   3.1 Establish desirable_exchange_list ($DEL$) with respect to heuristic desirability values and select one pair randomly
   3.2 Set the candidate neighbor ($CN$) to the sequence obtained after the exchange, reschedule $CN$, and calculate $Score_{CN}$
   3.3 Evaluate $\Delta E = Score_{CS_t} - Score_{CN}$
   3.4 If $\Delta E \leq 0$ then set $CS_t = CN$ and $Score_{CS_t} = Score_{CN}$
   3.5 Else if $\Delta E > 0$ then set $CS_t = CN$ and $Score_{CS_t} = Score_{CN}$ with probability $e^{\frac{-\Delta E}{T_t}}$. If not accepted goto 3.2
   3.6 If $Score_{CS_t} > Score_{BS}$ then $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$
   3.7 Call Procedure Mutate for $CS_t$ with a given probability $mp$
   3.8 Set $t = t + 1$ and $T_t = T_{t-1} * \alpha$
   3.9 If $T_t <$ (critical temperature) then $frozen :=$ true
```

In step 3.7, we mutate the current schedule with a probability mp as follows:

Procedure Mutate:

```
1. While $k < DOLW$ do
   1.1 Establish deletion_observation_list ($DOL$)
   1.2 Select one observation from $DOL$ randomly and delete it
   1.3 Set the $CN$ to the new sequence obtained after 1.2, reschedule $CN$ and calculate $Score_{CN}$. Let $\Delta E = Score_{CS_t} - Score_{CN}$
   1.4 If $\Delta E \leq 0$ then set $CS_t = CN$, $Score_{CS_t} = Score_{CN}$ and $k = DOLW$
   1.5 Else if $\Delta E > 0$ then set $CS_t = CN$, $Score_{CS_t} = Score_{CN}$ and $k = DOLW$ with probability to $c^{-\Delta E/T_t}$. If not accepted then $k = k + 1$
2. If $Score_{CS_t} > Score_{BS}$ then $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$
```

3.5. *GRASP*

Greedy randomized adaptive search procedure (GRASP) is an iterative process that provides a solution to the problem at the end of each iteration and the final solution is the best one that is obtained during the search. There are various applications of GRASP in the areas of production planning and scheduling, graph theory and location problems as discussed in Feo and Resende (1995). Feo *et al.* (1996) apply the GRASP methodology to a single machine scheduling problem with sequence dependent set up costs and linear delay penalties. A typical GRASP consists of mainly two phases. The first phase is the construction phase. In this phase GRASP builds a feasible schedule by selecting and adding one element from a restricted candidate list (RCL) randomly to the partial schedule at a time with respect to a greedy function. $RCL = \{j : \beta_j \geq \beta\}$ where β_j is the ratio of the greedy function score of observation j to the highest greedy function score obtained at that iteration and β is a predefined ratio parameter. The second phase is the local optimization phase, in which GRASP explores the neighborhoods of the schedule obtained from the construction phase and tries to move to a better neighbor. In this phase we propose an algorithm that is similar to the neighborhood generation mechanism of SA algorithm, although we do not use mutation and only move to the neighbor if it gives a better solution than the current schedule. Furthermore, we modify the generic GRASP by applying the second phase if the constructed schedule seems to be a promising one. If the ratio of the objective function value of the current schedule to the upper bound value is greater than the ''allowable_percentage'' parameter then we apply the second phase, otherwise we return to the first phase and construct a new schedule.

```
Algorithm While the number of iterations $\leq MT$ do
  1. Procedure Construct_the_greedy_randomized_schedule
      1.1 Calculate $st_{jt} = \max\{SC_{l_{tj}}, SL_{l_{tj}}\} + ct_{l_t}$ and $FAST_{jt}$ $\forall_j \in U_t$
      1.2 Find the $GF_{jt} = \frac{w_j}{p_j} * \exp\left(\frac{-(FAST_{jt} - ct_{l_t})}{k_1}\right)$ $\forall j \in U_t$
      1.3 Let $GF_{ht} = \max\{GF_{jt}\}$ $\forall_j \in U_t$ and set $RCL_t = \{j : \frac{GF_{jt}}{GF_{ht}} \geq \beta\}$. Select an observation $j^*$ randomly from $RCL_t$, calculate $cl_{j^*t} = FAST_{j^*t} + P_{j^*}$ and schedule $j^*$. Let $l_t = j^*, ct_{l_t} = ct_{j^*t}, S_t = S_{t-1} + \{j^*\}$, and $U_t = U_t - \{j^*\}$
  2. Set $CS_t$ to the constructed schedule and calculate $Score_{CS_t}$
  3. If $Score_{CS_t} > Score_{BS}$ then set $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$
  4. If $Score_{CS_t} < Allowable\_percentage * Score_{BS}$ then goto step 1
  5. Procedure Local_optimization_phase
      5.1 While iteration number < Maximum _num-ber_of_exchange do
          5.1.1 Establish desirable_exchange_list (DEL) as discussed in the SA algorithm and select one pair randomly
          5.1.2 Determine the candidate neighbor (CN), reschedule CN, and calculate $Score_{CN}$
          5.1.3 If $Score_{CN} > Score_{CS_t}$ then set $CS_t = CN$ and $Score_{CS_t} = Score_{CN}$
      5.2 If $Score_{CS_t} > Score_{BS}$ then set $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$
```

4. Computational results

The algorithms presented in the previous section were coded in Pascal language and compiled with Sun Pascal Compiler on a Sparc Station 10 under SunOS 5.4. In this section we perform an experimental design to compare the proposed algorithms along with the NN rule with respect to the objective function values and the corresponding computation times. The objective function value of each algorithm is equal to the ratio of the total weight of the observations that are not scheduled before their deadlines to the total weight of all observations. There are five experimental factors, which are listed in Table 3, that can affect the efficiencies of the proposed algorithms, hence our experiment is 2^5 full-factorial design corresponding to 32 combinations. The number of replications for each combination is taken as 10, giving 320 different randomly generated runs.

We have used the actual data on the visibility windows, right ascension and declination data of 76 observations at Level 0, and randomly generated additional 39 observations at Level 1. Time horizon is a function of number of observations, oversubscription rate and reconfiguration times. We determine the time horizon for each experiment by scheduling all of the observations with respect to the NN rule and

Table 3. Experimental Factors

Factors	*Definitions*	*Low (0)*	*High (1)*
A	Number of Observations	76	115
B	Oversubscription Rate	20%	40%
C	Reconfiguration Times	High	Low
D	Due Date Percentages	0	5
E	Weight Assignments	1-2-3	1-5-9

divide the makespan value by 1.2 and 1.4 for 20% and 40% oversubscription rates, respectively. Due to the technological restrictions, HST needs long reconfiguration times, although the new space mission projects may need shorter reconfiguration times between the instruments. For the high reconfiguration times case, major and minor reconfiguration times are selected randomly from the interval UN $\sim$ [5000, 12000] and UN $\sim$ [1500,4000] seconds, respectively, where UN stands for the uniform distribution. On the other hand, for the low case, major and minor reconfiguration times are selected randomly from the interval UN $\sim$ [1600,4000] and UN $\sim$ [800,2000] seconds, respectively. The fourth factor determines the percentage of user specific deadlines that are before the prespecified time horizon. In Level 0, there is no user imposed deadlines, whereas for Level 1, 5% of the observations have deadlines that are selected randomly from the interval UN $\sim$ [0.25 * time horizon, time horizon] seconds. As discussed earlier, STScI divides the observations into ''high'', ''medium'' or ''supplemental'' groups. There are various ways of assigning values to each priority group. We set them in two different ways of (1-2-3) and (1-5-9), where the first values in both levels are assigned to the ''supplemental'', the second values to ''medium priority'' and the third values to the ''high priority'' observations. The viewing times are treated as fixed parameters and generated randomly from the interval UN $\sim$ [100,600] seconds.

There are several parameters of the proposed local search algorithms that should be set to specific values. We specify several different values for some of these parameters to determine their impact on the performance of the corresponding algorithm. For the new dispatching heuristic (NDH), we set $c = 0.03$ and $k_2 = 0.0002$. The values of these parameters are determined after numerous pilot runs. For the filtered beam search method, there are three parameters of beamwidth, filterwidth and childwidth. We set 2 different values to each of them generating 8 different filtered beam search algorithms as summarized in Table 4. For the GRASP algorithm, we tested the impact of β (the rate that is considered while creating the RCL) and *MT* by selecting two different values to each. Furthermore, we also tested the effect of second phase, i.e. local optimization, as shown in Table 5. In the first four of the algorithms we did not use the second phase, whereas we allowed the second phase with the allowable_percentage of 97% for the last four. The parameters of maximum_number_of_exchanged and *DELW* that are used in the second phase are set to 7 and 6, respectively. For the simulated annealing algorithms, we examine the effect of the mutation rate as shown in Table 6 such that we did not allow any mutation for the algorithm Sno whereas the

Table 4. Parameter settings of different beam search algorithms

Algorithm	*B1*	*B2*	*B3*	*B4*	*B5*	*B6*	*B7*	*B8*
Beamwidth	4	4	4	4	6	6	6	6
Filterwidth	8	8	10	10	8	8	10	10
Childwidth	3	4	3	4	3	6	3	6

Table 5. Parameter settings of different GRASP algorithms

Algorithm	*G1*	*G2*	*G3*	*G4*	*G5*	*G6*	*G7*	*G8*
Allowable%	No	No	No	No	97%	97%	97%	97%
β	0.2	0.2	0.6	0.6	0.2	0.2	0.6	0.6
MT	250	500	250	500	250	500	250	500

Table 6. Parameter settings of different simulated annealing algorithms

Algorithm	*Sno*	*S20*	*S50*	*S100*
Mutation rate	No	20	50	100

mutation is always applied for the algorithm S100. After several pilot runs, it is determined that $\alpha = 0.998$, $T_0 = 5$, $DELW = 10$ and $DOLW = 3$ give good results and reasonable computational times. Consequently, we compare 22 different algorithms, namely NN, NDH, 8 filtered beam search, 8 GRASP, and 4 SA algorithms.

In Table 7, we give the minimum, average and maximum values of the unscheduled weight percentages and the computational times over the 320 different randomly generated runs for each algorithm. The unscheduled weight percentage for each experimental run is equal to $(TW - OFV) / (TW)$ where TW corresponds to the total weight of the observations in the experimental run and OFV is the objective function value of the corresponding algorithm.

As indicated in Table 7, the NN rule gives the worst unscheduled weight percentage value of 0.245 and the minimum computation time of 16 milliseconds on the average of the 320 experiments as expected. The new dispatching heuristic (NDH) gives a better objective function value than the NN rule on the average by 1.3% (0.245–0.232). We also performed a paired t-test and the corresponding t value to the pair NN-NDH is -5.88 with a significance level of $p \leq 0.0001$, although its computation time is slightly greater than the NN rule. The averages of the unscheduled weight percentages indicate that filtered beam search algorithms perform better than the other competing algorithms, while the B7 algorithm, which has the parameters of $b = 6$, $f = 10$ and $c = 3$, is the best one among the beam search algorithms. The B7 algorithm improves the NN rule significantly by 7.2% (0.245–0.173) on the overall average. The highest improvement of B7 algorithm over the NN rule is achieved by 9.2% (0.288–0196) for the experimental combination of (1-1-1-1-1). This experimental combination corresponds to the 115 observations, high oversubscription rate, low reconfiguration times, the case where some of the observations have deadlines, and the weight assignment of (1-5-9). This is quite meaningful since the beam search algorithms, so as the B7 algorithm, consider the weights and the deadlines that are assigned to the observations while scheduling. The B7 algorithm also reduces the myopic nature of the dispatching heuristics with the help of the global evaluation function mechanism. Furthermore, we used a childwidth parameter to restrict the number of beams that originates from the same parent. Note that B1, B3, B5 and B7 are the beam search algorithms that have a smaller childwidth parameters, whereas B2, B4, B6 and B8 have the same beamwidth and filterwidth parameters with the preceding algorithms with higher childwidth parameters. From Table 7, we can conclude that childwidth parameter improves the overall average by 0.33%. The corresponding t values of the pairs B1-B2, B3-B4, B5-B6, and B7-B8 are 6.17, 5.95, 5.89, and 6.69, respectively, with $p \leq 0.0001$, and the computation times do not differ too much.

GRASP algorithms also improve the NN rule on the overall average of the unscheduled weight percentages, although they are not as good as the beam search algorithms. The best GRASP algorithm with respect to the objective function value is G8 which is worse than the B7 algorithm. The corresponding t value of the pair G8-B7 is -4.29 with $p \leq 0.0001$. However 16 out of 32 different experimental combinations when there is a high reconfiguration time, the G8 algorithm gives better objective function values than B7. In Table 8, we summarize the unscheduled weight percentages and the computation times of the algorithms for the high and low reconfiguration time cases. The unscheduled weight percentages are the averages of 160 experiments that correspond to each state. For the high reconfiguration time case, we find that G8 has the best overall average of 0.191, which is better than the B7 algorithm by 0.5% over the 160 experiments. The corresponding t value of the pair G8-B7 is 5.92 with $p \leq 0.0001$. This is mainly due to the local optimization phase of the GRASP algorithm that utilizes the ''family scheduling concept'', which becomes even more important when there is a high setup time between the families. We can also see that the local optimization phase improves the overall results of the GRASP algorithms, although it requires more computation time, and the t values for the pairwise comparison of objective function values of G1–G5, G2–G6, G3–G7, and G4–G8 are -14.43, -14.07, -7.14 and -7.46, respectively with $p \leq 0.0001$. Simulated annealing algorithms also do not perform as good as the beam search algorithms for the overall experiments, although they perform relatively better for the high reconfiguration time cases due to the neighborhood generation mechanism as indicated in Table 8. S100 has the best overall average of 0.194 among the simulated annealing algorithms, and it improves the NN rule by 5.1%. The highest improvement of S100 over the NN rule is achieved at the experimental

Table 7. Unscheduled weight percentages and computational times

Algo	*Unscheduled weight per.*			*Computation time (millisec.)*		
	Min	*Ave*	*Max*	*Min*	*Ave*	*Max*
NN	0.122	0.245	0.312	8	16	25
NDH	0.105	0.232	0.296	30	51	76
B1	0.050	0.174	0.244	8065	19919	33125
B2	0.056	0.177	0.250	8098	19889	32911
B3	0.050	0.174	0.242	10050	24624	41351
B4	0.051	0.177	0.245	10063	24610	41418
B5	0.053	0.174	0.244	12366	29978	49339
B6	0.052	0.177	0.250	11921	29806	50009
B7	0.050	0.173	0.243	14771	36789	61353
B8	0.057	0.177	0.244	14850	36941	62111
G1	0.129	0.225	0.280	5423	8827	12250
G2	0.123	0.219	0.276	10751	17623	24501
G3	0.074	0.185	0.240	5406	8821	12271
G4	0.073	0.180	0.239	10693	17716	24770
G5	0.121	0.214	0.279	6712	12575	20323
G6	0.111	0.208	0.271	12366	22465	34960
G7	0.072	0.183	0.238	9271	19070	30237
G8	0.071	0.178	0.237	16305	32637	49891
Sno	0.103	0.208	0.262	28690	100376	285239
S20	0.092	0.197	0.252	67303	165110	326189
S50	0.088	0.195	0.254	74116	206346	390555
S100	0.089	0.194	0.252	118283	333074	699810

Table 8. High and low reconfiguration time cases

Algo	*High Setup*		*Low Setup*	
	Unsch. weight per.	*Comp. times*	*Unsch. weight per.*	*Comp. times*
NN	0.266	16	0.224	16
NDH	0.254	51	0.212	52
B1	0.202	19521	0.145	20316
B2	0.204	19529	0.149	20249
B3	0.204	24144	0.144	25104
B4	0.206	24170	0.148	25050
B5	0.202	29411	0.145	30545
B6	0.205	29266	0.149	30346
B7	0.202	36083	0.144	37495
B8	0.205	36255	0.148	37628
G1	0.229	8821	0.221	8832
G2	0.224	17620	0.215	17624
G3	0.199	8831	0.171	8810
G4	0.194	17704	0.167	17704
G5	0.217	11583	0.211	13566
G6	0.212	21250	0.205	23679
G7	0.196	17015	0.169	21123
G8	0.191	30387	0.165	34886
Sno	0.220	87222	0.196	113030
S20	0.204	137242	0.189	192979
S50	0.203	175019	0.187	237674
S100	0.203	281535	0.186	384613

combination of (0-0-0-0-1) by 8.6% (0.230–0.144). Furthermore, the mutation concept improves the efficiency of the algorithms in the expense of computation times. The objective function value decreases from 0.208 to 0.194 as the mutation percentage increases, since SA can search more nodes of the decision tree.

From the above discussion, we can conclude that in general filtered beam search algorithms give the best objective function values on the overall average of 320 experiments. The main reason of this fact is the guided search methodology that is used in beam search algorithms. By the help of the local and global evaluation functions, the search on the decision tree is guided so that lower level searches focus in areas most likely to contain good solutions, however there is a danger of local entrapment at this methodology especially for the high reconfiguration times case. On the other hand, GRASP algorithms can break the local entrapment by the help of the random search methodology, hence they perform better than the beam search algorithms for the high reconfiguration times case on the average of corresponding 160 experiments. To sum up, we will present the time versus scientific return graphs of these algorithms in Figs. 2 and 3. The scientific return corresponds to average of scheduled weight percentages such that Scientific Return = 1 − (average of unscheduled weight percentages). Note that in Fig. 2, the scientific return is presented as the average of scheduled weight percentages of 320 experiments, whereas it is the average of 160 experiments that correspond to high reconfiguration time case in Fig. 3. The computation times are presented in milliseconds and time axis is in logarithmic scale. The lines in the figures correspond to the Pareto curves, hence only for the algorithms on

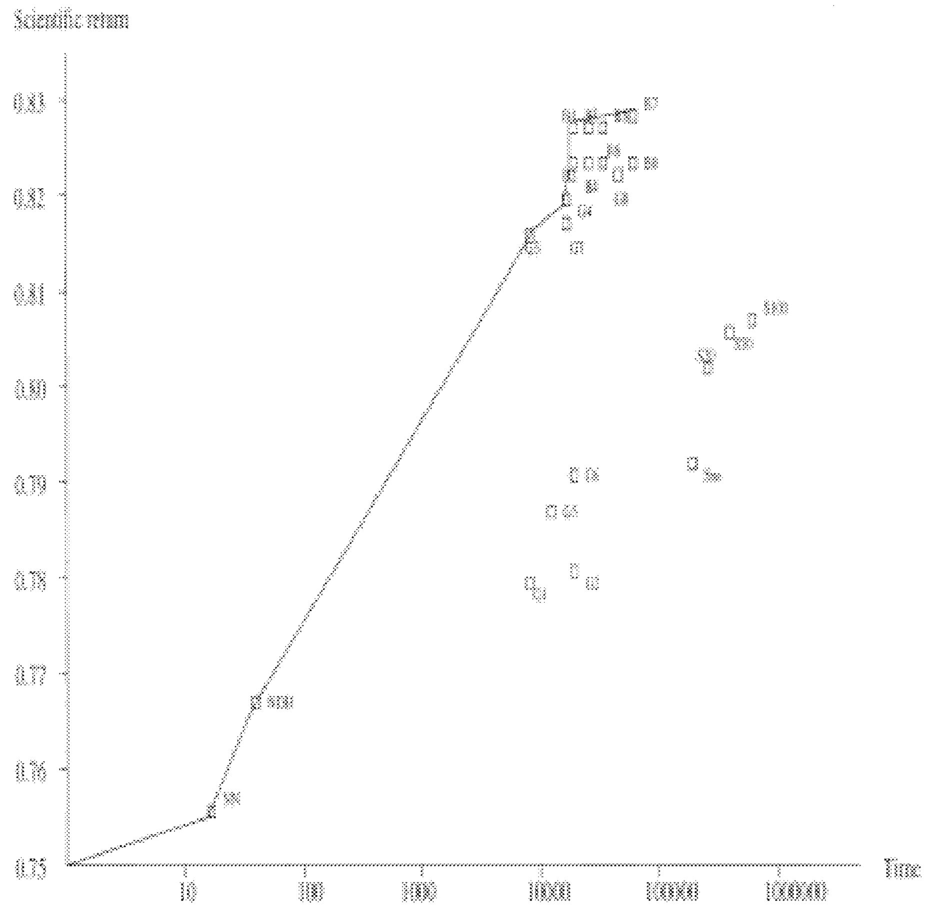

Fig. 2. Time versus the scientific return for each algorithm.

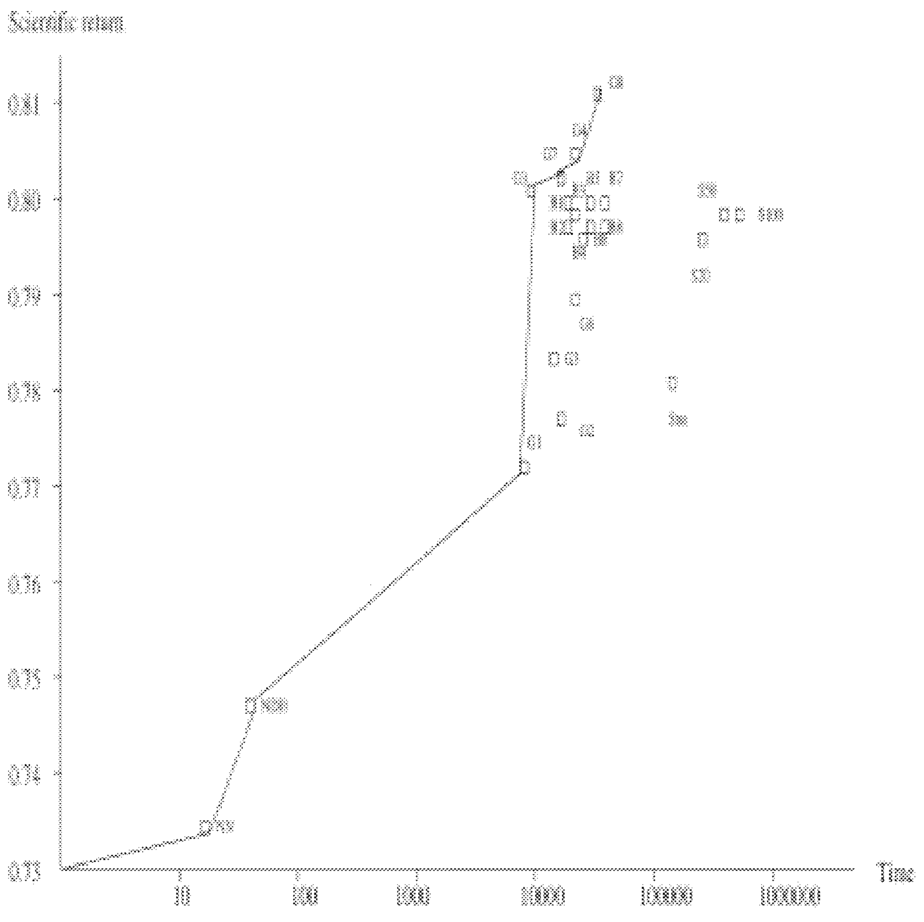

Fig. 3. Comparison of the algorithms for the high reconfiguration times.

the Pareto curves, there is no other algorithm that performs better both in scientific return and computation time.

5. Conclusions

In this paper, we have developed a new dispatching rule, which provides a higher scientific return than the NN rule used by Smith and Pathak (1991), and a set of more sophisticated local search algorithms that can identify the complex interactions between the candidate observations to construct the short-term observation schedules for space mission projects. While the specific constraints we have considered are those most relevant to Hubble Space Telescope, the proposed framework is more general, and could easily handle other over-subscribed scheduling problems. We can divide the proposed algorithms into two groups. The first group consists of the algorithms that require low computational times although they do not give good objective function values, and the second group consists of the algorithms that give better objective function values but require higher computational times. From Fig. 2 we can conclude that the simple dispatching rules NN and NDH belong to the first group whereas the local search algorithms belong to the second group. So if the time is limited for scheduling it is more preferable to use NDH since it gives better objective function values than the NN algorithm and requires a very small computational time compared to the local search algorithms. However, if the time permits we can use the B7 algorithm which gives the highest objective function value for the overall experiments. On the other hand, it is more preferable to use the G8 algorithm if the problem domain has a high reconfiguration time between the equipments.

Our proposed algorithms offer several advantages such as inclusion of the priorities that are assigned to the candidate observations by Space Telescope Science Institute, user specified deadlines, and the modifications to the generic local search algorithms to have a more realistic representation of the problem. In filtered beam search algorithms, we have utilized a childwidth parameter that restricts the number of beams that is generated from a particular beam. Our experimental results indicate that this restriction improves the objective function values of the beam search algorithms with almost no additional computational time requirement. Furthermore, we have introduced a mutation concept for the simulated annealing algorithms that improve the overall solution by 1.3%, although it requires more computational time. Finally, we have created the desirable exchange lists and eliminated less promising schedules in the GRASP and simulated annealing algorithms to reduce the search space.

There are several future research directions emanating from this study. Other local search algorithms such as tabu search and genetic algorithms can be applied to the problem domain, and the performance of these algorithms can be analyzed. Finally, the new dispatching rule is used as the local evaluation function for the beam search algorithms, and it is also used to generate initial schedules for the GRASP and simulated annealing algorithms. Different dispatching rules can be used for these algorithms and the performance of these new dispatching rules can be analyzed.

Acknowledgment

The authors would like to thank Dr. Stephen F. Smith from the Carnegie Mellon University for introducing the problem to us and providing the actual data of the observation requests.

References

Collins, N., Eglese, R. and Golden, B. (1988) Simulated annealing—an annotated bibliography. *American Journal of Mathematics and Management Science*, **8**, 209–308.

Connoly, D. T. (1990) An improved annealing scheme for QAP. *European Journal of Operational Research*, **46**, 93–100.

Desrochers, M., Lenstra, J. and Savelsbergh, M. (1990) A classification scheme for vehicle routing and scheduling problems. *European Journal of Operational Research*, **46**, 322–332.

Desrosiers, J., Dumas, Y., Solomon, M. M. and Soumis, F. (1994) *Time constrained routing and scheduling*, Technical Report G-92-42, GERAD, McGill University.

Feo, T. A. and Resende, G. (1995) Greedy randomized adaptive search procedure. *Journal of Global Optimization*, **6**, 109–133.

Feo, T. A., Sarathy, K. and McGahan, J. (1996) A GRASP for single machine scheduling with sequence dependent setup costs and linear delay penalties. *Computers & Operations Research*, **23**, 881–895.

Fisher, M. and Jaikumar, R. (1978) An algorithm for the space shuttle problem. *Operations Research*, **26**, 166–182.

Hall, N. G. and Magazine, M. J. (1994) Maximizing the value of a space mission. *European Journal of Operational Research* **78**, 224–241.

Hochbaum, D. S. and Landy, D. (1994) Scheduling with batching: Minimizing the weighted number of tardy jobs. *Operations Research Letters*, **16**, 79–86.

Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C. (1989) Optimization by simulated annealing: An experimental evaluation; Part 1, graph partitioning. *Operations Research*, **37**, 865–892.

Johnston, M. D. (1987) *HST Planning Constraints*, SPIKE Technical Report 87-1, STScI, Johns Hopkins University, Baltimore, MD.

Johnston, M. D. and Adorf, H. (1992) Scheduling with neural networks—the case of the Hubble Space Telescope. *Computers & Operations Research*, **19**, 209–240.

Kirkpatrick, S., Gelatt, C. and Vecchi, M. (1983) Optimization by simulated annealing. *Science*, **220**, 671–680.

Lann, A. and Mosheiov, G. (1996) Single machine scheduling to minimize number of early and tardy jobs. *Computers & Operations Research*, **23**, 679–681.

Laporte, G. (1992) The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, **59**, 345–358.

Lowerre, B. (1976) *The HARPY speech recognition system*, Ph.D. Thesis, Carnegie Mellon University, PA.

Minton, S., Johnston, M. D., Philiphs, A. B. and Laird, P. (1992) Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, **58**, 161–205.

Moore, J. (1968) An *n* job one machine sequencing algorithm for minimizing the number of late jobs. *Management Science*, **15**, 102–109.

Morton, T. E. and Pentico, D. W. (1993) *Heuristic Scheduling Systems: With Applications to Production Systems and Project Management*, John Wiley & Sons.

Muscettola, N., Smith, S. F., Cesta, A. and D'Aloisi, D. (1992) Coordinating space telescope operations in an integrated planning and scheduling architecture. *IEEE Control Systems*, **12**, 28–37.

Ow, P. S. and Morton, T. E. (1988) Filtered beam search in scheduling. *International Journal of Production Research*, **26**, 35–62.

Potts, C. and Wassenhove, L. V. (1988) Algorithm for scheduling a single machine to minimize the weighted number of late jobs. *Management Science*, **34**, 843–858.

Sabuncuoglu, I. and Karabuk, S. (1998) A beam search-based algorithm and evaluation of scheduling approaches for flexible manufacturing systems. *IIE Transactions*, **30**, 179–191.

Smith, S. F. and Pathak, D. K. (1991) Balancing antagonistic time and resource utilization constraints in over-subscribed scheduling problems. *Technical Report CMU-RI-TR-91-05*, The Robotics Institute, Carnegie Mellon University, PA.

STScI (1986) *Proposal Instructions for the Hubble Space Telescope*, Technical Report, STScI, Johns Hopkins University, Baltimore, MD.

Waldrop, M. (1989) Will the Hubble Space Telescope Compute? *Science*, **243**, 1437–1439.

Zegordi, S. H., Itah, K. and Enkawa, T. (1995) Minimizing makespan for flow-shop scheduling by combining simulated annealing with sequence knowledge. *European Journal of Operational Research*, **85**, 515–531.