GENERATING SHORT-TERM OBSERVATION SCHEDULES FOR SPACE MISSION PROJECTS

A Thesis
Submitted te the Department of Industrial Engineering
and the Institute of Engineering and Sciences
of Bilkent University
In Partial Fulfillment of the Requirements
For the Degree of
Master of Science

By
Kemal Kılıç
August, 1997

GENERATING SHORT-TERM OBSERVATION SCHEDULES FOR SPACE MISSION PROJECTS

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

Kemal Kılıç.

By [illegible]

Kemal Kılıç

August, 1997

ABSTRACT

GENERATING SHORT-TERM OBSERVATION SCHEDULES FOR SPACE MISSION PROJECTS

Kemal Kılıç
M.S. in Industrial Engineering
Supervisor: Asst. Prof. Selim Aktürk
August, 1997

Space mission scheduling (SMS) has been an important research area for several years. The basic features of the space mission projects are the high investment and operational costs, and limited resource availability. Therefore, it is very important to justify the high investment on the space mission projects by generating good schedules. In this thesis, we have proposed several new solution algorithms for generating short term observation schedules of space mission projects and test their efficiencies on a good representative of SMS problem; Hubble Space Telescope (HST) scheduling problem. HST is an exceptional space observatory at low earth orbit among the others that are used for space exposures. The main features of generating short-term observations of HST are state dependent set up times, user specified due dates, priorities and the visibility windows assigned to the candidate observations. The objective of HST scheduling is to maximize the scientific return.

We have proposed four new algorithms. The first one is a new dispatch rule that considers the basic features of the problem domain while scheduling the observations. The second one is a filtered beam search algorithm. We have introduced a new concept of childwidth, which is a parameter that restricts the number of beams that generates from the same parent. The third one is a Greedy Randomized Adaptive Search Procedure (GRASP) that needs to be tailored to be applicable to the problem domain. Finally, we proposed a simulated annealing algorithm with a new introduced concept of mutation. We have tested the relative performances of the proposed algorithms, as well as the nearest neighbor algorithm, both in objective function value and computational time aspects by utilizing a 2^k full-factorial experimental design.

Key words: Space mission scheduling, Hubble Space Telescope, local search algorithms.

ÖZET

UZAY PROJELERİNİN KISA DÖNEMLİ GÖZLEM ÇİZELGELEMESİ

Kemal Kılıç
Endüstri Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Yrd. Doç. Selim Aktürk
Ağustos, 1997

Son bir kaç yıldır, uzay projeleri çizelgelemesi (UPÇ) önemli bir araştırma konusu olmuştur. Uzay projelerinin en temel özellikleri yüksek yatırım ve işletim maliyetleri ile kısıtlı sayıda kaynak bulunmasıdır. Uzay projelerinin yüksek maliyetinin çok iyi çizelgeleme yapılarak karşılanabilmesi bu yüzden çok önemlidir. Bu tezde uzay projelerinin kısa dönemli gözlem çizelgelemesini sağlamaya yönelik yeni methodlar öneriyor ve bu methodların verimliliklerini, tipik bir UPÇ problemi olan Hubble Uzay Teleskopu (HUT) çizelgelemesi probleminde test ediyoruz. HUT dünya yörüngesindeki yeri itibarı ile, gözlem için kullanılan diğer gözlemevlerinin arasında önemli bir konuma sahiptir. HUT'un kısa dönemli gözlem çizelgelemesi probleminin en temel özellikleri teleskopun hazırlama zamanının duruma bağlı olması, görünebilirlik aralıkları ve gözlemlerin değişik ağırlıkları ile termin zamanlarının bulunmasıdır. HUT çizelgelemesinin amacı bilimsel kazancın arttırılmasını sağlamaktır. Bu amaçla dört yeni method önerdik. Bunlardan ilki, yeni tanımlanan ve gözlem çizelgelemesi yaparken problemin temel özelliklerini dikkate alan, bir öncelik sıralama kuralıdır. İkincisi ise, süzülmüş ışın taraması yöntemini temel alan bir algoritmadır. Süzülmüş ışın taraması yöntemini "çocuk sayısı" adını verdiğimiz ve aynı anadan çıkan ışınların sayısını kısıtlamayı sağlatan bir parametre ile geliştirdik. Üçüncü önerdiğimiz method ise problem ortamına yönelik geliştirilmiş bir GRASP uygulamasıdır. En son önerdiğimiz method ise, mutasyon ile geliştirilmiş bir yumuşatma benzetim methodudur. Önerilen dört algoritmanın ve de en yakın komşuyu seçmeye dönük olarak geliştirilmiş olan en yakın komşu algoritmasının, göreceli performanslarını hem bilimsel kazanım açısından hem de çözüm süresi açısından 2^k tüm-etkenli deneysel tasarımı ile test ettik.

Anahtar sözcükler: Uzay projeleri çizelgelemesi, Hubble Uzay Teleskopu, yerel tarama algoritmaları.

To my parents and to my love Alkım

ACKNOWLEDGEMENT

I would like to express my gratitude to Asst. Prof.Selim Aktürk due to his supervision, suggestions, and understanding throughout the development of this thesis and his patience.

I am also indebted to Prof. Nesim Erkip and Assoc. Prof. Erdal Erel for showing keen interest to the subject matter and accepting to read and review this thesis.

I cannot fully express my gratitude, love and thanks to Alkım Doğan for her morale support, encouragement and patience.

I would also like to thank to my officemates Muzaffer Tanyer and Özgür Tüfekçi and to Alev Kaya, Murat Bayız and Serkan Özkan for their help during the preparation of this thesis.

And my special thanks goes to Özgür Demirel for his great help and morale support throughout my studies.

And, finally I would like to thank to all my graduate friends for their friendship.

Contents

List of Figures

List of Tables

Chapter 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. There are several major sources of complexities in SMS problems. One of them comes from the fact of limited and inflexible resources that are available in the problem domain, and the safety requirements of these resources. Limited electric power and thermal balance requirements that constraint the parallel usage of the instruments on board, the transmission data bands width, the limited capacity of the data storage instruments can be stated as some of the constraints that should be taken into consideration on most of the Space mission scheduling problems. One major outcome of such constraints is the lack of relying on specific representational assumptions as it is done in most of the manufacturing scheduling problems. For example it is not enough to allocate time to the main activities, researchers in SMS must consider enough detail to ensure safe and feasible execution of the each component of the problem domain.

Another important characteristic and source of complexity in SMS problems is the extremely high number of requests that comes from the astronomers from all over the world. This is mainly because, space mission projects are very expensive to build and operate. Many of the scientific research that require the conditions that are unique in the space such as weightlessness and extreme vacuum should be scheduled in limited number of resources. This is why SMS problem is highly over-subscribed and why it is very important to use these scarce sources efficiently.

One of the good representative of the SMS problem is generation of short-term observation schedules for Hubble Space Telescope of NASA. Hubble Space Telescope (HST) is an exceptional observatory among the others that are used for space exposures. This exceptional place of HST comes from its location at low earth orbit which enables it to overcome many limiting conditions that are available in the earth surface and atmosphere and provides unsurpassed combinations of sensitivity, wavelength coverage and angular resolution. The six viewing instruments on HST allows space exposures and analysis of the celestial objects 7 or 10 times further than the ones that can be done with other observatories on earth surface.

With these features and opportunities the scientific community obviously eager to take the advantageous of such an unique observatory. So the request for observations from HST far exceeds the capability of the telescope. The expensive investment of $ 1.5 billion must be justified by high scientific outcome so that the scheduling of the candidate observations should be done efficiently.

A request for an observation is mainly an exposure of a particular celestial object with a particular configuration of a particular viewing instrument. In order to start an exposure, several constraints must be satisfied. Firstly, the specified instrument and the configuration must be active on HST. To activate the specified instrument some time is needed which depends on the previous instrument used for the previous exposure. Furthermore, the telescope must be pointing to the specified celestial object. This process is called as the slewing process. The needed time for slewing the telescope depends on the previous position of the telescope. Finally the specified celestial object must be visible. Since HST is at low earth orbit the celestial objects are occulted periodically by earth. So to start an exposure of an celestial object is only possible if it is in its visibility window. Space Telescope Science Institute assigns different priorities to the observation requests with respect to their relative importances. So the aim of generating short term schedules of HST is to maximize the scientific return by satisfying the constraints of the problem domain.

In this thesis we have proposed several new solution procedures for generation of short term schedules of SM problems and test their efficiency by applying them to a typical SM problem; HST scheduling problem. We also compared the performances of the proposed algorithms with the nearest neighbor heuristic. The well-known nearest neighbor heuristic selects the first available observation, schedules it as early as possible and repeats the same procedure in the following steps. The proposed algorithms consist of a new dispatch heuristic, a filtered beam search algorithm, a Greedy Randomized Adaptive Search Procedure (GRASP) and a simulated annealing algorithm. The new dispatch heuristic considers the priorities, needed setup times, processing times and the remaining times that are available to schedule a particular observation. A new concept, namely the "child width", is introduced to the classical filtered beam search algorithm. By the help of the child width we restricted the number of beams that are generated from a particular parent. We tested the effect of the filterwidth, beam width as well as the newly introduced concept child width on a set of randomly generated problems. The third algorithm proposed is GRASP. This algorithm consists of two phases. The first phase is constructing an initial greedy randomized solution and the second phase is the local optimization search. The effect of the local optimization search phase and different parameter settings of GRASP are tested on a set of randomly generated problems. Finally, a simulated annealing algorithm is proposed. The classical simulated annealing algorithm is modified with a mutation concept and the effect of this modification is tested at different mutation ratios on a set of randomly generated problems. For the local search algorithms, we have selected the filtered beam search and GRASP algorithms, since they are recently developed algorithms and do not have many applications in the literature. On the other hand, simulated annealing is selected as a representative of the well-known local search techniques and has a wide area of applications that are available in the literature.

The remainder of the thesis can be outlined as follows. In chapter 2, we will give a review of the literature on SMS, HST scheduling, and related topics such as single machine scheduling, vehicle routing planning and different local search algorithms. In chapter 3, we will define the problem, state the assumptions and propose a mathematical formulation of the problem. In chapter 4, we will present our proposed algorithms. In chapter 5, we will give the computational results of the experimental design, present a comparison of the algorithms and discuss the effects of the different parameter settings of each algorithm. Finally, in chapter 6 we will provide some concluding remarks and the future research issues.

Chapter 2

LITERATURE REVIEW

In this chapter after reviewing of the space mission problems, the basic components and structure of HST domain will be presented. A detailed review of two main research directions to the solution of HST scheduling problem namely, SPIKE and HSTS, will be done. A review on single machine scheduling problem and vehicle routing problem, which are closely related to the problem will be presented. And finally a brief discussion on some heuristic search methods that are used in the proposed algorithms will be provided.

2.1 SPACE MISSION SCHEDULING

We have described the main characteristics of the Space mission scheduling (SMS) problems in chapter 1. Now we present a review on the applications that are found in the literature. Mainly there are two different approaches existing in the literature. 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 (CSP). In this approach due to the complex nature of the problem domain software architectures which use constrained directed scheduling techniques are constructed.

Fisher and Jaikumar [14] provide an algorithm for another example of SMS, namely the scheduling of the NASA space shuttle program. It requires a selection of mission launch times that minimize the number of late missions, where each mission has an earliest start time and a latest start time. A one machine approximation is provided for the problem. The proposed algorithm is inspired by Moore's [43] algorithm for minimizing the number of tardy jobs on a single machine.

Hall and Magazine [19] model the problem as a single machine scheduling and use traditional OR methodology. The problem is simplified and one resource is considered (time; whereas it is also claimed that other resources such as data band width and electric power can be easily incorporated). In the model each activity has a weight and a single time window (a specific time interval that the execution of the activity is allowed). The objective is maximizing the total value of the projects. Eight heuristics and two upper bounding procedures are described. Finally a dynamic programming algorithm that incorporates the heuristics and the bounding procedures is proposed.

However the above approximations assume single time windows associated with the activities which is not a general feature of SMS problem domains, hence it is not realistic for many cases. Mainly time windows are related to the occultation of the targets with the other astronomical bodies such as earth, sun, etc. In many space mission problem domain (such as low orbit earth satellites and HST) there are multiple time windows that are associated to each target periodically. Even though the planning horizon consists of multiple time windows, one can claim that each period can be considered one at a time, however this would lead to solutions that are far away from the optimum. A second misleading point that is not considered in the above formulations is the sequence and state-dependent setup times required for the execution of each activity. More detailed discussion on the reasons of the sequence and state-dependent setup times will be done in the next section.

Gabrel [16] presents a parallel machine scheduling approximation of the low-orbit Earth satellites scheduling that will be discussed in more detail in the single machine section of this chapter.

As mentioned earlier, another approach to the SMS was building software architectures mainly depending on constraint-directed search scheduling methodology. One such tool is the Ground Processing Scheduling System (GPSS), a scheduler tool proposed by Deale et al. [7] that uses GERRY [64] scheduling engine developed at NASA. GPSS is designed for the space shuttle ground processing scheduling system. After the shuttle returns to Earth and until the time it leaves the launching pad, there are many activities related to the maintenance, modifications, testing and repair that should be scheduled very efficiently. One of the main sources that increases the complexity of the problem is the amount of work that are not predetermined which requires an interactive scheduling. Also Aarup et al. [1] propose a knowledge-based planning system for spacecraft AIV which is a system developed for European Space Agency.

Some other tools that use a similar perspective are also developed for a typical SMS problem, namely the HST scheduling problem. These proposed approaches will be discussed in detail in the following sections. But before going on to this discussion let us first describe the HST domain components.

2.2 HST DOMAIN COMPONENTS

Hubble Space Telescope is an approximately $ 1.5 billion project that is carried out by NASA and placed at its orbit on April 1990. The overall management of the telescope is done by Space Telescope Science Institution (STScI) and it has an expected lifetime of 15 years. In the beginning, a manufacturing flaw in the primary mirror of the HST happened to cause some problems in the project. However this did not affect to obtain valuable scientific results from the exposures that are done by the observatory. In late 1993 space shuttle was sent to the observatory and the main camera was replaced with a second generation instrument and the mirrors figure was compensated which in turn increased the optical qualities of the HST.

The main components of the HST domain can be stated as follows; There are six viewing instruments on board: two cameras, two spectrographs, a photometer and a fine guidance sensors. There are some possible different configurations of each of these viewing instruments. The primary mirror of 2.4 m diameters gathers the light from the celestial objects called **targets** and focuses the light to the viewing instruments. The scientific data gathered from the targets are either communicated directly to earth or stored to three on-board tape recorders. Transmission of the data to earth is done through one of the two satellites of Tracking and Data Relay Satellite System (TDRSS). There are two links between the HST and earth with different communication rates, 4 kilobits per second (kps) and 1 megabits per second (mps). However, since HST is in low earth orbit of 590 km with an orbital period of 95 minutes, most of the targets as well as the TDRSS satellites are periodically occulted by earth, and it is not possible to use these links all the time. Another reason for this is that HST shares the TDRSS system with many spacecrafts. HST cannot use 4 kps link approximately 10% of the time when it cannot communicate with either of the two satellites and can use the 1 mps link on the average 20 minutes per orbit.

Since the targets are occulted periodically by earth, exposures on targets are possible only at some portion of each orbit. This time intervals at which the exposures are possible are called the **visibility windows.** Occultation by earth is not the only factor that affects the starting and ending times of the visibility windows; South Atlantic Anomaly (SAA), the occultation of the targets by the moon and the sun are some of the other main factors.

Proposals for the candidate observation is mainly a request of an exposure of a particular celestial object with a particular configuration of a particular viewing instrument. So it is not sufficient to start an execution of a candidate observation, even if it is in its visible time window. Obviously to start an execution of an exposure, HST must be pointed at the target, and the requested configuration of the instrument should be set as well as the target should be in its visible time window.

If the telescope is not pointed to the target then some amount of time is needed for the pointing process. This pointing process is called as **slewing process** and the needed time is called as **slewing duration.** Slewing duration depends on both the average slewing rate and the angular difference between the requested observation target and the current target that HST is pointing at. In fact there are some other factors that affect the slewing duration in the complex domain of HST, such as pointing restriction relative to sun.

Similarly, some amount of time is needed in order to set the requested configuration of the selected instrument if it is not the one that is currently set. This process is called as **reconfiguration process,** and the needed time as **reconfiguration duration.** The reconfiguration process of the instruments are one of the sources that increases the complexity because of the complex power up/ power down sequences needed in order to satisfy the limited electric power and thermal constraints of HST.

For generating short-term schedules for HST, researchers must consider the user-imposed constraints as well as the concepts of visibility windows, slewing and reconfiguration process, and limited electric power constraints. There are many sources of scheduling flexibility for an astronomer when preparing a proposal for request of HST observations. Proposals from the astronomers may be observation programs that include several observations. The astronomer may request specific precedence, ordering, minimum and maximum time separations, repetitions and interruptibility for their observation programs. It is also possible for an astronomer to specify some conditions on exposures and these *conditional* exposures are contingent and are not scheduled until the astronomer decides whether it should be scheduled or not. It is quite possible that a need for an exposure can emerge as a consequence of some other results obtained from different observations.

Astronomers prepare their proposals in two phases. In the first phase they just describe the scientific intent and required resources, and submit these proposals to STScI. These proposals are reviewed at STScI on an annual basis and the decision of programs for the coming year is done by an allocation committee. The number of approved programs are expected to exceed the capacity of the observatory in order to ensure the high utilization of the telescope. The astronomers must prepare a detailed proposal for the second phase if their programs are approved by STScI. In the second phase they give detailed information about the needed instrument parameters, astronomical objects, individual exposures and the other user-imposed constraints if any. They can submit their proposal through Remote Proposal Submission System (RPSS) electronically in an ASCII file. RPSS can also determine some of the errors that can be done while preparing a proposal. A detailed information about the preparation of the proposals can be found at [59]. These proposals are the main inputs for the planning and scheduling process of HST.

STScI also assigns priorities to the approved observation programs considering their scientific value and operational efficiency as follows:

i) "high-priority" observations that take nearly 20% of the estimated available time

ii) "medium-priority" observations that take nearly 70% of the estimated available time

iii) "supplemental-pool" observations that takes nearly 30-50% of the estimated available time

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. However some important shortcomings of SOGS emerged and it is understood that it was not working as desired. Main problems of the SOGS were, its inflexible nature which cannot accommodate some constraints due to the characterization of the problem, incompleteness due to not considering many physical constraints and leading to extra manual work, incorrect programming methodology used when developing the software and computational infeasibility that emerges from the non-hierarchical nature of the solution approach. A detailed discussion on shortcomings of SOGS can be found in Waldrop [62]. This important shortcomings of SOGS lead STScI to find out new solutions to the planning and scheduling problem of HST. One of the major outcome of this new research was SPIKE.

2.3 SPIKE

SPIKE is developed by Johnston and Adorf [25] in order to overcome the shortcomings of SOGS and augmented to the system. Mainly, the contribution of SPIKE was partitioning the problem into two parts; long-term schedule and short-term schedule. 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. This is quite logical because of two reasons. First of all, the magnitude of the problem makes it impossible to generate detailed schedules in feasible computation time. Secondly, orbital constraints loose certainty on longer horizons.

The underlying philosophy of SPIKE is constrained-directed search paradigm. This paradigm is an incremental problem solving methodology based on repeated global analysis of the characteristics of constraints implied by the current solution as a means for structuring and exploring the underlying search space. Briefly it can be stated as identifying the relation on the set of all variables that satisfies all of the constraints.

SPIKE views the scheduling as a constraint satisfaction problem that maximizes the total number and importance of the constraints that are satisfied. SPIKE divides the constraints into two sets; one of them is the *hard constraints* that are required to be satisfied in order to obtain executable schedules and the other set is *soft constraints* which are not required to be satisfied but should be considered because they represent a preference for the related activities. Different weights are assigned to the constraints. Obviously, hard constraints have higher weights than the soft ones. Since SPIKE currently handles the long-term scheduling of HST, temporal constraints associated with the activities have higher weights than the resource utilization constraints.

SPIKE takes the detailed proposals of the astronomers and creates suitability function for the activities from the constraints. The suitability function is a function of time and represents the desirability of an activity to start at the specified time. Then SPIKE generates a schedule by taking these suitability functions into consideration.

Initially the schedule search was done by a neural network developed by Johnston and Adorf [25], namely the Guarded Discrete Stochastic (GDS) network. However, Minton and Philips [41] and Minton et al. [40] showed that a Min-conflict algorithm distilled from GDS, is at least as effective as GDS and moreover very easy to implement and at least an order of magnitude faster than the GDS network. So the GDS is replaced with the multi-start stochastic repair routine which is an heuristic repair-based search method. This new routine consists of *trial assignment, repair* and *deconflict* steps.

In trial assignment step an initial solution is generated which should not have to be consistent, that is to say it is allowed to have some constraint violations. This initial solution will be the input to the repair step and it is known that good initial solutions give better results at the repair step both in computational aspect and objective aspect. Over thousands of combinations of heuristics SPIKE team identified several good heuristics. One of the most successful of these heuristics is as follows: Select the most-constraint activity to assign first. The number of Min-conflict times is used as the measure for degree of constraints. Then the activities are assigned to the times with minimum conflict, with ties broken by maximum preference derived from suitability functions or by the earliest time [26].

Then a repair heuristic is applied to this initial solution. There are several alternatives in selection of the heuristics. One of the best is Min-conflict heuristic that is mentioned previously. This heuristic selects an activity in conflict with another one, and assigns it to the place which minimizes the number of conflicts. As an improvement to this algorithm such as in the selection of the activity a Max-conflict activity selection can be used. Repair search continues until a predefined effort has been extended or no conflicts is left.

Finally SPIKE uses a deconflict routine that selects the activities based on lower priority, higher number of constraint conflicts and lower preference time assignments values, and removes them from the schedule. It also tries to schedule unscheduled observations if there are gaps in the schedule that are suitable for them.

Since the heuristics used in SPIKE are stochastic and it is possible to use different heuristics in both trial assignment and repair step, better results can be obtained by generating as many schedules as time permits.

The small observation buckets constructed by SPIKE can be easily handled by SPSS. This hierarchical approach to the problem of using SPIKE as long-term scheduling tool and SPSS as the short-term scheduling tool overcomes the computational infeasibility of SOGS. After SPSS completes the generation of short-term detailed schedule, SPIKE analyzes and determines the factors that can affect the future schedule. The observations that are not scheduled by SPSS, become the candidate observations for rescheduling in SPIKE.

Although SPIKE is developed for HST scheduling problem, many different applications of SPIKE to other problems are available. Some of these problems are astronomical scheduling problems, such as ASTRO-D mission scheduling [22] and XTE [44]. There are further discussions available in Johnston and Miller [26] of some other applications to astronomical scheduling problems of SPIKE. For the mentioned two applications, SPIKE is both used as the long-term and short-term scheduling tool. But several adaptations are needed in order to implement it to the short-term scheduling problem. These were task preemption, new classes of short-term scheduling constraints such as modelling target occultation, maneuver and setup times between the targets (note that since SPIKE was designed to be a long-term scheduler user-imposed constraints were the main constraints that were considered rather than the physical constraints) and a post-processor that examines short-term schedule for utilizing any remaining gaps [26].

There are even some different problem domains that SPIKE is applied. Johnston and Minton [27] apply the multi-start stochastic repair routine, which is the heart of SPIKE, to n-queen problems and job-shop scheduling problems, and obtain remarkably good results after comparing it to some other approaches found in the literature.

However SPIKE is currently the long-term scheduling tool of HST and to adapt it to short-term scheduling of HST is a complex task and beyond the topic of this thesis. Whereas we will provide some information about the short-term scheduling methodology of the SPSS later.

2.4 HSTS

Second major search direction in HST planning and scheduling is Heuristics Scheduling Testbed System (HSTS) [45]. HSTS is a software architecture that integrates planning and scheduling. It was partly supported by NASA and developed mainly at the Carnegie Mellon University in order to generate more effective solutions to HST scheduling problem. The software architecture provides a domain description language (DDL) for modelling the static and dynamic structure of the system and a temporal data base (TDB) for representing possible behaviors of the system. It also provides an opportunistic, constraint-directed methodology.

In a previous research by Ow and Smith [50], it was shown that this approach is more powerful than the classical approach in large-scale manufacturing scheduling problem. The main motivation behind the development of HSTS was adopting and implementing the concept of opportunistic, constraint directed scheduling and test its performance in a highly complex domain such as HST scheduling. Thus in the very beginning HSTS denoted the opening form of Hubble Space Telescope Scheduling [46]. But several years after the birth, the project team renamed it and chose the current one as its name. This change was not only in the name but also they improved the software and used it at some other problem domains such as job-shop scheduling and transportation planning [45]. More detailed information about the software architecture of HSTS is available at [45], [46], and [47]. Even though the adaptations and the architecture of HSTS is beyond our topic, the methodology that is provided for generating executable short-term schedules for HST is obviously very important and should be discussed in detail.

Similar to SPIKE approach, HSTS also decomposes the problem into long-term and short-term scheduling problems and focuses to the short-term scheduling. However there is a difference between the approach of SPIKE and HSTS to the long-term scheduling. HSTS look to the aggregate level as if it is a level that the observation programs are assigned to the smaller buckets considering capacity requirements. That is to say, there is no need, in fact no possibility, to ensure satisfaction of the feasibility constraints while at aggregate level. In fact it is even desired to have certain amount of oversubscription in order to guarantee high utilization. But obviously short-term schedule must satisfy such constraints in order to have executable schedules.

HSTS proposes to handle the problem of generating short-term schedule in different levels of abstractions. This is a quite logical approach for the domains that are complex and highly interactive such as HST scheduling. Furthermore the magnitude of the overall problem suggests to construct the detailed and executable schedules after some commitments are done. By this way the problem can be analyzed at an aggregate level that allows to focus to more important aspects that are mainly effective in overall objective, whereas the detailed levels can be used to refine the intermediate solutions in order to satisfy all of the constraints.

In this regard HSTS uses two different level of abstractions, namely *abstract level* and *detailed level.* 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 as discussed in Muscettola [45].

There is a strong communication between these two levels. Observations that are sequenced in the abstract level are communicated to detailed level. The abstract level implicitly accounts the overall telescope reconfiguration, slewing and warm-up and shut downs, as temporal delays rather than explicitly model them as it is in the detailed levels. Since the observation programs are sequenced with respect to the abstract estimates of reconfiguration times, it is quite possible some modifications would be required for that sequence. So the detailed level communicates back to the abstract level a more precise account of the reconfiguration delays.

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. So it is important to have a good sequencing methodology at the abstract level.

Smith and Pathak [57] proposes three strategies for the abstract level of HSTS, and to the best of our knowledge these strategies are the current ones in the core of HSTS. The first strategy tries to maximize the utilization of the HST. The observation that can first start after the last observation of the partial schedule is assigned as the next job to be selected. This procedure continues in this manner until there is no time left or no job left to be selected. This is a dispatch-based methodology namely the Nearest Neighbor (**NN**) that is a simple heuristic used to solve the Travelling Salesperson Problem (TSP) type problems. Also a modified version of NN with Look-Ahead (**NNLA**) is provided. NNLA works as NN while tries to minimize the rejections (Note that an observation is rejected if there is no time available for the execution of the observation in the partial schedule) of the observations by a look-ahead mechanism. But even with this look-ahead mechanism it is impossible to generate good solutions in the interacting nature of the constraint, and thus very myopic.

Second strategy focuses on the maximizing the number of scheduled observation programs and thus minimization of the rejection of the observations. It tries to add the job with the fewest allowable start time, where this process is named as the Most-constrained First (**MCF**). It is also a notable aspect of MCF, that the next assigned observation should not be added just after the last job of the partial schedule. A shortcoming of this algorithm emerges from the fact of the myopic nature of assigning the next most constrained in the cycles that have nearly the same allowable start times.

Moreover a third strategy is suggested which in fact is a multi-perspective scheduling methodology that tries to balance the both of the objectives of maximizing the utilization of HST and maximizing the number of scheduled observation programs. **MCF/NN** strategy achieves this by opportunistically selecting one of the first two strategies at each step. Analysis on the results provides important insight about the relative strength of alternative strategies. These insights give the chance of opportunistically selecting the appropriate strategy at each step by analyzing the current state of the partial solution. So there is no a priori strategy that the scheduler must obey. That is to say in this strategy if the difference of the maximum allowable start time and the minimum allowable start time (that are determined just by calculating the allowable start times for each unscheduled job) decreases below a certain threshold value than the scheduling continues with NN strategy and otherwise uses MCF.

These three heuristics are tested by Smith and Pathak [57] with an experimental design that only considers two different instruments. In their research, **NN** strategy is determined as the one that provides the highest utilization where as **MCF** provides the minimum rejection, so they sum up with the third one and suggest **MCF/NN** as the initial building blocks for the solution to the problem. However as it was mentioned earlier these methodologies are very simple and myopic. Infact, Smith and Pathak [57] also accepted that these are not sophisticated strategies and developing better methodologies are mentioned as future research directions. Also it is important that these strategies do not consider the different priorities of the observation programs, which is also stated in [57] as a future research direction. And this thesis demonstrates more powerful heuristics where the priorities are also considered.

2.5 VEHICLE ROUTING PLANNING

A typical vehicle routing planning (VRP) problem is finding the minimum costing routes for a fleet of vehicles that serves to a set of customers with fixed demand. Desrochers et al. [8] provide a comprehensive classification scheme for vehicle routing planning and scheduling problems. Laporte [36] represents an overview of the main exact and approximate algorithms to VRP. In the vehicle routing scheduling and planning with time windows (VRSPTW) problem there is an additional time constraint associated with each customer. That is to say each customer has a time window which starts with an allowable earliest start time and ends with an allowable latest start time. There are basically two types of time windows in VRSPTW. The first one is the "hard" time windows which specifies the time interval that is required to serve to the customer. However, in the second case time windows are "soft", so with a penalty cost it is allowed to violate the time window constraint.

VRSPTW emerged from the fact of time-constrained activities that should be considered while routing and scheduling. Some important application areas of this problem are the routing and scheduling of bank deliveries, meal delivery services, dial-a-ride services, and school bus routing and scheduling.

Solomon [58] presents four heuristics for the solution of the problem. First one is a saving heuristic which begins with n distinct routes in which each customer is served by a dedicated vehicle and tries to merge the routes that will bring profit to the objective function. Second one is a sequential tour-building heuristic that starts every route by finding the closest unrouted customers and continues in this manner. This is the famous nearest neighbor procedure. Third one is an insertion heuristic that initializes a route (with respect to several greedy heuristics such as earliest deadline or the farthest), for each unrouted customer computes the best feasible insertion place which looks for the one that minimizes the extra distance and the extra time required to visit the customer. And finally a sweep heuristic which repeatedly applies to phases. In phase one by the sweeping algorithm the customers are clustered and assigned to a vehicle. In the second phase a single-vehicle schedule is created for the customers that are assigned. Then this process is repeated with the unassigned customers remained from phase one and the customers that are assigned but could not be scheduled in phase two due to the time window constraints, until all customers have been scheduled.

Potvin and Rousseau [53] represent an algorithm that makes use of the generic insertion heuristic of Solomon but introduces another customer selection cost as well as a parallel route building philosophy. In this philosophy a set of routes are initialized at once and the remaining customers are inserted to any one of the routes. Kolen et al. [31] represent a B&B method for the same problem.

Atkinson [3] describes a general greedy heuristic which includes a look ahead capacity for large-scale vehicle scheduling problem with time windows. The problem is scheduling of vehicles to deliver school meals to dining centers from kitchens. In the formulation each dining center is supplied by a particular kitchen whereas several dining centers can share a particular kitchen. There is an earliest pick up time associated with the kitchens and a latest delivery time associated with the dining centers. Maximum time allowed that a food can be kept in the vehicle is another constraint that is considered in the formulation.

Balakrishnan [4] proposes three heuristics for the VRP problem with soft time windows. In the formulation the penalty cost of violating the time windows is a linear function of the amount of the time window violation. The heuristics proposed are basically nearest neighbor, savings and space-time heuristic. Jones [28] proposes a mean distance heuristic to VRSPTW problem under uncertain demand. A typical case of this problem is scheduling service engineers within time windows where schedules are prepared interactively. Gendreau et al. [17] proposes an integer programming formulation and a specialized enumerative algorithm that tries to minimize the number of customers visited after their deadlines. However the lack of the allowable start time differs the problem from the concept of time windows. This problem is denoted as the travelling salesman problem with deadline.

Desrosiers et al. [9] provide an extensive overview on time constrained routing and scheduling. Some problems such as TSPTW, constrained shortest path problem, VRPTW, pick up and delivery problems with time windows are formulated and some solution procedures based on Dantzig-Wolfe decomposition/column generation, Lagrangian relaxation and dynamic programming are proposed.

Vehicle routing scheduling and planning with time windows (VRSPTW) problems share many common features with the SMS problems. 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 target and multiple time windows associated with the customer can be viewed as the visibility windows of the celestial targets. Furthermore, the time required between two customers is both sequence and state-dependent. However as discussed above the VRSPTW literature considers the single time windows and mostly tries to find the minimum costing route as the objective, where as the objective of an HST approximation should be cared for minimizing the number of unvisited customers within a time horizon. Furthermore the customers considered in the literature usually have equal priorities.

2.6 SINGLE MACHINE SCHEDULING

The objective of HST scheduling is stated as to maximize the scientific return. Scientific return both depends on the number of the observation programs that are scheduled and the priorities that are associated with the corresponding observation programs. So this objective can be stated as minimizing weighted number of tardy jobs in terms of single machine scheduling. It is possible to visualize HST scheduling problem as single machine scheduling in order to minimize the weighted number of tardy jobs with sequence and state-dependent setup times. Here state-dependent term is related to visibility windows of the targets. It is state-dependent since even the needed time to start a particular observation after another particular observation is not constant since it depends to the ending time of the previous observation and the visibility window of the next observation. Where as sequence-dependent setup cost between two particular activities is constant.

Time windows concept mostly refers to due windows of the jobs in single machine scheduling literature. In fact due windows implicitly can be viewed as single time windows (visibility windows in HST). Anger et al. [2] formally introduce the term due window and provide a polynomial time algorithm for the minimization of the number of jobs completed outside their time window [33]. Lann and Mosheiov [35] study due-windows with the objective of minimizing the (weighted) number of early and tardy jobs. Some other researchs on single machine scheduling with due windows such as Kramer and Lee [33] and, Liman and Ramaswamy [38] focus to the earliness and tardiness penalties which become to be an important research area with the evolution of Just-In-Time (JIT) philosophy in manufacturing.

Moore [43] presents a polynomial time algorithm that solves the single machine sequencing problem of minimizing the number of the tardy jobs. However minimizing the weighted number of tardy jobs is NP-hard. Lawler and Moore [37] present a pseudopolynomial dynamic programing algorithm to minimization of weighted number of tardy jobs as well as maximization of weighted earliness, minimization of tardiness with respect to common relative and absolute deadlines, and minimization of weighted tardiness with common due date. Potts and Wassenhove [52] propose a branch-and-bound algorithm that reduces the size of the search tree with dominance reductions and reductions with respect to the lower bounds obtained from knapsack formulation of the problem. Villarreal and Bulfin [61] also provide a branch-and-bound algorithm for the weighted number of tardy jobs problems. Kise et al. [30] describe a solvable case of single machine scheduling with ready and due dates where the objective is minimizing the number of tardy jobs. They also propose a polynomial time for the case where $r_i < r_j \rightarrow d_i \leq d_j$.

Until now none of the above papers deal with the sequence dependent set-up time constraints. Hochbaum and Landy [20] show that the decision problem of weighted number of tardy jobs with batch setup is NP-complete and propose a pseudopolynomial algorithm. At their formulation the completion time of a job is same with the completion time of the corresponding batch. Monma [42] also considers the batch setup and provides a dynamic programming algorithm which is exponential in the number of batches. Nowicki and Zdrzalka [48] represent a tabu-search approach for general cost functions for single machine scheduling with minor and major setup times and give results of computational experiments for maximum weighted tardiness and total weighted tardiness. Feo et al. [13] apply Greedy Randomized Adaptive Search Procedure (GRASP) methodology to single machine scheduling with sequence dependent setup costs and linear delay penalties.

Obviously, single machine can only correspond to single time window at space mission scheduling. However it is possible to visualize multiple orbits (which in turn causes multiple time windows) as parallel machine at scheduling literature. In a planning horizon T, the low orbit Earth satellites (or HST) make k revolutions (orbit) around the Earth. If we refer to k revolution as k identical parallel machine then the problem with time windows will be a slightly more realistic representation of SMS. A very important shortcoming point of this approach to the SMS problems emerges from the fact that scheduling orbit by orbit would lead to very myopic solutions since it will not consider the overall performance of the planning horizon which typically consist of tens of orbits. Gibrel [16] discusses the problem of jobs within time windows on identical parallel machines and proposes an algorithm for the decision problem. In the formulation each job can be processed only on a subset of machines. Also a discussion of this formulation in a low-orbit satellite scheduling concept is provided. However in the formulation neither the slewing time nor the reconfiguration time is considered. Furthermore, there is no priority differences of the targets and only the decision problem is discussed.

2.7 HEURISTIC SEARCH METHODS

Most of the real-life scheduling problems, as well as the SMS problem, cannot be solved with exact algorithms in a reasonable computational time because of their complex nature. So heuristics are developed in order 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. Some of the most promising search methods such as GRASP, filtered beam search and simulated annealing will be used in our proposed algorithms. We will now present a brief discussion on these concepts.

2.7.1 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. A typical GRASP consists of mainly two phases. The first phase is the construction phase. In this phase GRASP builds a feasible solution by selecting and adding one element from the alternatives to the list at a time. At each stage of the construction phase all the elements in the candidate list is ordered with respect to a greedy function. Then a restricted candidate list (**RCL**) is constructed, where the best elements are selected while constructing the RCL. Then randomly any of the element is selected from of the RCL as the next alternative while constructing the solution. It is also adaptive because after adding the lastly selected alternative to the partial solution list, the state of the partial solution list is also updated in order to reflect the consequences of the currently added alternative. As we update the state of the partial solution, obviously, the greedy function values of the elements will change in the proceeding stages of the construction phase.

The second phase is the local optimization phase, in which GRASP explores the neighborhoods of the solution obtained from construction phase and tries to move to a better neighbor. If such a neighbor is determined then GRASP replaces the current one in hand with the neighbor and repeats the same procedure with the new solution. This procedure goes until no such neighbor is found.

The main motivation behind this two-stage structure is as follows. Even though the construction phase quickly generates a good solution it is not guaranteed to be a local optimum. It is quite possible to improve a solution by applying a different iterative improvement method that seeks the local optima [13]. Also it is known that the selection of a good initial solution leads the local optimization methods to better results. Especially the speed of such local optimization techniques significantly increases by starting with a good solution. Since the initial solution of the second phase is the one obtained from the first phase, second phase works very effectively. Randomized approach of GRASP gives the chance to the user to spend some time for improving the solution as the time permits.

There are basically two different parameters that should be selected. We must decide the number of the best candidates that enters to the RCL at each stage of the construction phase. One way is just selecting a constant number that specifies the number of the elements of RCL. Feo et al. [12] suggest a decision parameter (α) and determine the ratio of each candidate elements greedy function value over the best greedy function value of that current state and selects the candidates that has higher ratio than α . Another parameter to be decided is *stopping criterion.* There are various possible ways of terminating the search. One and the most common way is just fixing a constant number of iterations and then terminate.

It is also possible to modify the first phase of GRASP, by not selecting the element to be added to the solution from the RCL randomly. The selection of the next job can be done by assigning different probabilities with respect to their greedy function values and we can consider these probabilities while constructing the initial solution.

A typical GRASP procedure is as follows

```
1. Define the Initial State
2. While **not** *(stopping criteria met)* do the following
  2.1 **Procedure** Construct the greedy randomized solution
    2.1.1 Set the Partial Solution $(PS) = \{\}$
    2.1.2 While **not** (greedy solution constructed) do the following
      2.1.2.1 Construct RCL
      2.1.2.2 Select randomly an element from RCL $(s)$
      2.1.2.3 Add this element to partial solution (set $PS = PS \cup \{s\}$)
      2.1.2.4 Adapt Greedy function
    2.1.3 Set greedy solution $(GS) = PS$
  2.2 **Procedure** Local Optimization Search
    2.2.1 While **not** ($GS$ locally optimum) do the following
      2.2.1.1 Find a better solution $GS^*$ from neighbors ($GS^* \in N(GS)$)
      2.2.1.2 let new $GS = GS^*$
  2.3 Set the current solution to GS ($CS = GS$)
  2.4 If the current solution is better than the best solution ($BS$) found
      until now than update the best solution ($BS = CS$)
3. End. (The output is the best solution ($BS$))
```

There are various applications of GRASP available in the literature. For example Feo et al. [13] apply the GRASP methodology to single machine scheduling with sequence dependent set up costs and linear delay penalties. Laguna and Gonzalez-Velarde [34] use a hybrid heuristic that combines some features of tabu search with GRASP and uses it as a search heuristic for just-in-time scheduling in parallel machines. Ghosh [18] develops a GRASP heuristic for maximum diversity problem. Feo and Resende [12] represent a detailed discussion on the GRASP methodology and numerically show how it works for two graph theory problem, set covering and maximum independent set problems. They also represent a very good review of the GRASP applications at the areas such as production planning and scheduling, graph theory and location problems, and propose a way of augmenting the mutation concept to GRASP.

2.7.2 BEAM SEARCH

Beam Search is a heuristic search method developed to search the decision trees of the optimization problems in a fast way. Mainly it resembles the famous branch and bound (B&B) algorithm. However it differs from it by pruning the nodes that do not *seem* to be the most promising ones, which can be done only after a guarantee of non optimality in B&B algorithm. The most promising b nodes, where b is the *beamwidth*, is determined with respect to a heuristic evaluation function. Tree moves downward only from these most promising b nodes while pruning the other nodes. The heuristic evaluation function that is used when determining the most promising nodes, can either be a *one-step priority evaluation function* or a *total cost evaluation function.* One-step priority evaluation function has a local view that only considers the profit that would be obtained only after the next decision to be made and assigns this profit as the value of the associated node. Whereas total cost evaluation function has a global view by projecting the current partial solution to a complete solution and by considering the overall profit at the end that would be obtained and assigns this profit as the value of the associated node.

Obviously local view of the evaluation functions is very fast, however global view will probably lead to better decisions. In order to select the most promising b nodes it may not be feasible to use total cost evaluation function for large scale problems. In order to overcome this fact a filtering mechanism is generated. In this mechanism f nodes, where f is the *filterwidth*, are filtered based on a local evaluation function and the rest are pruned. Then these f nodes are evaluated with respect to the global evaluation function and b of them are selected in order to move down on the tree. This approach is known as filtered beam search as discussed in Ow and Morton [51].

In filtered beam search there are two decision parameters. These are the beamwidth (b) and filterwidth (f). There is a close relation between the magnitude of this two parameters and the computation time of the algorithm. If a high beamwidth and filterwidth are selected than the computational time will be high. This is because of extra calculations needed to search over b nodes and f computations of global evaluation function is needed for each b. However, if small values for these parameters are chosen then the trade-off will be less diversity in the search space, and thus entrapment in a local optima. So it is important to select good parameters for the routine which in fact closely dependent to the nature of the problem.

Beam search was firstly developed in 1976 and mostly used in artificial intelligence domain. Lowerre [39] firstly used beam search on a speech recognition called HARPY. Fox [15] used beam search in ISIS which was the first constraint based scheduling system. He used an incremental beamsearch through a space of partial schedules and reschedules by restarting the beam search. Fargher and Smith [11] use beam search for planning in a flexible semiconductor manufacturing environment. Later Ow and Smith [51] present a thorough analysis of a filtered beam search methodology in single machine early/tardy and weighted tardiness problem of flow shops. Recently, Sabuncuoglu and Karabuk [56] propose a filtered beam search algorithm for scheduling in FMS and conclude their superiority to other dispatch-based algorithms. Finally Sabuncuoglu and Bayiz [55] represent an application of beam search on job shop scheduling problems.

2.7.3 SIMULATED ANNEALING

Simulated Annealing (SA) is a well known widely used iterative improvement technique for optimization problems, initially developed by Kirkpatric et al. [29]. It has a close relation with the physical process of annealing, in which physical substance (such as metal) is melted, i.e. raised to higher energy levels, and gradually cooled until it becomes solid. The aim of this process is to obtain the minimal energy state of the substance at the end. It is natural for the substance, to pass from higher states to lower states. Moreover with a probability that depends on the current temperature it is also possible to pass to the higher levels from the lower energy levels. This probability is determined with the following expression: $p = e^{\frac{-\Delta E}{kT}}$ where ΔE is the difference between the energy levels, k is the Boltzmann constant and T is temperature.

Analogically SA works as follows. 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 $e^{\frac{-\triangle E_{ij}}{T}}$, where $\triangle E_{ij}$ 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, and both $\triangle E_{ij}$ and T are positive numbers. The probability of accepting a transition is called as the *acceptance function.* This procedure continues until no improvement can be done on the objective.

The main difference between SA and the *down-hill search* (or descent algorithms) is the fact that SA allows occasional uphill moves. In this procedure, the probability of making uphill moves is initially higher. This is provided by selecting a high value of initial T (T_0). This is just to avoid from premature entrapment of local optima. As the iterations proceed, by a mechanism T is lowered until it approaches to zero. This reduction of T is known as cooling and the final state as *frozen* level. Vidal [60] classifies two different strategies for the cooling mechanism of T. First one is *homogeneous* where T is decreased after a number of transitions (L), and the other is inhomogeneous where T is decreased after each transition.

Johnson et al. [23] classified the choices which should be made by the user of SA as *problem specific* and *generic.* It is stated that initial solution (I_0), neighborhood generation and evaluation of $\triangle E_{ij}$ are problem specific, whereas initial temperature (T_0), number of iterations (L_k), temperature function (T_k; the function that determines the temperature at the k^{th} iteration. Usually $T_k = \alpha T_{k-1}$ is used in literature) and stopping criteria are generic.

It is possible to add transition selection structure to generic choices. That is to say when do we decide to move. Diekmann et al. [10] represent four different move mechanism. *First win;* chooses the first accepted configuration for an update, *Best win;* chooses the configuration with the best objective function, *Boltzmann;* weights the configurations according to the boltzmann distribution, and *Random*; chooses a random configuration out of the set of accepted configurations.

It is also possible to generate different transition selections. For example Ishibuchi et al. [21] propose an algorithm of selecting randomly N neighbors and moving to the best of this N configurations. By this way they save from the computational time of generating all possible neighbor configurations and try to avoid from a poor solution that a first-win strategy may lead.

A typical Simulated Annealing algorithm is as follows (In the light of above discussion following algorithm is an homogeneous algorithm that uses a first-win strategy) ;

```
1. Construct an Initial Solution ($I_0$)
2. Choose an Initial Temperature ($T_0$) where $T_0 > 0$
3. Set $k = 1$ and $T_k = T_0$
4. Set Current Solution ($CS_k$) $= I_0$
5. While not *frozen* do
  5.1 Repeat the following loop $L$ times
```

5.1.1. Generate a random neighbor of CS_k , namely CS_k^*

5.1.2. Evaluate $\triangle E = \triangle E_{CS_k} - \triangle E_{CS_k^*}$

5.1.3. If $\triangle E \leq 0$ then set $CS_k = CS_k^*$ (downhill movement)

5.1.4. If $\triangle E > 0$ then set $CS_k = CS_k^*$ with a probability of $e^{\frac{-\triangle E}{T_k}}$(uphill movement)

5.2 Set $k = k + 1$

5.3 Set $T_k = \alpha T_{k=1}$

6. End.

The efficiency of the SA depends on several factors. One of the most crucial is the way of how the neighbors are generated. Most of the research define a neighborhood generation structure and choose the next potential solution at random from the set of the neighbors of the current solution, However, Connolly [6] presents 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. For example Zegordi et al. [63] propose such an algorithm for flow-shop scheduling problem. It is proposed to generate a list of promising neighbors that is called *Moving Desirability of Jobs Index* and generate the next neighbor according to that index rather than random or sequential. By this way it is possible to construct smaller neighborhood space from the most promising ones. It is also suggested by Zegordi et al. [63] to use a tabu list in order to avoid from infinite loops.

Another important factor is the quality of initial solution. Johnson et al. [23] experimentally show that starting with a good solution increases the power of SA.

A third factor is the tuning of the parameters of SA. For example, it is very important to select good initial temperature and a cooling scheme. If the temperature is too high than very bad uphill moves would be accepted, while if it is too low than the move will quickly be entrapped in a local optimum and the rest of the search will be only useless trials to escape from the local optimum. Connolly [6] proposes an annealing scheme that tries to maximize the portion of the search near an *optimum temperature* where he suggests a way of estimating the initial and final temperature. There are various cooling scheme available in the literature. Johnson et al. [23] suggest to use the standard geometric cooling method since there seems to be no reason to replace it with the other methods such as logarithmic cooling, linear cooling, etc.

There are hundreds of papers available in the literature on both theoretical aspects and applications of SA. The applications range from pollution control to travelling salesman problem, from layout problem in flexible manufacturing systems (FMS) to DNA mapping and many more. For example Kouvelis and Chiang [32] propose SA to layout problem in FMS, Osman and Potts [49] propose SA for permutation flow-shop scheduling. Collins et al. [5] represent a good survey paper that lists the applications in the literature. Finally Johnson et al. [23], [24] represent a good review of SA and test its performance in graph partitioning [23] and graph coloring problem [24].

2.8 SUMMARY

Until now we have described the nature of the problem briefly and discussed the proposed approaches to the SMS problems as well as the other related topics such as single/parallel machine scheduling and vehicle routing problems. As previously mentioned, the algorithms that are proposed to SMS problems are either far from reflecting the important features of the problem such as multiple time windows and state dependent set up times or very myopic dispatch rules such as NN and MCF. It is clear that there is a need of further research that takes the complex nature of the problem into consideration and at least explicitly explore the interactions between the candidate observations. In this research we will propose tailored heuristic search procedures and test their efficiency with an experimental design. We will also compare these procedures with the one of the dispatch based heuristics suggested by Smith and Pathak [57], namely with the NN. In the following chapter we will provide detailed information about the problem. Next we will define our proposed algorithms and the modifications that are done on the general algorithms in order to handle the specific problem of SMS.

Chapter 3

PROBLEM STATEMENT

In this chapter we will first present the objective of the problem. Next we will define the constraints of the problem domain. We will state the assumptions that are made through out this thesis and finally we will present the notation that we have used along with the mathematical formulation of the problem.

3.1 OBJECTIVE

The aim of this thesis is to propose new solution procedures that generate good and executable short-term schedules to the Space Mission Scheduling (SMS) problems and to test their efficiencies under different experimental conditions. Due to high complexity and interacting nature of the state dependent constraints of the problem environment, even to generate executable schedules is a hard task. However it is not sufficient to provide feasible schedules. In order to justify the great investment and high operation costs the scheduling algorithm must perform well.

In the literature mainly two different performance measures are proposed as the objective of the SMS problems. First one is related to resource utilization and tries to maximize the observation time. Since the resource investment is very high (mostly billions of dollars), the idle time and the set up time of the resource must be as few as possible. Second objective that is proposed is maximizing the scientific return of the mission which tries to maximize the scheduled programs as much as possible by taking into account the assigned priorities to the programs. Since SMS problems are mostly over-subscribed it is important to schedule as many programs as possible.

In HST scheduling problem STScI assigns different priorities (high, medium and supplemental) to the programs proposed by the astronomers. Even though utilization of HST is very important, this objective disregards the priorities associated with the observations. On the other hand, maximizing the scientific return considers the priorities. So it is more realistic to select it, rather than the former one. Maximizing the number of the scheduled observation is same as minimizing the number of unscheduled observations. So the latter objective turns out to be, the famous "minimizing the number of weighted tardy jobs" in scheduling terminology.

So the objective of this research is generating a schedule that minimizes the number of unscheduled observations in a given scheduling horizon by taking into account the priorities associated with the observations and satisfies all of the constraints.

3.2 CONSTRAINTS AND PREFERENCES

Both "hard constraints" that are required to be satisfied in order to obtain executable schedules and "soft constraints" which are not required to be strictly satisfied but preferred to be considered, are available in the problem domain of HST scheduling problem. From now on we will refer to the hard constrains with the term *constraints* and the soft constraints will be referred as *preferences.* There are mainly two type of constraints. The first type of the constraints emerge from the physics of observations of HST and the second type consists of the user imposed constraints that are specified by the astronomers when

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

Table 3.1: The viewing instruments of HST and their possible configurations

they send their proposals for observations.

There are six different viewing instruments on HST. These are 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. The instruments and their possible configurations are presented in table 3.1. For example WF/PC can be used in four different modes of WFN, WFUV, PCN and PCUV. The percentages given in table 3.1 present the possible percentages of the observations that use the specific instrument.

An exposure request from HST is mainly the collection of data from a celestial object by using a viewing instrument with a particular configuration. Since there is limited power on board, the instruments are not allowed to be operational simultaneously. Some time is needed in order to reconfigure the instrument that is needed for the next observation. This **reconfiguration** can be changes of instruments which we will refer as the *major reconfiguration* or can be mode changes of the instruments which will be referred as the *minor reconfiguration.* Since these changes are depending to the previous state of the instrument, execution of complex power up/power down sequences are required. Also reconfiguration must be synchronized appropriately among different instruments. Furthermore some stabilization time delay may be needed in order to achieve thermal stability. So the time needed for reconfiguration is the total amount of time needed for configuration time and the stabilization time delay. However this total time can only be specified at very detailed levels. For abstract level scheduling the worst case time estimates are used [57].

In order to start an execution HST must be pointing at the target. However, since HST is placed at low orbit of the earth, the targets are not visible for all of the time. Periodically they are occulted by the earth surface. Furthermore there are other constraints that limit the time available for the execution of an observation at a target. For example some instruments cannot be operated when HST is passing over the South Atlantic Ocean due to the radiation belt of the earth (South Atlantic Anomality). It is not generally permitted to observe targets that are close than 35^0 to sun. Exception may be made for observing inner planets with the sun blocked by the earth. Also it is not permitted to observe targets within 15^0 to moon. Some observations must be executed when HST is in earth shadow in order to minimize the stray light. These above factors limit the size of the **visibility windows** of the targets. We present an example of visibility windows in figure 3.2.

Even the next target is in its visibility window, 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 to the slewing angle between the previous direction and the next direction, and to the angular speed of slewing.

Astronomers locate the stars by a system similar to the equatorial system used by navigators. Both the celestial and terrestrial coordinate systems are based upon the earth's rotation, and therefore upon the position of its South

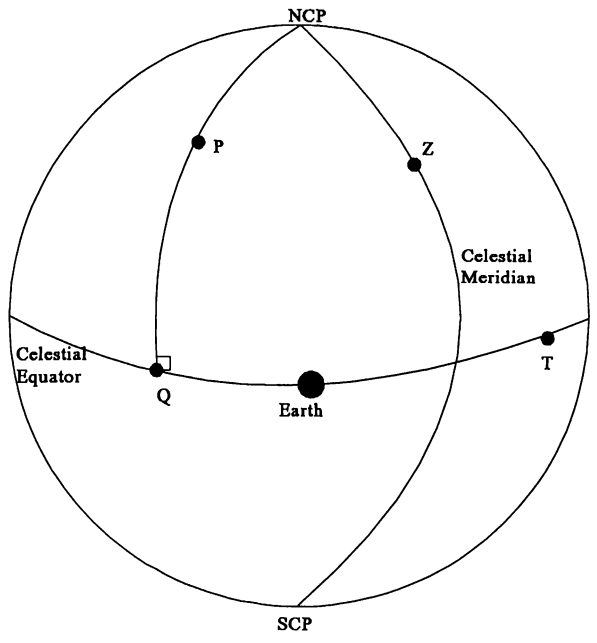

Figure 3.1: The equatorial system of celestial coordinates

and North poles and the position of the its equator. The celestial equator is the intersection of a plane through the Earth's center, whose axis is the axis of rotation of the Earth, with the celestial sphere. To fix a particular place on the celestial equator one of the equinoxes, the points of intersection of the celestial equator with the ecliptic is chosen. The vernal equinox (First Point of Aries) is the one that is used and denoted with the symbol **T** in figure 3.1. Two angular distances namely the **declination** (DEC) and **right ascension** (RA) is used in order to specify the position of the celestial object on the coordinate system. The DEC of the point P (see figure 3.1) is the angular distance measured positive (negative) toward the North (South) Celestial Pole from the celestial equator along the great circle passing through the point P and the North Celestial Pole (NCP). The declination of the point P in the figure 3.1 is the angular distance between P and Q ($\widehat{PQ}$). The RA of the point P is the angular distance measured toward the east, from the vernal equinox, along the celestial equator to the intersection of the great circle passing through the point P and the NCP with the celestial equator. The RA of the point P in figure 3.1 is the angular distance between T and Q ($\widehat{TQ}$).

In order to calculate the slewing time between two celestial objects the well known cosine formula is used. Suppose that HST was previously picturing point P (from-f) and next will take the pictures of the point Z (target-t) in the figure 3.1. Then the slewing time of the telescope is calculated as follows.

DEC_f : Declination of the 'from' celestial object in radians;

DEC_t : Declination of the 'target' celestial object in radians;

RA_f : Right ascension of the 'from' celestial object in radians;

RA_t : Right ascension of the 'target' celestial object in radians;

$rel - RA = |RA_f - RA_t|$

$b = \frac{\pi}{2} - |DEC_f|$

if $signum\ DEC_f = signum\ DEC_t$ then $c = \frac{\pi}{2} - |DEC_t|$

else $c = \frac{\pi}{2} + |DEC_t|$

Slewtime from 'f' to 't' $= \frac{\arccos[(\cos(b) \times \cos(c) + \sin(b) \times \sin(c) \times \cos(rel-RA))]}{angular\ slewing\ velocity}$ where the $angular\ slewing\ velocity = 0.0017453294$

The visibility windows of two celestial targets of A and B are presented in figure 3.2 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 figure 3.2, the exposure of the celestial target B can only start if it is in its visiblity window, if the required instrument is active and if the telescope is repointed. We will refer to time that is spent after maximum of the reconfiguration and slewing time until the next scheduled observation is visible again as the idle time.

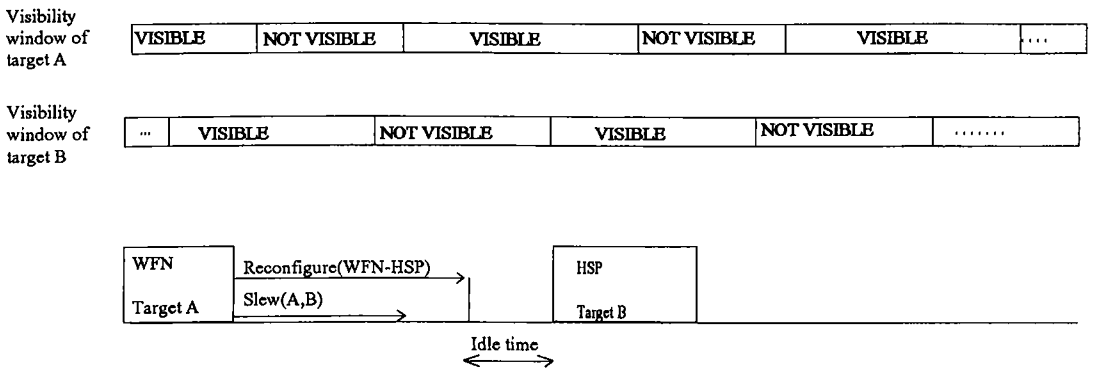

Figure 3.2: Visibility windows

Furthermore data communication and data storage are other constraints that should be considered. However it is not possible to determine the schedule of any of the TDRSS satellites that manages the communication of data to earth. So this constraints can only be satisfied by real-time scheduling.

As stated above there are some constraints that are specified by the proposers. Each proposal may be an observation program that is consist of several different observations. A proposer may specify precedence, maximum or minimum time separation, a specific due date for any of the observations, the duration of the exposures, interruptibility, conditional exposures and contingent exposures as constraints for the programs.

Several preferences are also available in the problem domain, although it is not strictly required to satisfy these preferences completely. For example, it is preferable to schedule the observations when the signal-to-noise ratio is at the desired level. Also it is preferable to avoid from bright light which would possibly damage the instruments or impact science data quality.

Finally, the scheduler must consider the priorities that are assigned by STScI institute to the observations as it was discussed in detail in chapter 2.

3.3 ASSUMPTIONS

Various assumptions are made throughout this thesis. These assumptions do not damage the characteristics of the problem, however simplify the problem domain and ease the handling and understanding the nature of it.

- The reconfiguration time is implicitly accounted as temporal time delay rather than explicitly modelled it as complex power-up/power-down sequences.
- We modelled the reconfiguration time required to start an observation as the maximum of the power-up/power-down sequence durations.
- Only one instrument is operational at any time because of the limited power on board.
- Even though the limitations of pointing at sun complexes the estimation of slewing durations, we assumed that slewing time is a function of angular difference between the preceding two exposures and the angular velocity of the telescope.
- Reconfiguration of the instruments can be managed simultaneously with the slewing of the telescope. So the maximum time of both is used as the overall set-up of HST.
- All the mentioned factors that limit the size of visibility windows, such as SAA, limitations of pointing to sun and moon, etc., 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.

All these stated assumptions reflect the characteristics of the real HST scheduling problem. These assumptions are very similar with the ones that are available in the literature such as Smith and Pathak [57].

Until now we have stated the objective and the constraints of the problem domain. Also the assumptions that are used throughout the thesis are presented. Now we will present the mathematical formulation of the problem and the notation that will be used in the thesis.

3.4 MATHEMATICAL FORMULATION

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

Minimize $\sum_{j=1}^{N} \omega_j x_j$

s.t.

$$\begin{array}{rcllr}
S_{jk} & \geq & b_{jv} z_{jv} & \forall j,k,v & (1) \\
S_{jk} + P_j & \leq & e_{jv} + M(1 - z_{jv}) & \forall j,k,v & (2) \\
\sum_{v=1}^{V} z_{jv} & = & 1 & \forall j & (3) \\
S_{ik} - (S_{j(k-1)} + P_j) & \geq & SC_{ji} & \forall i,j,k & (4) \\
S_{ik} - (S_{j(k-1)} + P_j) & \geq & SL_{ji} & \forall i,j,k & (5) \\
S_{jk} + P_j - D_j & \leq & M x_j & \forall j,k & (6) \\
S_{jk} & \geq & R_j & \forall j,k & (7) \\
S_{jk} & \leq & M y_{jk} & \forall j.k & (8) \\
\sum_{j=1}^{N} y_{jk} & = & 1 & \forall k & (9) \\
\sum_{k=1}^{N} y_{jk} & = & 1 & \forall j & (10) \\
x_j & = & 0,1 & \forall j & (11) \\
y_{jk} & = & 0,1 & \forall j,k & (12) \\
z_{jv} & = & 0,1 & \forall j,v & (13)
\end{array}$$

where

$j = 1,..,N$, $k = 1,..,N$ and $v = 1,..,V$

Decision variables are:

S_{jk} = Starting time of an observation request j at sequence k

$$x_j = \begin{cases} 1, & \text{if an observation request } j \text{ is rejected} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{jk} = \begin{cases} 1, & \text{if an observation request } j \text{ is scheduled at sequence } k \\ 0, & \text{otherwise} \end{cases}$$

$$z_{jv} = \begin{cases} 1, & \text{if an observation request } j \text{ is scheduled during the time window } v \\ 0, & \text{otherwise} \end{cases}$$

Parameters :

N = Total number of observations

w_j =Relative priority of an observation request j

P_j =Requested viewing time of an observation j

T =The length of the planning horizon

b_{vj} =Beginning time of a visibility window v for an observation j

e_{vj} =Ending time of a visibility window v for an observation j

TW_{vj} = Visibility window interval v for an observation j

SC_{ji} =Instrument reconfiguration time from observation request j to observation request i

SL_{ji} =slewing time from observation request j to observation request i

D_j =Specific deadline for an observation request j (i.e. $D_j \leq T$)

R_j =User specified earliest start of an exposure for an observation request j (i.e. $0 \leq R_j < T$)

M =Very large positive number

In this formulation, objective function corresponds to maximizing the scientific return, or 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. If there is any user specified timing requirements for an earliest start time of an exposure, constraint (7) guarantees that an observation request cannot start before its requested time. Constraint sets (8), (9), and (10) 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 sets (11), (12) and (13) give the integrality requirements for some of the decision variables.

Chapter 4

ALGORITHMS

In this chapter we will present the algorithms that we have proposed for generating short term observation schedules of SM projects as well as the nearest neighbor (NN) algorithm that we have used while testing the efficiencies of the proposed algorithms. In section 4.1, we will present the notation that is used throughout this chapter. In section 4.2, we will present the steps of the nearest neighbor algorithm. Next, in section 4.3, the new dispatch heuristic that we have proposed is presented. The beam search and the simulated annealing algorithms are given in sections 4.4 and 4.5, respectively. In section 4.6, we will present the Greedy Randomized Adaptive Search Procedure (GRASP). Finally, a brief summary of this chapter is given in section 4.7.

4.1 NOTATION

In this section we will present the common notation that is used in the following algorithms. The additional notation that is specific for a particular algorithm will be given just before that algorithm.

- **Notation :**

W = Set of all the observations

S_t = Set of observations that are scheduled until the iteration t

U_t = Set of observations that are not scheduled until the iteration t

l_t = last scheduled observation at iteration t

ct_{l_t} = Completion time of the last scheduled observation l at iteration t

st_{jt} = End of setup time of observation j if scheduled at iteration t

$FAST_{jt}$ = First available start time of observation j at iteration t

$slack_{jt}$ = The remaining time available to schedule observation j after iteration t

$\overline{s} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij}}{N^2}$: Average of the set up times

$\overline{p} = \frac{\sum_{i=1}^{N} p_j}{N}$: Average of the processing times

c = constant

$k_1 = c/(2 * \sqrt{\overline{s}/\overline{p}})$

k_2 = constant

The remaining time available to schedule observation j after iteration t, $slack_{jt}$ can be calculated as follows. There are four possible cases;

a. $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$ and $\exists v' \in V$ such that $D_j \in TW_{v'j}$ then

$$slack_{jt} = (e_{v^*j} - st_{jt}) + \textstyle\sum_{v^*+1}^{v'-1}(e_{vj} - b_{vj}) + (D_j - b_{v'j})$$

b. $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$ and $D_j \notin TW_{vj}$ $\forall v \in V$ in this case $v'$is the maximum v that $e_{v'j} < D_j$ then

$$slack_{jt} = (e_{v^*j} - st_{jt}) + \textstyle\sum_{v^*+1}^{v'}(e_{vj} - b_{vj})$$

c. $st_{jt} \notin TW_{vj}$ $\forall v \in V$ in this case $v^*$is the minimum v that $b_{vj} > st_{jt}$ 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. $st_{jt} \notin TW_{vj}$ $\forall v \in V$ in this case v^* is the minimum v that $b_{vj} > st_{jt}$ and $D_j \notin TW_{vj}$ $\forall v \in V$ in this case $v'$is the maximum v that $e_{v'j} < D_j$

$slack_{jt} = \sum_{v^*}^{v'}(e_{vj} - b_{vj})$

4.2 NEAREST NEIGHBOR ALGORITHM

The nearest neighbor (NN) algorithm is proposed by Smith and Pathak [57] for scheduling the over-subscribed systems such as SMS problems. In their research they test the efficiency of the algorithm at HST domain which reflects all the characteristics of SMS problems. NN algorithm is a well-known, simple, dispatch based algorithm widely used at Travelling Salesperson Problems. The basic procedure of the algorithm is selecting the first available candidate for the next step. The motivation behind this approach is mostly, obtaining high resource utilization.

The basic outline of the NN algorithm can be given as follows:

- **Algorithm**

1. Get the initial data, TW_{vj}, S_{0j}, SC_{ij}, SL_{ij}, P_j, D_j

2. Problem initialization : $t = 1$, $S_t = \{\}$, $U_t = W$, $l_t = 0$, $ct_{l_t} = 0$

3. Calculate $st_{jt} = \max\{SC_{l_tj}, SL_{l_tj}\} + ct_{l_t}$, $\forall j \in U_t$

4. For any observation $j \in U_t$ calculate $FAST_{jt}$, there are two possible cases,

a. $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$ in this case $FAST_{jt} = st_{jt}$

or

b. $st_{jt} \notin TW_{vj}\ \forall v \in V$ in this case $v^*$is the minimum v that $b_{vj} > st_{jt}$ and $FAST_{jt} = b_{v^*j}$

5. Select the observation j^* that has the earliest $FAST_{jt}$ and schedule it as early as possible.

6. $l_t = j^*$, $ct_{l_t} = FAST_{j^*t} + P_{j^*}$ and $t = t + 1$

7. Delete observation j^* from the set of unscheduled observations and add it to the scheduled set of observations, i.e. $U_t = U_{t-1} - j^*$ and $S_t = S_{t-1} + j^*$

8. Goto step 3 until the $ct_{l_t} > T$

Note that in step 4, either the end of setup time of the observation j is in the visibility window of observation j as stated in (a), or not as in (b). In the first case first available start time of observation j is just the ending time of the needed setup, however in the second case to start the observation j we must wait until it is visible again.

NN algorithm has an advantage of being simple. The simplicity of the algorithm also affects its computational complexity which is $O(n^2)$. Even in a complex domain like SMS, it works quite fast. However it is far from handling all of the aspects of the problem. For example this algorithm neither considers the weight nor the due date information of the observations. Its myopic nature might prohibit it from good and near optimal solution. The decision is done only with the data available in the current step and the overall interactions of the observations are not considered.

4.3 NEW DISPATCH HEURISTIC

The motivation behind the new dispatch heuristic is taken from the Apparent Tardiness Cost (ATC) rule proposed by Morton and Rachamadugu [54] for the single machine weighted tardiness problem. Even though it needs more calculations then NN and tries to overcome the myopic nature of the dispatch based heuristics with a lookahead feature, it is still simple, myopic, and dispatch based heuristic. The first algorithm we have proposed is an ATC based heuristic with several modifications that are done in order to be applicable to the SMS problem environment. The basic procedure of the proposed algorithm is assigning ATC values to the remaining unscheduled observations at each iteration. Next we schedule the observation with the highest ATC value. The next observation scheduled is deleted from the set of unscheduled observations and added to the scheduled observations. In the next iteration the ATC values of the remaining unscheduled jobs are updated and again the observation with the highest ATC value is selected to be the one that is scheduled. This process goes until the end of the time horizon.

- **Additional Notation :**

ct_{jt} = Completion time of observation j if scheduled at iteration t

ATC_{jt} = ATC value of observation j at iteration t

- **Algorithm**

1. Get the initial data, TW_{vj}, S_{0j}, SC_{ij}, SL_{ij}, P_j, D_j

2. Problem initialization : $t = 1, S_t = \{\}$, $U_t = W$, $l_t = 0$, $ct_{l_t} = 0$

3. Calculate $st_{jt} = \max\{SC_{l_t j}, SL_{l_t j}\} + ct_{l_t}$, $\forall j \in U_t$

4. For any observation $j \in U_t$ calculate $FAST_{jt}$, there are two possible cases,

a. $\exists v^* \in V$ such that $st_{jt} \in TW_{v^* j}$ in this case $FAST_{jt} = st_{jt}$

or

b. $st_{jt} \notin TW_{v^*j}$ $\forall v \in V$ in this case $v^*$is the minimum v that $b_{vj} > st_{jt}$ and $FAST_{jt} = b_{v^*j}$

5. Calculate $slack_{jt}$ as it is described in section 4.1, $\forall j \in U_t$

6. Find the ATC_{jt} ,$\forall j \in U_t$

$ATC_{jt} = \frac{w_j}{p_j} * \exp(\frac{-(FAST_{jt}-ct_{l_t})}{k_1}) * \exp(\frac{-slack_{jt}}{k_2})$

7. Select the observation j^* that has the highest ATC_{jt}

8. For the observation j^* calculate $ct_{j^*t} = FAST_{j^*t} + P_{j^*}$

9. If $ct_{j^*t} < D_{j^*}$ then schedule it as early as possible and goto step 10

else $U_t = U_t - j^*$ and goto step 6.

10. $l_t = j^*$, $ct_{l_t} = ct_{j^*t}$ and $t = t + 1$

11. Add observation j^* to the scheduled set of observations $S = S_{t-1} + j^*$, $U_t = U_{t-1} - j^*$

12. Goto step 3 until the $ct_{l_t} > T$

Note that the proposed rule is a composite dispatching rule that combines the weighted shortest processing time (WSPT) rule, nearest neighbor rule and min-slack rule. It works as the WSPT rule when the observations are away from their due dates 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 due dates of the observations the third term becomes more important and higher ATC values are assigned to the observations with the closer due dates. Same thing is also true for the middle term. If the difference between the set up times is large then this term becomes more urgent and higher ATC values are assigned to the observations that can be scheduled earlier because of their low set up time. Finally the constants c and k_2 are determined by pilot runs. It is found that promising values of these parameters are $c = 0.03$ and $k_2 = 0.0002$. These two parameters are used during the experimental design, that will be discussed in the next chapter.

4.4 BEAM SEARCH

The second algorithm that is proposed as a solution approach is based on a filtered-beam search. In this methodology the decision tree is searched such as B&B however it is also possible to prune the nodes that are not seemed to be promising. As previously mentioned, a generic filtered beam search has two decision parameters: 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 $b * f$ partial schedules that are obtained by temporary addition of these f promising unscheduled observations to the corresponding b partial schedules. Best b of the $b * f$ partial schedules are selected with respect to the global evaluation function scores. And continue this procedure until no more observations can be scheduled to any of the partial schedules.

In this thesis we add another feature to the classical filtered beam search and test its effects to the solution. This modification to the search algorithm is called as the "childwidth". We call the initial observation scheduled at each beam as parent, and childwidth (c) determines the maximum number of the children allowed for each parent. The motivation behind this modification is to prohibit the premature entrapment of local optima that is quite possible after several iterations. In order to provide diverse search of the decision tree we limit the number of beams that originate from the same parent. And this limitation is obtained by setting a parameter c.

Before continuing with the additional notation and the algorithm, we will state the local and global evaluation functions that are used in our beam search algorithm.

local evaluation function: Since at each iteration we will calculate the local evaluation function scores of the all unscheduled jobs for each beam b this function should be quite fast and present good results. We directly use the ATC_j values that we have proposed in the previous section which is shown to work fast and give better results than NN.

global evaluation function: The global evaluation function is the total weight obtained by exploding $b * f$ temporary partial schedules until the end of the time horizon. This explosion of $b * f$ partial schedule is obtained by scheduling each $b * f$ partial schedule with respect to a dispatch rule until the end of the time horizon to find an upper bound value o the scientific return. We used a dispatch rule that is very similar to the ATC_j values that are proposed while exploding each $b * f$ partial schedules. The dispatch rule that is used as follows.

$$ATC_j^e = \frac{w_j}{p_j} * \exp(\frac{-(FAST_{jt} - ct_{l_t})}{k_1})$$

- **Additional Notation:**

S_b = Set of scheduled observations of partial schedule of beam b

U_b = Set of unscheduled observations of partial schedule of beam b

CNS_b = Set of observations that cannot be scheduled at partial schedule of beam b because of due date restrictions

l_b = Last observation scheduled at partial schedule of beam b

ct_{l_b} = Completion time of the last observation that is scheduled at partial schedule of beam b

ct_{jl_b} = Completion time of observation j if scheduled after the observation l_b of the partial schedule of beam b

st_{jl_b} = End of setup time of observation j if scheduled after observation l_b of the partial schedule of beam b

$slack_{jb}$ = Remaining time that is available to schedule observation j at partial schedule of beam b

ATC^{l}_{jb} = Local evaluation score of observation j for the partial schedule of beam b

ATC^{e}_{jb} = Exploding function score of observation j for the partial schedule of beam b

GES_{jb} =Global evaluation function score of the augmented partial schedule obtained by adding observation j to the end of the partial schedule b

$FAST_{jb}$ = First available start time of observation j at partial schedule of beam b

i = Counter

- **Algorithm:**

1. Get the initial state TW_{vj}, S_{0j}, SL_{ij}, SC_{ij}, P_j, D_j

2. Set the beamwidth (b) , filterwidth (f), and childwidth (c)

3. Initialize for each b

$CNS_b = \{\}$, $U_b = w$, $S_b = \{\}$, $l_b = 0$, $ct_{l_b} = 0$, $done_b = False$, $done = False$

4. While not *done* do

4.1. For each b repeat the following

4.1.1. **Procedure** Filter_with_one_step

4.1.1.1. Calculate $st_{jl_b} = \max\{SC_{jl_b}, SL_{jl_b}\} + ct_{l_b}$

4.1.1.2. For any observation $j \in U_b$ calculate $FAST_{jb}$; there are two possible cases

```
        a) $\exists\ v^* \in V$ such that $st_{jl_b} \in TW_{v^*j}$ in this case $FAST_{jb} = st_{jl_b}$
      or
        b. $st_{jl_b} \notin TW_{v^*j}\ \forall v \in V$ in this case $v^*$is the minimum $v$ that $b_{vj} > st_{jl_b}$ and $FAST_{jb} = b_{v^*j}$
      4.1.1.3. Calculate $slack_{jb}\ \forall j \in U_b$
      4.1.1.4. Find the $ATC^l_{jb}$ ,$\forall j \in U_b$
        $ATC^l_{jb} = \frac{w_j}{p_j} * \exp(\frac{-(FAST_{jb}-ct_{l_b})}{k_1}) * \exp(\frac{-slack_{jb}}{k_2})$
      4.1.1.5. $i := 1$
      4.1.1.6. While $i < f$ do
        4.1.1.6.1. Select the observation $j^* \in U_b$ that has the highest $ATC^l_{jb}$
        4.1.1.6.2. For the observation $j^*$ calculate $ct_{j^*lb} = FAST_{jb} + P_{j^*}$
        4.1.1.6.3. If $ct_{j^*l_b} < D_{j^*}$ then goto 4.1.1.6.4
                   else $U_b = U_b - j^*$ and $CNS_b = CNS_b + j^*$ and goto 4.1.1.6
        4.1.1.6.4. $i = i + 1$ , $U_b = U_b - j^*$ , $Filterset_b = Filterset_b + j^*$
  4.2. For each $b$ repeat the following
    4.2.1. **Procedure** Evaluate_the_global_evaluation_scores
      4.2.1.1. For each $j^* \in Filterset_b$ repeat the following
        4.2.1.1.1. Augment $j^*$ to partial schedule of beam $b$
        4.2.1.1.2. Schedule the remaining unscheduled observations to the Augmented partial schedule with respect to $ATC^e_{jb}$
```

4.2.1.1.3. Calculate the total weight of the schedule obtained by explosion and set GES_{j^*b} to this total weight

4.3. Select the best b of $b * f$ partial schedules with respect to the global evaluation scores (GES_{jb}) by considering the limitation of c for the beams that are originating from the same parent

4.4. For each b repeat the following

If $|W| = |S_b| + |U_b| + |CNS_b|$ then $done_b = True$ else $done_b = False$

4.5. $done = \prod_b done_b$

In the Procedure Filter_with_one_step we evaluate the local evaluation function scores of the unscheduled observations for each b partial schedules. We select the best f of them in order to explode and calculate the global evaluation function scores.

After the f promising candidates for each of the b partial schedules are determined, $b * f$ augmented partial schedules are exploded and we select the best b with respect to the global evaluation function scores. As mentioned the global evaluation scores are the total weight of the observations that are scheduled before their due dates for each exploded partial schedules. The dispatch rule mentioned before is used while exploding the augmented partial schedules. Then the best b of the augmented partial schedules are selected. Note that while selecting the best partial schedules the number of childs that belongs to a specific parent is limited with the parameter c (childwidth).

4.5 SIMULATED ANNEALING

Our third proposed algorithm is an application of Simulated Annealing (SA) algorithm to the SMS problem.The basic structure of the algorithm is same as the generic SA given in Chapter 2. In this algorithm we start with an initial solution and try to search the decision tree by a neighborhood search mechanism. We pass to the generated neighbor if the total score of it is greater than the total score of the current schedule on hand. But as a characteristic feature of the SA it is also possible to move to a neighbor that has less total score with the probability of $e^{\frac{-\Delta}{t}}$. This probability is known as the probability of acceptance where Δ is the difference between the total score of the current schedule and the total score of the candidate neighbor. Total score of a schedule is simply the total weight of the observations that are scheduled before their due dates in a given schedule.

For the generation of the initial solution, we used the new dispatch heuristic we have proposed in section 4.3. Now we will present the neighbor generation procedure that we have proposed.

4.5.1 NEIGHBOR GENERATION MECHANISM

In HST there are six different viewing instruments and each instrument has several operating modes. These instruments and the operating modes of the instruments are presented in table 3.1. We have previously mentioned that there are major reconfigurations between the instruments and minor reconfigurations are needed between the modes of each instrument. Our neighbor generation mechanism takes this fact into consideration. Each instrument mode can be viewed as a "family". So totally we have 15 different observation families. We define the family setup matrix that presents the set up times needed between the observations those belong to different families.

We generate the neighbors of the current solution simply by exchanging groups of jobs. The consecutive jobs that belong to same families constitute the job groups in the current solution on hand. First of all with a heuristic we calculate the Exchange Desirability Values between each group of jobs and then we generate the neighbors with respect to the most promising Exchange Desirability Values. This process is presented on a 5 observations and 2 families example.

Observation Number	Needed Instrument	Family Index
1	WF/UV	A
2	WF/UV	A
3	WF/N	B
4	WF/N	B
5	WF/UV	A

Table 4.1: Instrument data of the observations of the example

FAMILY SETUP	A	B
0	300	400
A	-	600
B	600	-

Table 4.2: Family setup values of the example

The viewing instruments that are needed for each observation are as the ones that are presented in table 4.1. The family setups are presented in table 4.2. '0' corresponds to the initial state of the telescope. The values below the families show the needed setup time to be scheduled just after the previous state. Suppose that the current schedule on hand is 5-1-3-2-4. This schedule has the following family structure : A(2) - B(1) - A(1) - B(1). The numbers in paranthesis represent the number of consecutive jobs that belong to the given family. '5' and '1' are from family A whereas 3 is from family B so we represent this structure as A(2) - B(1), and so on.

There are $\binom{4}{2} = 6$ different ways of possible group exchange for the given example. All the possible exchanges, the outcomes of the exchanges and the Exchange Desirability Values are presented in table 4.3. Note that the exchanged groups at each exchange number are represented in bold letters.

Exc. #	Exchange	New Family Structure	New Schedule Structure	Des. Value
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

Table 4.3: Possible exchanges and the outcomes in the example

In table 4.3, New Family Structure represents the new structure that is obtained after the exchange.For example for the exchange number 1, we interchanged the groups of observations A(2) and B(1) in the current schedule. The new schedule now has 3 consecutive jobs that belong to family A (5-1-2) which is represented as A(3). Note that New Schedule Structure just represents the structure of the new schedule. The order of the observations in the paranthesis does not necessarily reflect the exact order. The exact order within the groups are determined by rescheduling each group. For example, for exchange number 1 in the new schedule observation 3 will precede observations 5, 1 and 2, however the second observation is not necessarily observation 5. It is quite possible to schedule observation 1 or 2 before 5 because of the visibility windows and slewing time needed. So rescheduling of group 5-1-2- is required. This rescheduling is done by using the new dispatch heuristic mentioned in section 4.3.

However it is computationally ineffective to reschedule all the possible exchanges. In order to have more efficient algorithm we just reschedule the most promising exchanges. The promising exchanges are determined by the heuristic desirability values given in the table. These exchange desirability values indicate the net gain from the family setups that will be obtained by the corresponding exchange. These gains are calculated as follows.

For exchange number 4, we exchange B(1) and A(1). The saved family setups are $S_{AB}(600) + S_{BA}(600) + S_{AB}(600) = 1800$. Extra family setups are $S_{AA}(0) + S_{AB}(600) + S_{BB}(0) = 600$. Net gain from the exchange is 1200. Note that the numbers in the paranthesis are the family setup times which are given in table 4.2.

We only generate the neighbors that have the highest exchange desirability values. The number of recheduling is determined by the parameter desirable_exchanges_list_width (delw). We use the first win strategy while passing to the next neighbor. The main motivation behind this neighborhood generating mechanism is to augment the groups of jobs that belong to same family and save from major reconfiguration times. By this way we try to overcome

In table 4.3, New Family Structure represents the new structure that is obtained after the exchange.For example for the exchange number 1, we interchanged the groups of observations A(2) and B(1) in the current schedule. The new schedule now has 3 consecutive jobs that belong to family A (5-1-2) which is represented as A(3). Note that New Schedule Structure just represents the structure of the new schedule. The order of the observations in the paranthesis does not necessarily reflect the exact order. The exact order within the groups are determined by rescheduling each group. For example, for exchange number 1 in the new schedule observation 3 will precede observations 5, 1 and 2, however the second observation is not necessarily observation 5. It is quite possible to schedule observation 1 or 2 before 5 because of the visibility windows and slewing time needed. So rescheduling of group 5-1-2- is required. This rescheduling is done by using the new dispatch heuristic mentioned in section 4.3.

However it is computationally ineffective to reschedule all the possible exchanges. In order to have more efficient algorithm we just reschedule the most promising exchanges. The promising exchanges are determined by the heuristic desirability values given in the table. These exchange desirability values indicate the net gain from the family setups that will be obtained by the corresponding exchange. These gains are calculated as follows.

For exchange number 4, we exchange B(1) and A(1). The saved family setups are $S_{AB}(600) + S_{BA}(600) + S_{AB}(600) = 1800$. Extra family setups are $S_{AA}(0) + S_{AB}(600) + S_{BB}(0) = 600$. Net gain from the exchange is 1200. Note that the numbers in the paranthesis are the family setup times which are given in table 4.2.

We only generate the neighbors that have the highest exchange desirability values. The number of recheduling is determined by the parameter desirable_exchanges_list_width (delw). We use the first win strategy while passing to the next neighbor. The main motivation behind this neighborhood generating mechanism is to augment the groups of jobs that belong to same family and save from major reconfiguration times. By this way we try to overcome the myopic nature of the dispatch heuristic that we have used while creating initial solution.

We also modified the neighborhood selection mechanism described above with a mutation mechanism. With a probability called as mutation probability we mutate the current solution obtained after exchange. Mutation is done on a basis of selecting an observation in the current schedule, deleting and adding it to the end of the schedule. We considered two criteria while selecting the observation to be deleted. First one is selecting the observations that are scheduled before the time horizon but after their own due dates. We will refer to this as type I mutation. The second one is selecting the job that consumes highest amount of time. Remember that between each observation there are needed times that comes from the fact of reconfiguration times, slewing times and visibility windows availability. We will call this needed time as the idle time of the telescope. Each observation has a backward idle time that is the time needed to execute the current observation after the previous observation and forward idle time, that is the time needed to execute the next observation after the current observation. The total idle time of the observation calculated by adding the backward and forward idle times corresponding to the observation. After we determine the total idle times we create the deletion_observation_list (DOL). In this list we add 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). After the creation of the list we delete and add each observation in the list to the end of the schedule one by one. We reschedule the obtained sequence and calculate the total weight of observations that are scheduled before their due dates. And with the probability acceptance function we either generate the next neighbor or select the next mutation candidate. We again use the first wins strategy while mutating the current schedule.

The aim of mutation is both obtaining diversity of search in the decision tree and check if it is more profitable to schedule the jobs just after the finishing of the time horizon by deleting one job from the current schedule on hand.

- **Additional Notation:**

$\alpha =$ Temperature decreasing rate

$T_0 =$ Initial temperature

$T_t =$ Temperature at iteration t, i.e. $T_t = \alpha * T_{t-1}$

$del =$ List of most promising exchange pairs

$delw =$ Width of the most promising exchanges list

$DOL =$ List of most promising observations to mutate

$DOLW =$ Width of DOL

$mp =$ Probability of mutating the current solution

$IS_0 =$ Initial schedule

$BS =$ Best schedule found

$CS_t =$ Current schedule at iteration t

$FS_{ab} =$ Required major reconfiguration time between an observation belonging to family a with another observation belonging to family b

$FS =$ Matrix of family setups

$frozen =$ Boolean variable. If it is false then repeat the search, else end

$score_c =$ The total weight of the observations that are scheduled before their due dates in a given schedule c

$CN =$ Candidate neighbor obtained from exchange

$\Delta E =$ Difference between the score of current solution and the candidate neighbor

t = iteration number of annealing procedure

k = iteration number of mutating procedure

- **Algorithm:**

1. Get the initial state, TW_{vj}, S_{0j}, SL_{ij}, SC_{ij}, P_j, D_j

2. Set the initial temperature (T_0), alpha (α), desirable_exchange_list_width ($delw$), deletion_observation_list_width ($DOLW$), mutation probability (mp)

3. Create the initial schedule (IS_0) by scheduling all the observations with the new dispatch heuristic mentioned at section 4.3

4. Set $t = 1, frozen = false$ and current schedule $CS_t = IS_0$

5. Calculate the score of CS_t ($Score_{CS_t}$)

6. Set $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$

7. $T_t = T_0$

8. Establish FS matrix

9. While not ($frozen$) do

9.1. Establish desirable_exchange_list (DEL) with respect to heuristic desirability values

9.2. Select one pair randomly from DEL and exchange

9.3. Set the candidate neighbor (CN) to the sequence obtained after exchange

9.4. Reschedule CN

9.5. Calculate the score of the CN ($Score_{CN}$)

9.6. Evaluate $\Delta E = Score_{CS_t} - Score_{CN}$

```
    9.7. If $\Delta E \leq 0$ then set $CS_t = CN$ and $Score_{CS_t} = Score_{CN}$
    9.8. Else if $\Delta E > 0$ then set $CS_k = CN$ and $Score_{CS_k} = Score_{CN}$ with probability $e^{\frac{-\Delta E}{T_t}}$. If not accepted goto 9.2
    9.9. If $Score_{CS_t} > Score_{BS}$ then $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$
    9.10. With probability $mp$ mutate $CS_t$
    9.11. If no mutation goto step 9.13
    9.12. Procedure Mutate
    9.13. Set $t = t + 1$ and $T_t : T_{t-1} * \alpha$
    9.14. If $T_t <$ $critical\ temperature$ then $frozen := true$
```

With the mutation probability at step 9.12 we mutate the current solution on hand. The mutation procedure is as follows:

Procedure Mutate:

```
1. Set $k := 0$
2. While $k < DOLW$ do
    2.1. Establish deletion_observation_list ($DOL$)
    2.2. Select one observation from the list ($DOL$) randomly
    2.3. Delete the selected observation from the current schedule and add it to the end
    2.4. Set the candidate neighbor ($CN$) to the new sequence obtained after 2.3
    2.5. Reschedule $CN$
    2.6. Calculate the score of $CN$ ($Score_{CN}$)
```

2.7. Evaluate $\Delta E = Score_{CS_t} - Score_{CN}$

2.8. If $\Delta E \leq 0$ then set $CS_t = CN$, $Score_{CS_t} = Score_{CN}$ and $k = DOLW$

2.9. Else if $\Delta E > 0$ then set $CS_t = CN$, $Score_{CS_t} = Score_{CN}$ and $k = DOLW$ with probability of $e^{\frac{-\Delta E}{T_t}}$. If not accepted then $k := k + 1$

3. If $Score_{CS_t} > Score_{BS}$ then $BS = CS_t$ and $Score_{BS} = Score_{CS_t}$

4.6 GRASP

Finally as the fourth algorithm we have proposed Greedy Randomized Adoptive Search Procedure (GRASP) for generating short term schedules of SM projects. As mentioned in chapter 2, GRASP consists of two phases. The first phase is the construction phase and the second phase is the iterative improvement.

In the construction phase GRASP builds a feasible schedule iteratively with respect to a greedy function. The main difference here is at each step we do not select the observation that has the highest greedy function score but construct a Restricted Candidate List (RCL) and select one observation from this list randomly. $RCL = \{j : \alpha_j \geq \alpha\}$ where α_j is the ratio of the greedy function score of observation j to the highest greedy function score obtained at that step and α is the ratio parameter that is predefined. Since we select the next observation randomly from RCL at each step and construct a feasible schedule every time we construct a schedule, we construct a different schedule than the previous ones.

Second phase is the iterative improvement procedure that tries to locally optimize the schedule obtained from the construction phase. In this phase we propose an algorithm that is similar to the neighborhood generation algorithm of SA presented in the previous section. However for GRASP second phase we do not use mutation and only pass to the neighbor if it gives a better solution than the current schedule. Remember that it was also possible to move to a neighbor that gives worse solution.

We modify the generic GRASP by not applying the second phase each time we construct a schedule at the first phase. We just allow to apply the second phase if the constructed schedule is a promising schedule. In order to decide if the constructed schedule is a promising one or not we define "allowable_percentage" and check if the ratio of the total objective function value of the current solution to the total objective function value of the best schedule found up to then is greater than this percentage. If it is greater than this percentage, then we apply the second phase. However, if it does not satisfy this condition we turn back to the first phase and construct a new schedule. Suppose that the allowable_percentage is 95% and the best solution on hand has total weight of the observations that are scheduled before their due dates 100. Then if the newly constructed schedule has total objective function value greater than 95 we apply the second phase. If it has a smaller objective function value then we do not apply the second phase and return to phase one in order to construct a new schedule. The main motivation of this limitation is searching more nodes of the decision tree rather than spending much time to improve a schedule that is not strongly promising.

- **Additional Notation**

k = Iteration number

T = Maximum number of iterations

α = The rate used when constructing restricted candidate list (RCL)

$Score_c$ = Total weight of observations that are scheduled before their due dates in a given schedule c

BS_k = Best schedule obtained until iteration k

CS_k = Current solution at iteration k

$Allowable\_percentage$ = The parameter that determines the promising schedules constructed at phase 1 in order to apply phase 2

$Maximum\_number\_of\_exchange$ = maximum number of exchanges that will be done at phase 2

t = Iteration number that is used in phase 1

ATC_{ht} = Highest greedy function score at iteration t

RCL_t = Restricted candidate list at iteration t

j_{ht} = Observation j that has the highest greedy function score at iteration t

ct_{jt} = Completion time of observation j if scheduled at iteration t

m = Iteration number that is used in phase 2

DEL = List of most promising exchange pairs established with respect to the heuristic exchange_desirability values

$delw$ = Width of DEL

CN = Candidate neighbor obtained from exchange

- **Algorithm**

```
1. Get the initial data $TW_{vj}$, $S_{0j}$, $SL_{ij}$, $SC_{ij}$, $P_j$, $D_j$

2. Set the maximum number of iterations ($T$), $\alpha$, $Maximum\_number\_of\_exchanges$, $Allowable\_percentage$ and $delw$

3. $k = 1$

4. $Score_{BS_k} = 0$

5. While $k < T$ do
```

5.1. **Procedure** Construct_the_greedy_randomized_schedule

5.1.1. Get the initial data, TW_{vj}, S_{0j}, S_{ij}, P_j, D_j

5.1.2. Problem initialization : $S_t = \{\}$, $U_t = W$, $l_t = 0$, $ct_{l_t} = 0$, $t = 1$

5.1.3. Calculate $st_{jt} = \max\{SC_{l_tj}, SL_{l_tj}\} + ct_{l_t}, \forall j \in U_t$

5.1.4. For any observation $j \in U_t$ calculate $FAST_{jt}$, there are two possible cases,

a. $\exists v^* \in V$ such that $st_{jt} \in TW_{v^*j}$ in this case $FAST_{jt} = st_{jt}$

or

b. $st_{jt} \notin TW_{v^*j}$ $\forall v \in V$ in this case $v^*$is the minimum v that $b_{vj} > st_{jt}$ and $FAST_{jt} = b_{v^*j}$

5.1.5. Find the ATC_{jt} ,$\forall j \in U_t$

$ATC_{jt} = \frac{w_j}{p_j} * \exp(\frac{-(FAST_{jt}-ct_{l_t})}{k_1})$

5.1.6. Select the observation j_{ht} that has the highest ATC_{jt}

5.1.7. $ATC_{ht} = ATC_{j_{ht}t}$

5.1.8. Set $RCL_t = \{j : \frac{ATC_{jt}}{ATC_{ht}} > \alpha\}$

5.1.9. Select j^* randomly from RCL_t

5.1.10. For the observation j^* calculate $ct_{j^*t} = FAST_{j^*t}+P_{j^*}$. Schedule it as early as possible

5.1.11. $l_t = j^*$ and $ct_{l_t} = ct_{j^*t}$

5.1.12. $t = t + 1$

5.1.13. Add observation j^* to the scheduled set of observations $S_t = S_{t-1} + j^*$, $U_t = U_t - j^*$

```
    5.1.14. Goto step 5.1.3 until $t > N$
  5.2. Set $CS_k$ to the constructed schedule
  5.3. Calculate $Score_{CS_k}$
  5.4. If $Score_{CS_k} > Score_{BS_k}$ then set $BS_k = CS_k$ and $Score_{BS_k} = Score_{CS_k}$
  5.5. If $Score_{CS_k} < Allowable\_percentage * Score_{BS_k}$ then go to 5.7
  5.6. **Procedure** Local_optimization_phase
    5.6.1. Set iteration number $m = 1$
    5.6.2. While $m < Maximum\_number\_of\_exchange$ do
      5.6.2.1. Establish desirable_exchange_list ($DEL$) with respect to heuristic desirability values as mentioned in the previous section
      5.6.2.2. Select one pair randomly from $DEL$ and exchange
      5.6.2.3. Set the candidate neighbor ($CN$) to the sequence obtained after exchange
      5.6.2.4. Reschedule $CN$
      5.6.2.5. Calculate the score of $CN$ ($Score_{CN}$)
      5.6.2.6. If $Score_{CN} > Score_{CS_k}$ then set $CS_k = CN$ and $Score_{Cs_k} = Score_{CN}$
      5.6.2.7. $m = m + 1$
    5.6.3. If $Score_{CS_k} > Score_{BS_k}$ then set $BS_k = CS_k$ and $Score_{BS_k} = Score_{CS_k}$
  5.7. $k = k + 1$
```

6. End.

4.7 SUMMARY

Until now we have presented the steps of the proposed algorithms. In the next chapter we will perform an experimental design in order to test the efficiencies of the proposed algorithms under different experimental conditions. We will also try to determine the effects of the specific parameters of the algorithms on the objective function score and the corresponding computational time requirements.

Chapter 5

COMPUTATIONAL RESULTS

The algorithms presented in Chapter 4 were coded in Pascal language and compiled with Sun Pascal Compiler on a Sparc Station 10 under SunOS 5.4. In this chapter we perform an experimental design and compare the efficiency of the proposed algorithms along with the NN algorithm on two basis. The first one is the objective function values and the second one is the CPU times of the algorithms. The objective function value of each algorithm is equal to the ratio of the total weight of the observations that are not managed to be scheduled before their due dates to the total weight of the observations.

5.1 EXPERIMENTAL CONDITIONS

There are five experimental factors that can affect the efficiencies of the proposed algorithms. These factors are listed in table 5.1. Since there are five factors that can affect the efficiencies of the algorithms, our experimental design is 2^5 full-factorial experimental design corresponding to 32 combinations. The number of replications for each combination is taken as 10, giving 320 different experiments.

The number of observations most likely would affect the computation times.

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

Table 5.1: Experimental Factors

Oversubscription rate is one of the factors that determines the planning horizon and most likely would affect the objective function values of the algorithms and the computational times. Reconfiguration time is another factor that would affect the objective function values of the algorithms. Reconfiguration time also determines the planning horizon so most likely will also affect the computational times. Due to the technological facts HST needs high reconfiguration times. However one of the aim of this research is proposing efficient algorithms for updated space mission projects which do not need very high reconfiguration times between the instruments. For high reconfiguration times case, major reconfiguration times are selected randomly from the interval UN$\sim$[5000,12000] seconds and minor reconfiguration times are selected randomly from the interval UN$\sim$[1500,4000] seconds, where UN stands for the uniform distribution. On the other hand, for the low reconfiguration times case, major reconfiguration times are selected randomly from the interval UN$\sim$[1600,4000] seconds and the minor reconfiguration times are selected randomly from the interval UN$\sim$[800,2000] seconds. Due date percentages determine the user specific due dates that are before the prespecified time horizon. For due date percentages 0 there is no observation that have user imposed due date whereas for 5, 5% of the observations that are selected randomly have due dates that are selected randomly from the interval UN$\sim$[0.25 * time horizon, time horizon] seconds. As mentioned in Chapter 2, the observations have weights that are determined at STScI. However the observations are divided into "high", "medium" or "supplemental" groups. There are various ways of assigning values to each priority groups. We set them in two different ways. First one is 1-2-3 and the second alternative is 1-5-9 where the first values in both sequence are assigned to the "supplemental observations", the second values to "medium priority observations" and the third values to the "high priority observations". The weights are assigned randomly to the observations with the probabilities of 0.2, 0.6, 0.2 for high priority, medium priority, and supplemental observations, respectively. Time horizon is a function of number of jobs, oversubscription rate and reconfiguration times. We determined the time horizons for each experiment by scheduling the observations with respect to NN algorithm and we divide the obtained value to 1.2 for 20% oversubscription rate and divide it to 1.4 for 40% oversubscription rate.

Other variables were treated as fixed parameters and generated as follows:

- The visibility windows, right ascension and declination data of 76 observations are real data and obtained from Prof. Stephen Smith, Carnegie-Mellon University. Extra 39 observation data in order to obtain 115 observations are generated from the real 76 observation data by replicating 39 of them.
- The slewing times needed between each pair of the observations are calculated by the procedure defined at chapter 3.
- The processing times are generated randomly by selecting from the interval UN$\sim$[100,600] seconds.
- Required instrument configurations are assigned to the jobs randomly by considering the percentages given in table 3.1.

5.2 PARAMETERS OF THE ALGORITHMS

As it was presented in chapter 4 most of the proposed algorithms have several parameters. Some of the parameters are set to specific values. However we specify several different values for some of the parameters and determined the effect of them to the performance of the corresponding algorithm. These parameters are as follows:

- For the NN algorithm we do not have any parameters to be specified.

- For the new dispatch heuristic (NDH) we set $c = 0.03$ and $k_2 = 0.0002$. The values of these parameters are determined after numerous pilot runs. We tested one thousand numbers between 0.001 and 1 for c. After the determination of best c, we tested 10000 numbers between the values of 0.00001 and 0.1 for k_2.

- For the Beam Search algorithm we set 2 different values for each of the parameters; beamwidth, (b), filterwidth, (f), and childwidth, (c). So totally we have tested $2^3 = 8$ different Beam Search algorithms. These algorithms are specified in table 5.2.

- For the GRASP algorithm we tested the effect of α (the rate that is considered while creating the RCL) and T (the number of total iterations). We set two different values to each of these parameters. Furthermore, we also tested the effect of second phase, i.e. local optimization. In the first four of the algorithms we did not use the second phase where as for the next four of the algorithms we allowed the second phase with the allowable_percentage of 97%. The parameter of maximum_number_of_exchanged that is used in the second phase is set to 7. And the desirable_exchange_list_width is set to 6. The parameters of the GRASP and the corresponding algorithms are presented in table 5.3

- For the simulated annealing algorithms we tried to examine the affect of the mutation_rate. Four different mutation rates are determined and tested. After pilot runs it is determined that $\alpha = 0.998$ and initial temperature, $T_0 = 5$, give good results and reasonable computational times. In the pilot runs we tested the α values between 0.99 and 0.999. And we tested three different initial temperature values namely 3, 5, and 8. Other parameters are set to constants such as desirable_exchanges_list_width = 10, and deletion_observation_list_width=3. The parameters of the SA and the corresponding algorithms are presented in table 5.4.

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

Table 5.2: Parameter settings of different Beam Search algorithms

Algo	**Al._per.**	α	T
G1	No	0.2	250
G2	No	0.2	500
G3	No	0.6	250
G4	No	0.6	500
G5	97%	0.2	250
G6	97%	0.2	500
G7	97%	0.6	250
G8	97%	0.6	500

Table 5.3: Parameter settings of different GRASP algorithms

Algo	**mutation rate**
Sno	no
S20	20
S50	50
S100	100

Table 5.4: Parameter settings of different SA algorithms

5.3 COMPUTATIONAL RESULTS

In this section we will discuss the computational results obtained from 320 randomly generated experiments. The results are presented in the appendices in detail. In appendix A we have presented the percentages of unscheduled weights of each algorithm for each experimental setting. The values in tables A.1, A.2, A.3 and A.4 are the averages of 10 replications that correspond to each one of the 32 experimental setting.

Note that there are 5 different experimental factors as mentioned in section 5.1. Each experimental factor is set to high and low values. So there are 160 experiments out of 320 experiments that correspond to high state of each particular experimental factor whereas the remaining 160 experiments correspond to the low state of that particular experimental factor. The averages of the percentages of unscheduled weights of 160 experiments that correspond to each state of the each experimental factor are also presented in appendix A (see tables A.5, A.6, A.7 and A.8).

The percentages of unscheduled weights for each experimental run is calculated as follows;

Percentage of unscheduled weights $= \frac{(TW-OFV)}{TW}$

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.

In appendix B, we present the computational times of the algorithms. Note that the times are given in milliseconds. The construction of appendix B is same as appendix A. In tables B.1, B.2, B.3 and B.4 we present the computational times of the algorithms for each one of the 32 different experimental combinations. The values presented in these tables are the averages of the 10 replications. In tables B.5, B.6, B.7 and B.8 we present the averages of computational times of 160 experiments that correspond to each state of the each experimental factor.

Table 5.5 represents the minimum, average and maximum values of the unscheduled weight percentages and the computational times of 22 different algorithms, namely NN, New dispatch heuristic, 8 beam search algorithms, 8 GRASP, and 4 simulated annealing algorithms. The minimum (Min) and maximum (Max) values correspond to one of the 32 experimental combination that has the minimum and maximum values respectively. These values are the averages of 10 replications that correspond to the particular experimental combination. The average (Ave.) values in table 5.5, correspond to the average values of the 320 different experiments. Our discussion on the results will mainly depend on table 5.5.

We also presented the statistical analysis results of the objective function values and computational time data obtained from the experimental design. The tables of these analyses are presented in appendix C. Note that the objective function values of each experiment are represented as the total weight of the observations that are scheduled before their due dates. From the statistical analysis of the objective function data we find out that there is 99.9% of correlation between the beam search algorithms, 99.5% of correlation between the GRASP algorithms and 99% correlation between the simulated annealing algorithms. The correlation tables among the beam search algorithms, among the GRASP algorithms and among the simulated annealing algorithms are presented in tables C.1, C.2, and C.3, respectively. Therefore, for the further analysis of ANOVA results we have used one representative from each type of algorithm. The significance levels (p) and F values for the experimental factors are presented in tables C.4 to C.8 for each algorithm.

The ANOVA analysis of the experimental factors showed that all of the experimental factors except the fourth one, which is the case of assigning due dates to the observations or not, were significant for the objective function values with the significance $p \leq 0.000$. Only exception is that the third factor, which is the reconfiguration time, is significant for the objective function values of the simulated annealing algorithms with the significance $p \leq 0.017$. ANOVA analysis of the computational times showed that the first three factors that determine the planning horizon affect the computational times of the algorithms with the significance of $p \leq 0.000$. Also as an interesting result we find out that the weight assignment of the observations affects the computational time of the simulated annealing algorithm with $p \leq 0.000$ and the computational times of the beam search and GRASP algorithms with the significance of $p \leq 0.001$. Now let us briefly explain the reason behind this observation.

Note that the weight assignment of 1-5-9 leads to more diverse total weight of the observations than the weight assignment of 1-2-3. The difference between the weights of two schedules is greater in the first case. Remember that in simulated annealing algorithms we were accepting to pass to the next neighbor with the probability of $e^{\frac{-\Delta}{T}}$. For the first assignment case of 1-5-9, the schedules that have lower objective function values will have bigger Δ values which will lead to lower probability of assignment. So the annealing procedure will require a higher computational time. For the beam search algorithm, we will explain the reason with an example. For example, we are nearly at the end of the time horizon and we have five candidate observations. Suppose that four of them are supplemental but require lower reconfiguration times and the fifth one is a high priority observation and requires high reconfiguration time. Suppose that the remaining time only allows to schedule the first four successively or only the fifth one. For the weight assignment of 1-2-3 the global evaluation function score of scheduling first four will add 4 to the weight of the partial schedule where as scheduling the fifth one will only add 3. So with respect to the global evaluation function score beam search algorithm will prefer to schedule the first four and the total number of additional iterations will be four. However for the weight assignment of 1-5-9, scheduling the fifth one will increase the total weight of the partial schedule by 9, where as scheduling the first four will still increase it by 4. So for the latter weight assignment case, beam search algorithm will choose to schedule only the fifth one, and the algorithm ends since the time horizon do not allow to schedule any more observations. So different weight assignment procedures may lead to different schedules, which lead to different number of iterations, hence require different computational times. We can make a similar observation for the GRASP algorithms since different weight assignment procedures will lead to different schedules which will lead

	unsch.	wt.	per.	Comp.	time	(milisc)
Algo	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 5.5: Unscheduled percentages and the computational times

to different computational times. The ANOVA results of five algorithms are available in tables C.4, C.5, C.6, C.7, and C.8 that correspond to NN, NDH, beam search, GRASP and simulated annealing algorithms, respectively.

Finally in appendix C, paired sampled t-test results that are done between the algorithms to identify the significance level of the differences between the algorithms, and within the algorithms to identify the significance levels of different parameter settings are presented.

From these experiments we concluded the following results.

5.3.1 NEAREST NEIGHBOR

- NN algorithm has the worst unscheduled weight percentage value on the average of 320 experiments (see table 5.5). On the average the unscheduled weight percentage is 0.245. Where as it has the best computational time which is 16 milliseconds on the average of the 320 experiments as expected. It has the minimum value of unscheduled weight percentage 0.122 at the experimental combination of 0-0-1-0-0 (see table A.1). Note that 0.122 is the average of 10 replications that correspond to the mentioned experimental combination. This experimental combination corresponds to 76 observations, low oversubscription rate, low reconfiguration time case, no due dates for the observations and for the weight assignment of 1-2-3 (see table 5.1). It has the maximum value of 0.312 at the experimental combination of 0-1-0-1-1. This experimental combination corresponds to 76 observations, high oversubscription rate, high reconfiguration time case, the case where there exists due dates for some of the observations and for the weight assignment for the observations of 1-5-9 (see table 5.1).

- We refer to the states of each experimental factor that the algorithms give higher values of the percentage of unscheduled weights on the average of the corresponding 160 experiments as the hard instances. The hard instances for NN algorithms are as follows; 115 observations, high oversubscription rate, high reconfiguration times case, the case where some of the observations have due dates and for the weight assignment for the observations of 1-2-3 (see table A.5). These hard instances are also same for the remaining 21 algorithms (see table A.5, A.6, A.7 and A.8). In fact it is quite meaningful to have these conditions as the hard instances. For high reconfiguration times case, since the required setup times between the observations are high, less number of observation can be scheduled. It is again true for the high oversubscription rate case since the time horizon is tighter. For the case where some of the observations have due dates, some observations cannot be scheduled before their due dates which also result with less number of scheduled observations. To have 115 jobs as the hard instances shows us the myopic nature of the algorithms. Finally since all the algorithms, except NN, consider the weights that are assigned to the observations while scheduling, it is also expected that these algorithms will perform better for the case of weight assignment of 1-5-9.

- As mentioned NN algorithm gives smaller unscheduled weight percentages on the average of 160 experiments that correspond to the weight assignment of 1-5-9 than the unscheduled weight percentages on the average of remaining 160 experiments that correspond to the weight assignment of 1-2-3. In fact this was not an expected result. Since NN does not consider the weights that are assigned to the observations it was expected that it would perform better for the instances where the relative weight differences are smaller, namely for the weight assignment of 1-2-3. However, NN algorithm performs nominally better for the case of weight assignment of 1-5-9. When we compare it with the other proposed algorithms, which all consider the individual observation weights while scheduling, we conclude that it performs relatively better at the case of weight assignment of 1-2-3. For example, for the NN algorithm the average of the unscheduled weight percentages of the 160 experiments that correspond to weight assignment of 1-2-3 is 0.246 and the average of the unscheduled weight percentages of the remaining 160 experiments that correspond to weight assignment of 1-5-9 is 0.245. For the B7 algorithm, which gives the smallest unscheduled weight percentage value on the average of the overall 320 experiments (see table 5.5), the average of the unscheduled weight percentages of the 160 experiments that correspond to weight assignment of 1-2-3 is 0.181 and the average of the unscheduled weight percentages of the remaining 160 experiments that correspond to weight assignment of 1-5-9 is 0.170. When we compare NN algorithm with B7 algorithm, we can see that B7 algorithm improves NN algorithm by 7.5% (0.245-0.170) for the case of weight assignment of 1-5-9 and by 6.5% (0.246-0.181) for the case of weight assignment of 1-2-3. So we can conclude that B7 algorithm improves NN algorithm more for the case of weight assignment of 1-5-9, which means that NN relatively performs better at the case of weight assignment of 1-2–3. Note that this oservation is not only true for the B7 algorithm, but also true for the remaining 20 algorithms (see tables A.5, A.6, A.7 and A.8).

- As mentioned earlier NN algorithm has the smallest computational time on the average of the 320 experiments.

5.3.2 NEW DISPATCH HEURISTIC

- New dispatch heuristic (NDH) gives better objective function value than the NN algorithm on the average. The average of the unscheduled weight percentages of the 320 experiments is 0.232 for the new dispatch heuristic (see table 5.5). It improves the average of the percentage of unscheduled weights of the observations of the NN algorithm 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 and with the significance level (p)$\leq$ 0.000 (see table C.9). However due to the more complex calculations its computational time is slightly greater than the NN algorithm. Note that on the average of 320 experiments, the computational time of the NN algorithm is 16 milliseconds whereas the computational times of the new dispatch heuristic is 51 milliseconds (see table 5.5). The corresponding *t value* of the computation time comparison for the pair NN-NDH is -37.73 with the significance $p \leq 0.000$ (see table C.9).

- The new dispatch heuristic has the minimum value of unscheduled weight percentage of 0.105 (see table A.1) at the same experimental combination with the NN algorithm, namely 0-0-1-0-0. It has the maximum value of unscheduled weight percentage at experimental combination of 0-1-0-0-0, which is 0.296 (see table A.1). This experimental combination corresponds to 76 observations, high oversubscription rate, high reconfiguration time case, no due dates for the observations and for the weight assignment to the observations of 1-2-3 (see table 5.1).

- Note that since new dispatch heuristic gives better percentage of unscheduled weights than NN on the average of 320 experiments, we will use this dispatch rule rather than the NN algorithm in the remaining proposed algorithms to generate an initial schedule.

5.3.3 BEAM SEARCH ALGORITHM

- The averages of the percentages of unscheduled weights of the 320 experiments show that beam search algorithms perform better than the other competing algorithms (see table 5.5). Furthermore, B7 algorithm is the best among the other beam search algorithms with respect to the above criterion. Remember that B7 algorithm has the parameters of b=6, f=10 and c=3. The overall average of the percentages of the unscheduled weights for B7 algorithm is 0.173 (see table 5.5). So it improves the NN algorithm by 7.2% (0.245-0.173) on the overall average. The corresponding *t value* to the pair B7-NN is 23.31 with the significance $p \leq 0.000$ (see table C.9). The highest improvement on objective function value of the NN algorithm done by B7 algorithm is at the experimental combination of 1-1-1-1-1 (see table A.1 and A.2). In this experimental combination, NN algorithm gives unscheduled weight percentage of 0.288, whereas B7 algorithm gives 0.196. Note that the stated unscheduled weight percentages are the averages of the 10 replications that correspond to the mentioned experimental combination. So the improvement is 9.2% (0.288-0196). This experimental condition corresponds to the 115 observations, high oversubscription rate, low reconfiguration times case, the case where some of the observations has due dates and the weight assignment to the observations is 1-5-9. This is quite meaningful since beam search algorithms, so the B7 algorithm, considers the weights and the due dates that are assigned to the observations while scheduling. Furthermore, B7 algorithm reduces the myopic nature of the heuristics with the help of the global evaluation function mechanism and can improve the objective function value of NN algorithm more for the cases of low setup configuration times, higher number of jobs and higher oversubscription.

- To restrict the number of beams that originates from the same parent, we used a childwidth parameter, which improves the overall average of the percentage of unscheduled weights of the beam search algorithms. Note that B1, B3, B5 and B7 are the beam search algorithms that have a smaller childwidth parameters, where as B2, B4, B6 and B8 are the beam search algorithms that have the same beamwidth and filterwidth parameters with the preceding algorithms with higher childwidth parameters. From table 5.5, we conclude that childwidth parameter improves the average of the percentage of the unscheduled weights by 0.33%. The corresponding *t values* to the pairs B1-B2, B3-B4, B5-B6, and B7-B8 are 6.17, 5.95, 5.89, and 6.69, respectively, with the significance of $p \leq 0.000$. (see table C.10). The computational times do not differ too much (see table 5.5 and table C.9). From the paired samples t-test results, we find that the difference between the computational times for the pairs B1-B2 and B3-B4 are not significant. For B1-B2 the corresponding *t value* is 2.14 with the significance of $p \leq 0.034$ and for B3-B4 the corresponding *t value* is 0.88 with the significance of $p \leq 0.380$ (see table C.10). For the pairs B5-B6 and B7-B8 the differences are significant with the significance $p \leq 0.000$. The corresponding *t values* are 6.91 and -6.38, respectively. Note that restriction with childwidth parameter leads to different schedules which lead to different number of iterations. So the childwidth parameter either increases or decreases the computational time.

- As mentioned above, the best algorithm with respect to the overall average of the unscheduled weight percentages is B7 with the parameters of b=6 and f=10. These two values are the highest values that are used while testing the beam search algorithms. However we cannot conclude that higher beamwidth and filterwidth values always lead to better objective function values. For example, for the experimental combination of 0-1-0-0-1, B1 algorithm with the parameters b=4, f=6 and c=3, has an average of the percentages of unscheduled weights of the 10 replications, 0.232. Whereas the corresponding value for B7 is 0.235 which is worse than B1. This is quite natural and known as the fact of "myopic nature of the global evaluation function value". But on the average high beamwidth and high filterwidth parameters have more chance to give better objective function values. From the paired samples t-test we find out that the differences of the objective function values among the beam search algorithms with different beamwidth and filterwidth are not significant (please refer to table C.10 for the corresponding *t values* and *p*).

- From the above discussion we can conclude that beam search algorithms improve the objective function value of the NN algorithm significantly but the trade off is the high computational time. For example the computational time of B7 is 36789 milliseconds on the average of the 320 experiments, however NN algorithm has computational time of only 16 milliseconds on the average of the 320 experiments (see table 5.5). The corresponding *t value* $= 32.47$ with the significance $p \leq 0.000$. An important factor that affects the computational time of the beam search algorithm is the beamwidth and the filterwidth. As the beamwidth or the filterwidth increases the computational time of the beam search algorithm increases (see table 5.5). This is mainly because, as beamwidth and filterwidth increases then the nuber of nodes that can be searched also increases, which lead to higher computational times. These increases are also statistically significant for all cases with the significance of $p \leq 0.000$ (for the corresponding *t values* refer to table C.10).

5.3.4 GRASP

- GRASP algorithms also improve the NN algorithm on the overall average of the unscheduled weight percentages (see table 5.5). However these average values are not as good as the beam search algorithms (see table 5.5). The best GRASP algorithm with respect to the objective function value is G8. The average of the unscheduled weight percentages of the 320 experiments for G8 is 0.178 and it is worse than the B7 algorithm. The corresponding *t value* to the pair G8-B7 is -4.29 with the significance of $p \leq 0.000$ (see table C.9). However out of 32 different experimental combinations at 16 experimental combinations, G8 gives better objective function values than B7. These 16 experimental combinations all have high reconfiguration time state (see tables A.2 and A.3). Table 5.6 shows the unscheduled weight percentages and the computational times of the algorithms for the high reconfiguration time and low reconfiguration time cases. The unscheduled weight percentages are the average of 160 experiments that correspond to each state. For the high reconfiguration time case, we find that G8 has the best average of the unscheduled weight percentages of the corresponding 160 experiments, which is 0.191 (see table 5.6). At this case it improves B7 algorithm by 0.5%. The corresponding *t value* to the pair G8-B7 is 5.92 with the significance $p \leq 0.000$ (see table C.13). This is mainly due to the local optimization phase of the GRASP algorithm which depends on the "family scheduling concept". Note that family scheduling concept becomes more important when the setup times between the families increase.

- We see that as we increase the iteration numbers we obtain better objective function values (see table 5.5). Note that G1, G3, G5 and G7 are the GRASP algorithms with the iteration number of 250 where as G2, G4, G6 and G8 are the GRASP algorithms with the iteration number of 500. The improvements of the objective function values of the pairs G1-G2, G3-G4, G5-G6, and G7-G8 are significant with $p \leq 0.000$ and the corresponding *t values* are -5.82, -7.69, -5.23, and -7.28 respectively. However in this case we loose from the computational time (see table 5.5). The differences between the computational times are significant with significance of $p \leq 0.000$. The corresponding *t values* are -45.43, –44.10, -35.76, and -27.10, respectively (see table C.11)

- Also $\alpha = 0.6$ leads to better objective function scores than $\alpha = 0.2$ (see table 5.5). This is mainly because $\alpha = 0.2$ leads to more looser schedules and more diverse search on the decision tree. Even it is possible to search more nodes with $\alpha = 0.2$, since the search on the decision tree is done by entering to the less promising nodes, it needs more time to finish the search. However the search is limited with the iteration number. So with

$\alpha = 0.6$, even the search on the decision tree is more narrow, since it always searches the most promising nodes it can still give good results if the iteration number is small. However if the iteration number was not limited and if it was possible to search for an infinitely many time, the GRASP algorithm with $\alpha = 0.2$ would probably lead to a better solution. This fact can be viewed at table 5.5. Note that G1, G2, G4, and G5 algorithms correspond to $\alpha = 0.2$ whereas G3, G4, G7, and G8 algorithms correspond to α=0.6. The *t values* of the pairs G1-G3, G2-G4, G5-G7, and G6-G8 are -26.08, -25.99, -22.0, and -22.00, respectively with the significance of $p \leq 0.000$ (see table C.11).

- We can also see that the second phase improves the overall result of the GRASP (see table 5.5). Note that G1, G2, G3 and G4 are the GRASP algorithms that do not have phase two whereas for the remaining GRASP algorithms phase two is allowed. The *t values* for the objective function value of the pairs G1-G5, G2-G6, G3-G7, and G4-G8 are -14.43, -14.07, -7.14 and –7.46, respectively with the significance of $p \leq 0.000$ (see table C.11). However again the trade off is the computational time requirements (see table 5.5). The *t values* for the computational times of the pairs G1-G5, G2-G6, G3-G7, and G4-G8 are -21.83, -24.20, -27.16 and –27.26, respectively with the significance of $p \leq 0.000$ (see table C.11).

- On the average of 320 experiments, G8 has a computational time of 49891 milliseconds, whereas B7 requires 61353 milliseconds (see table 5.5). The main factors that affect the computational time of the GRASP are the iteration number and the allowance of the second phase. Also from table 5.5 we can conclude that, if the second phase is allowed then higher α leads to a higher computational time (see table 5.5). This is mainly because it is more likely to apply the second phase if α is higher since the initial solution will likely to be more promising. Note that the corresponding pairs are G5-G7 and G6-G8 with the *t values* of -19.48 and -21.06, respectively, and the significance are $p \leq 0.000$. For the cases without phase two, the corresponding pairs are G1-G3 and G2-G4. The difference between the computational times of G1-G3 is not significant with $p \leq 0.122$. However the difference between the computational times of G2-G4 is significant with $p \leq 0.000$ and corresponding *t value* is -9.81. So the affect of α to the computational time is not always significant for the cases where the phase two is not allowed. Note that different α levels lead to different schedules which may increase or decrease the number of iterations, so the computational time. However for the case where the phase two is allowed the major effort is spend in the phase two, so the level of α becomes more significant on the required computational times.

5.3.5 SIMULATED ANNEALING

- Simulated annealing algorithms do not perform as good as the beam search algorithms for the overall experiments. S100 has the best overall average of unscheduled weight percentages among the other simulated annealing algorithms. The corresponding value of S100 is 0.194 (see table 5.5). On the average, it still improves NN algorithms objective value by 5.1% (0.245-0.194). The corresponding *t value* of the pair S100-NN is 20.25 with $p \leq 0.000$. The highest improvement on the objective function value of NN algorithm achieved by S100 is at the experimental combination of 0-0-0-0-1 (see tables A.1 and A.3). At this experimental combination the unscheduled weight percentages of the algorithms on the average of 10 corresponding replications are 0.144 for S100 and 0.230 for NN (see tables A.1 and A.3). So the improvement is 8.6% (0.230-0.144). This experimental combination corresponds to 76 observations, low oversubscription rate, high reconfiguration time case, no due dates for the observations and for the weight assignment to the observations of 1-5-9 (see table 5.1).

- Again simulated annealing algorithms perform relatively better for the high reconfiguration cases than the low reconfiguration cases (see table 5.6). The average of unscheduled weight percentages of 160 experiments that correspond to high reconfiguration case is 0.203 (see table

5.6). Whereas the beam search algorithm that performs best for same case is the B7 algorithm with the corresponding value of 0.202. The difference between these two values is only 0.1% and not significant with significance of $p \leq 0.358$ (see table C.13). When we compare the 16 different experimental combinations where there is high reconfiguration time case we see that at 7 experimental combination S100 performs better then B7 and perform same at 1 combination with respect to the objective function value. The reason behind this is same with the reason described in section 5.3. That is to say, the neighbor generation mechanism used at simulated annealing also depends on the concept of family scheduling.

- Note that the mutation concept improves the efficiency of the algorithms. Sno which has no mutation has the overall average weight of unscheduled percentages of 0.208 (see table 5.5). This objective function value increases as the mutation percentage increases. It reaches to the best value when mutation percentage is 100 (S100) with the corresponding value of 0.194. So the mutation percentage of 100 improves the no mutation case by 1.4% (0.208-0.194) on the overall average, since SA can search more nodes of the decision tree by the help of the mutation. The corresponding *t value* of the pair Sno-S100 is -11.98 with $p \leq 0.000$. However the trade off is again the required computational times. The corresponding *t value* of the pair Sno-S100 is -16.74 with $p \leq 0.000$.

- When we compare the computational times of simulated annealing algorithms with the other algorithms we can see that simulated annealing algorithms have computational times that are significantly higher than the other algorithms (see table C.9). The simulated annealing algorithm that has the smallest computational time on the overall average is Sno with the computational time of 100376 milliseconds, whereas S100 has the highest computational time on the average of 320 experiments which is 333074 milliseconds (see table 5.5).

	high	setup	low	setup
Algo	wt per.	comp. t.	wt. per.	comp. t
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

Table 5.6: Average unscheduled weight percentages and computational times of the algorithms for high and low reconfiguration time cases

5.4 SUMMARY

From the above discussion, we can conclude that in general beam search algorithms give the best objective function values on the overall average. 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 contain good solutions. However there is a danger of local entrapment at this methodology. For the high reconfiguration times case beam search algorithms cannot break the local entrapment. On the other hand, GRASP algorithms can break the local entrapment by the help of the **random search** methodology. Hence, for the high reconfiguration times case GRASP algorithms give better objective function value than the beam search algorithms on the average of corresponding 160 experiments. Now we will present the time versus scientific return graphs of these algorithms. The scientific return corresponds to average of scheduled weight percentages.

scientific return = 1- average of unscheduled weight percentages

Note that in figure 5.1, the scientific return is presented as the average of scheduled weight percentages of 320 experiments. In figure 5.2, the scientific return is presented as the average of scheduled weight percentages of 160 experiments that correspond to high reconfiguration time case. Finally, in figure 5.3, the scientific return is presented as the average of scheduled weight percentages of 160 experiments that correspond to low reconfiguration time case. The computational times are presented in milliseconds and time axis is in logarithmic scale.

The lines in the figures correspond to the pareto curves. Note that, only for the algorithms on the pareto curves, there is no other algorithm which performs better both in objective function value and computational time aspects.

To sum up, we can divide the algorithms into two groups. The first group consists of the algorithms that require low computational times although they

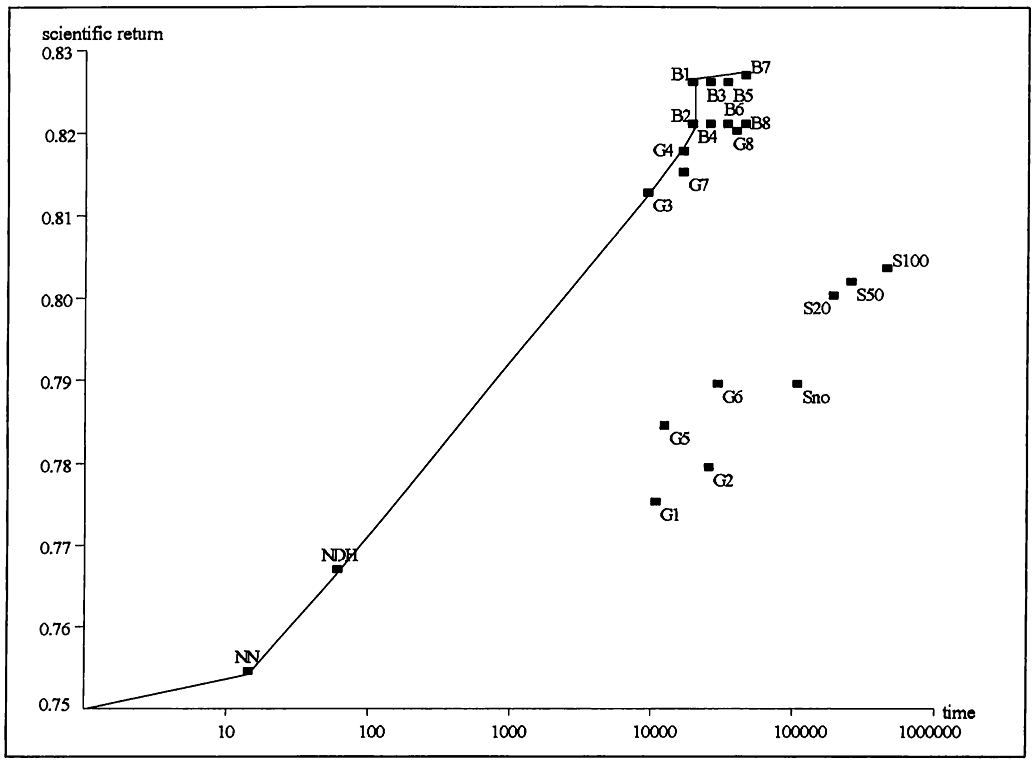

Figure 5.1: Time versus the scientific return for each algorithm

do not give good objective function value, and the second group consists of the algorithms that give better objective function value but require higher computational times. From the figures we can conclude that the simple dispatch heuristics NN and NDH belong to the first group whereas the local search heuristics such as beam search, GRASP and simulated annealing algorithms, belong to the second group. So if the time is limited for scheduling it is more preferrable to use NDH since it gives better objective function values than the NN algorithm and requires a very small computational time with respect 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, although it is more preferable to use the G8 algorithm if the problem domain has a high reconfiguration time between the equipments.

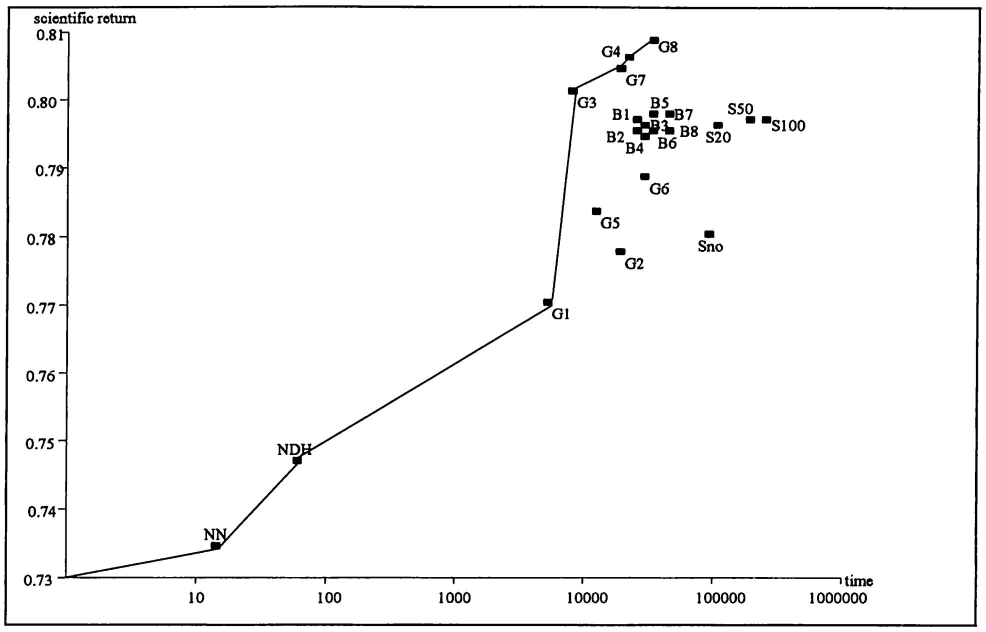

Figure 5.2: Time versus the scientific return for the high reconfiguration times case

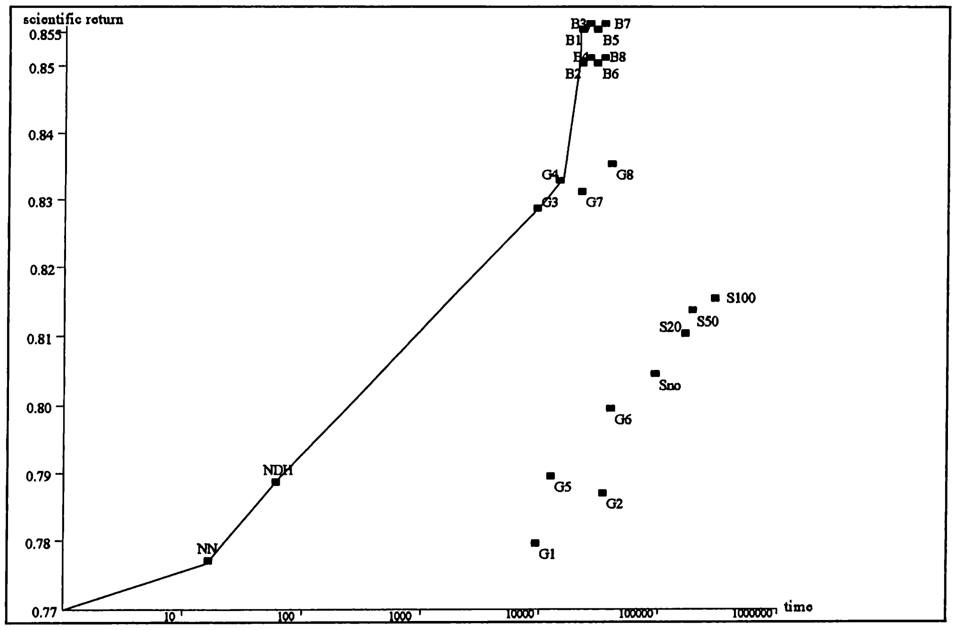

Figure 5.3: Time versus the scientific return for the low reconfiguration times case

Chapter 6

CONCLUSION

In this chapter we will provide a brief summary of the contributions done in this thesis and address of some of the future research directions. In this thesis we have studied space mission scheduling problem and concentrated mainly on the short term scheduling of Hubble Space Telescope (HST). We proposed new solution methodologies to generate good short term schedules of the candidate observations. The solution procedures that are proposed are a new dispatch heuristic, a beam search algorithm, a GRASP algorithm and a simulated annealing algorithm. In the next section, we will make a short summary of the contributions we have made to the solution of this problem.

6.1 CONTRIBUTIONS

First of all, we have proposed a new dispatch based heuristic which provides better objective function value than the nearest neighbor heuristic that is proposed by Smith and Pathak [57]. Secondly, we applied more sophisticated local search algorithms that can identify the complex interactions between the candidate observations better than the simple dispatch based heuristics. By this way we have improved the objective function values significantly.

We have considered some important features of the problem domain that are not considered by the algorithms that are available in the literature. These features are the priorities that are assigned to the candidate observations by Space Telescope Science Institute and the user specified due dates. Both of these constraints are important and realistic constraints that should be considered while generating short term schedule of HST.

We have presented the steps of each proposed algorithm and test the performance of the proposed algorithms on 320 randomly generated problems. In the proposed experimental design we have tested the influence of five experimental factors that can possibly affect the performance of the algorithms. These experimental factors are the number of observations, over subscription rate, reconfiguration times between the observations, due dates of the observations and weight assignment procedures to the observations. We designed a 2^5 full factorial design with 32 different experimental condition with 10 replications generated for each combination.

From the experimental design we have concluded that the beam search algorithm, with the parameter settings of b=6, f=10 and c=3, gives the best objective function score on the average of 320 experiments. Beam search algorithm with the specified parameters improves the nearest neighbor algorithm 7.2% on the average. We have also concluded that the GRASP algorithm, with the parameter setting of $\alpha = 0.6$, number of iterations=500 and with phase two, provides the best objective function value on the average of 160 experiments that correspond to high reconfiguration time case.

We have also provide some modifications to the generic algorithms. For example we have introduced a new concept of childwidth that restricts the number of beams that generates from a particular beam. From the experimental design we have concluded that this restriction improves the objective function values of the beam search algorithms 0.33% on the average of 320 experiments.

Another fact that we have introduced is the mutation concept that we have proposed for the simulated annealing algorithm. By the help of the mutation concept simulated annealing algorithm can search more nodes of the decision tree. We have concluded that the simulated annealing algorithm with the mutation percentage of 100 improved the simulated annealing algorithm with no mutation 1.3% on the average of 320 experiments.

As a result of this thesis we have found that more sophisticated algorithms can improve the objective function values for the space mission scheduling problems such as HST scheduling problem. However one important thing to consider is the higher computational times of the sophisticated algorithms. So some clever modifications must be done that will reduce the search on the nodes of the decision tree. We have provided this at the GRASP algorithm by not allowing the phase 2 to the less promising schedules generated at phase 1 and restricted the number of iterations of the local search algorithms. For the simulated annealing algorithms we have provided the same fact by creating the desirable exchange list which consist of most promising exchanges in order to generate the next neighbor.

6.2 FUTURE RESEARCH DIRECTIONS

At the end there are several future research directions emanating from this research study as such:

- Some other sophisticated algorithms such as tabu search and genetic algorithms can be applied to the problem, and the performance of these algorithms can be analyzed.
- In this research we relaxed the complex power up/down sequences needed to reconfigure the instruments of HST and used the maximum time required to reconfigure the instruments. More sophisticated computerized algorithms can be generated that consider the fact of power up/down sequences while scheduling the candidate observations.

- The proposed new dispatch 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. Some other dispatch rules can be used at these algorithms and the performance of these new dispatch rules can be analyzed. Moreover the affect of different dispatch rules to the performance of these algorithms can be analyzed.

- Finally the new concepts that we have introduced to the generic algorithms of beam search, GRASP and simulated annealing, can be implemented to other research areas and the performance of these concepts can be analyzed.

Bibliography

[1] M. Aarup, M. Arentoft, Y. Parrod, I. Stokes, H. Vadon, and J. Stader. Optimum aiv: A knowledge-based planning and scheduling system for spacecraft aiv. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 16, pages 451–469. Morgan Kaufmann, 1994.

[2] F. Anger, C. Lee, and L. Martin-Vega. Single machine scheduling with tight windows. Technical Report 16, Department of Industrial and Systems Engineering, University of Florida, 1986.

[3] B. Atkinson. A greedy look-ahead heuristic for combinatorial optimization: an application to vehicle scheduling with time windows. *Journal of Operational Research Society*, 45:673–684, 1994.

[4] N. Balakrishnan. Simple heuristics for the vehicle routing problem with soft time windows. *Journal of Operational Research Society*, 44:279–287, 1993.

[5] N. Collins, R. Eglese, and B. Golden. Simulated annealing -an annotated bibliography. *American Journal of Mathematics and Management Science*, 8:209–308, 1988.

[6] D. T. Connoly. An improved annealing scheme for the quadratic assignment problem. *European Journal of Operational Research*, 46:93–100, 1990.

[7] M. Deale, M. Yvanovich, D. Schnitzius, D. Kautz, M. Carpenter, M. Zweben, G. Davis, and B. Daun. The space shuttle ground processing scheduling system. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 15, pages 423–449. Morgan Kaufmann, 1994.

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

[9] J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis. Time constrained routing and scheduling. Technical Report G-92-42, GERAD, 1994.

[10] R. Diekmann, R. Luling, and J. Simon. Problem independent distributed simulated annealing and its applications. In *Applied Simulated Annealing*, chapter 1, pages 17–44. Springer-Verlag, 1993.

[11] H. Fargher and R. Smith. Planning in a flexible semiconductor manufacturing environment. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 19, pages 545–580. Morgan Kaufmann, 1994.

[12] T. A. Feo and M. G. Resende. Greedy randomized adaptive search procedure. *Journal of Global Optimization*, 6:109–133, 1995.

[13] T. A. Feo, K. Sarathy, and J. McGahan. A grasp for single machine scheduling with sequence dependent setup costs and linear delay penalties. *Computers and Operations Research*, 23:881–895, 1996.

[14] M. Fisher and R. Jaikumar. An algorithm for the space shuttle problem. *Operations Research*, 26:166–182, 1978.

[15] M. Fox. Isis: A retrospective. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 1, pages 3–28. Morgan Kaufmann, 1994.

[16] V. Gabrel. Scheduling jobs within time windows on identical parallel machines: New model and algorithms. *European Journal of Operational Research*, 83:320–329, 1995.

[17] M. Gendreau, G. Laporte, and M. Solomon. Single vehicle routing and scheduling to minimize the number of delays. *Transportation Science*, 29:56–62, 1995.

[18] J. Ghosh. Computational aspects of the maximum diversity problem. *Operations Research Letters*, 19:175–181, 1996.

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

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

[21] H. Ishibuchi, S. Misaki, and H. Tanaka. Modified simulated annealing algorithms for the flow-shop sequencing problem. *European Journal of Operational Research*, 81:388–398, 1995.

[22] T. Isobe, M. Johnston, E. Morgan, and G. Clark. The application of spike to astro-d mission planning. *Proc. 2nd Astron. Data Analysis Software and Systems*, pages 340–344, 1993.

[23] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; part 1, graph partioning. *Operations Research*, 37:865–892, 1989.

[24] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; part 2, graph coloring and number partitioning. *Operations Research*, 39:378–405, 1991.

[25] M. D. Johnston and H. Adorf. Scheduling with neural networks - the case of the hubble space telescope. *Computers and Operations Research*, 19:209–240, 1992.

[26] M. D. Johnston and G. E. Miller. Spike: Intelligent scheduling of hubble space telescope observations. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 14, pages 391–422. Morgan Kaufmann, 1994.

[27] M. D. Johnston and S. Minton. Analyzing a heuristic strategy for constraint satisfaction and scheduling. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 9, pages 257–289. Morgan Kaufmann, 1994.

[28] S. Jones. Heuristics to schedule service engineers within time windows. *Journal of Operational Research Society*, 46:339–346, 1995.

[29] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[30] H. Kise, T. Ibraki, and H. Mine. A solvable case of one machine scheduling problem with ready and due times. *Operations Research*, 26:121–126, 1978.

[31] A. Kolen, A. R. Kan, and H. Trienekens. Vehicle routing with time windows. *Operations Research*, 35:266–273, 1987.

[32] P. Kouvelis and W.Chiang. A simulated annealing procedure for single row layout problems in flexible manufacturing systems. *International Journal of Production Research*, 30:716–732, 1992.

[33] F. Kramer and C. Lee. Common due window scheduling. Technical Report 5, Department of Industrial and Systems Engineering, University of Florida, 1992.

[34] M. Laguna and J. Gonzalez-Velarde. A search heuristic for just-in-time scheduling in parallel machines. *Journal of Intelligent Manufacturing*, 2:253-260, 1991.

[35] A. Lann and G. Mosheiov. Single machine scheduling to minimize the number of early and tardy jobs. *Computers and Operations Research*, 23:769–781, 1996.

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

[37] E. Lawler and J. Moore. A functional equation and its applications to resource allocation and sequencing problems. *Management Science*, 16:77–84, 1969.

[38] S. D. Liman and S. Ramaswamy. Earliness-tardiness scheduling problems with a common delivery window. *Operations Research Letters*, 15:195–203, 1994.

[39] B. Lowerre. *The HARPY Speech Recognition System.* PhD thesis, Carnegie Mellon University, 1976.

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

[41] S. Minton and A. B. Philips. Applying a heuristic repair method to the hubble space telescope scheduling problem. *Innovative Approaches to Planning, Scheduling and Control, Proceedings of a workshop held at San Diego CA*, pages 215–219, 1990.

[42] C. Monma and C. Potts. On the complexity of scheduling with batch setup times. *Operations Research*, 37:798–804, 1989.

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

[44] E. Morgan. Evaluation of spike for xte. Technical report, Center for Space Research, MIT, 1992.

[45] N. Muscettola. Hsts: Integrating planning and scheduling. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 6, pages 169–212. Morgan Kaufmann, 1994.

[46] N. Muscettola, S. F. Smith, G. Amiri, and D. Pathak. Generating space telescope observation schedules. Technical Report CMU-RI-TR-89-28, Carnegie Mellon University, 1989.

[47] N. Muscettola, S. F. Smith, A. Cesta, and D. D'Aloisi. Cordinating space telescope operations in an integrated planning and scheduling architecture. *IEEE Control Systems*, 12:28–37, 1992.

[48] E. Novicki and S. Zdrzalka. Single machine scheduling with major and minor setup times: A tabu search approach. *Journal of the Operational Research Society*, 47:1054–1064, 1996.

[49] I. Osman and C. Potts. Simulated annealing for permutation flow-shop scheduling. *OMEGA*, 17:551–557, 1989.

[50] P. Ow and S. Smith. Viewing scheduling as an opportunistic problem solving process. *Annals of Operation Research*, 12:85–108, 1988.

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

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

[53] J. Potvin and J. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66:331–340, 1993.

[54] R. Rachamadugu and T. Morton. Myopic heuristics for the single machine weighted tardiness problem. Technical Report 28-81-82, GSIA, Carnegie Mellon University, 1982.

[55] I. Sabuncuoglu and M. Bayiz. A beam search based algorithm for the job shop scheduling problem. Technical report, Bilkent University, 1997.

[56] I. Sabuncuoglu and S. Karabuk. A beam search algorithm and evaluation of scheduling approaches for flexible manufacturing systems's. *to appear in IIE Transactions.*

[57] S. F. Smith and D. K. Pathak. Balancing antagonistic time and resource utilization constraints in over-subscribed scheduling problems. Technical Report CMU-RI-TR-91-05, Carnegie Mellon University, 1991.

[58] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraint. *Operations Research*, 35:254–265, 1987.

[59] STScI. Proposal instructions for the hubble space telescope. Technical report, STScI, 1986.

[60] R. V. Vidal. *Applied Simulated Annealing.* Springer-verlag, 1993.

[61] F. Villarreal and R. Bulfin. Scheduling a single machine to minimize the weighted number of tardy jobs. *IIE Transactions*, 15:337–343, 1983.

[62] M. Waldrop. Will the hubble space telescope compute? *Science*, 243:1497, 1989.

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

[64] M. Zweben, B. Daun, E. Davis, and M. Deale. Scheduling and rescheduling with iterative repair. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 8, pages 241–255. Morgan Kaufmann, 1994.

Appendix A

Unscheduled Weight Percentages

Experiment	NN	NDH
0 0 0 0 0	0,227664	0,213766
0 0 0 0 1	0,230074	0,217588
0 0 0 1 0	0,236267	0,207809
0 0 0 1 1	0,239107	0,205632
0 0 1 0 0	0,121774	0,105228
0 0 1 0 1	0,12221	0,106801
0 0 1 1 0	0,131039	0,119126
0 0 1 1 1	0,131509	0,110255
0 1 0 0 0	0,297816	0,296492
0 10 0 1	0,301275	0,287991
0 1 0 1 0	0,307743	0,293183
0 1 0 1 1	0,312434	0,291977
0 1 1 0 0	0,264064	0,258769
0 1 1 0 1	0,263815	0,25983
0 1 1 1 0	0,272005	0,258769
0 1 1 1 1	0,271785	0,244687
1 0 0 0 0	0,238012	0,224499
1 0 0 0 1	0,234195	0,23402
1 0 0 1 0	0,248038	0,238448
1 0 0 1 1	0,243975	0,239434
1 0 1 0 0	0,213601	0,206626
1 0 1 0 1	0,210793	0,204331
1 0 1 1 0	0,224063	0,223191
1 0 1 1 1	0,221795	0,202934
1 1 0 0 0	0,283784	0,275501
1 10 0 1	0,278205	0,27052
1 1 0 1 0	0,29381	0,281604
1 1 0 1 1	0,287985	0,28554
1 1 1 0 0	0,279425	0,272014
1 1 1 0 1	0,274886	0,259343
1 1 1 1 0	0,29163	0,288143
1 1 1 1 1	0,288159	0,267552
Average	0,245092	0,232863

Table A.1: Unscheduled weight percentages of NN and ATC

Experiment	B1	B2	B3	B4	B5	B6	B7	B8
0 0 0 0 0	0,157512	0,160821	0,160159	0,162144	0,158835	0,160821	0,15685	0,160159
0 0 0 0 1	0,146121	0,149575	0,14745	0,152763	0,14745	0,151966	0,14745	0,153826
0 0 0 1 0	0,155526	0,159497	0,160159	0,160821	0,15685	0,159497	0,158835	0,160821
0 0 0 1 1	0,150106	0,151435	0,14745	0,1517	0,151435	0,151435	0,146387	0,150638
0 0 1 0 0	0,067505	0,068829	0,063534	0,06949	0,067505	0,068829	0,06552	0,068829
0 0 1 0 1	0,049681	0,055792	0,05101	0,05526	0,053932	0,052604	0,051807	0,057386
0 0 1 1 0	0,070152	0,06949	0,070152	0,072799	0,06949	0,070814	0,06949	0,072138
0 0 1 1 1	0,055792	0,053932	0,050478	0,051275	0,053666	0,054463	0,049681	0,053135
0 1 0 0 0	0,238253	0,242886	0,2409	0,242886	0,238253	0,242886	0,238253	0,240238
0 10 0 1	0,231934	0,231934	0,242561	0,241764	0,229012	0,2322	0,235919	0,234591
0 1 0 1 0	0,244209	0,250165	0,242224	0,245533	0,244209	0,250165	0,243547	0,244209
0 1 0 1 1	0,230074	0,231934	0,231403	0,235654	0,230074	0,231934	0,22848	0,230606
0 1 1 0 0	0,188617	0,18994	0,186631	0,198544	0,188617	0,191926	0,188617	0,19722
0 1 1 0 1	0,158608	0,164453	0,159139	0,162859	0,158342	0,159671	0,157545	0,160733
0 1 1 1 0	0,195897	0,197882	0,192588	0,196559	0,195897	0,197882	0,190602	0,194573
0 1 1 1 1	0,166047	0,170032	0,161796	0,166844	0,165515	0,172157	0,160733	0,169766
1 0 0 0 0	0,18483	0,190497	0,191369	0,195728	0,18483	0,193548	0,191369	0,197472
1 0 0 0 1	0,183549	0,185819	0,183723	0,191233	0,183549	0,185819	0,183723	0,191233
1 0 0 1 0	0,192241	0,193984	0,192241	0,191805	0,192677	0,195292	0,191369	0,193984
1 0 0 1 1	0,187216	0,189836	0,186518	0,189661	0,187216	0,189836	0,186518	0,189661
1 0 1 0 0	0,146033	0,1517	0,146905	0,147777	0,146905	0,152136	0,146469	0,149956
1 0 1 0 1	0,130283	0,132728	0,133601	0,136221	0,130632	0,132553	0,130981	0,131156
1 0 1 1 0	0,144725	0,14952	0,147341	0,152572	0,147341	0,152136	0,147341	0,154752
1 0 1 1 1	0,12941	0,131156	0,123123	0,124869	0,130283	0,130458	0,124345	0,126965
1 1 0 0 0	0,239756	0,241064	0,236269	0,236704	0,23932	0,241064	0,236704	0,238448
1 10 0 1	0,222669	0,223891	0,22424	0,222669	0,222669	0,223891	0,221795	0,224066
1 1 0 1 0	0,242807	0,243243	0,241064	0,2415	0,242371	0,242371	0,242371	0,242807
1 1 0 1 1	0,228432	0,230527	0,230353	0,230178	0,228432	0,230527	0,227908	0,231575
1 1 1 0 0	0,214473	0,216652	0,214908	0,216652	0,214037	0,21578	0,213601	0,215344
1 1 1 0 1	0,19263	0,205903	0,191233	0,201711	0,196822	0,207649	0,195424	0,206427
1 1 1 1 0	0,21578	0,21796	0,220139	0,224935	0,217088	0,21796	0,218396	0,219268
1 1 1 1 1	0,194377	0,202235	0,190534	0,19979	0,1949	0,202934	0,195599	0,203283
Average	0,173601	0,176729	0,173787	0,177216	0,174005	0,176975	0,173238	0,17704

Table A.2: Unscheduled Weight Percentages of Beam Search

Experiment	G1	G2	G3	G4	G5	G6	G7	G8
0 0 0 0 0	0,19325	0,180013	0,150232	0,143614	0,165453	0,173395	0,148908	0,140966
0 0 0 0 1	0,176939	0,171095	0,138948	0,138151	0,164187	0,15356	0,133369	0,137354
0 0 0 1 0	0,190602	0,19325	0,154864	0,146923	0,175381	0,178028	0,154864	0,143614
0 0 0 1 1	0,183316	0,178533	0,148247	0,142136	0,16153	0,166578	0,142136	0,13762
0 0 1 0 0	0,155526	0,140966	0,092654	0,088021	0,140966	0,134348	0,090668	0,08405
0 0 1 0 1	0,131775	0,123007	0,073858	0,073061	0,121413	0,111849	0,071998	0,070935
0 0 1 1 0	0,158173	0,154864	0,097287	0,094639	0,141628	0,129054	0,091992	0,090007
0 0 1 1 1	0,128852	0,131509	0,083422	0,073061	0,121148	0,113974	0,080499	0,072795
0 1 0 0 0	0,27002	0,252813	0,23362	0,227664	0,260093	0,236929	0,232296	0,223031
0 10 0 1	0,243092	0,24017	0,225027	0,217588	0,238045	0,236982	0,222901	0,216525
0 1 0 1 0	0,266711	0,264064	0,238253	0,230973	0,256784	0,25546	0,236929	0,229649
0 1 0 1 1	0,257173	0,25186	0,231668	0,225824	0,244952	0,240967	0,227152	0,223698
0 1 1 0 0	0,266049	0,264725	0,213104	0,209133	0,256122	0,249504	0,210457	0,208471
0 1 1 0 1	0,248406	0,237779	0,190755	0,18305	0,231668	0,225558	0,189426	0,182253
0 1 1 1 0	0,276638	0,265387	0,219722	0,209795	0,269358	0,257445	0,21906	0,209795
0 1 1 1 1	0,260361	0,246281	0,195005	0,194474	0,24017	0,236185	0,193677	0,192614
1 0 0 0 0	0,217524	0,202703	0,186138	0,178727	0,208806	0,197036	0,185266	0,178727
1 0 0 0 1	0,200314	0,198917	0,175864	0,174991	0,191408	0,188439	0,171498	0,173943
1 0 0 1 0	0,227114	0,21796	0,192241	0,187446	0,214037	0,202267	0,189625	0,183522
1 0 0 1 1	0,210094	0,206078	0,18774	0,177436	0,202235	0,196123	0,184946	0,17569
1 0 1 0 0	0,218396	0,211421	0,172188	0,166521	0,210985	0,197908	0,171316	0,166085
1 0 1 0 1	0,199267	0,193154	0,154384	0,148446	0,189312	0,183723	0,15124	0,148271
1 0 1 1 0	0,226242	0,218832	0,175676	0,176983	0,214908	0,21796	0,17524	0,17524
1 0 1 1 1	0,203458	0,202061	0,154558	0,150541	0,196472	0,195599	0,154034	0,150367
1 1 0 0 0	0,258936	0,257629	0,228858	0,225806	0,251526	0,249782	0,226678	0,224499
1 10 0 1	0,24869	0,237513	0,226336	0,218652	0,23891	0,227384	0,225114	0,216556
1 1 0 1 0	0,265911	0,261116	0,240192	0,237576	0,258936	0,248038	0,237576	0,234525
1 1 0 1 1	0,253929	0,256374	0,231051	0,228082	0,2438	0,245721	0,226685	0,226336
1 1 1 0 0	0,275501	0,265039	0,235833	0,22973	0,265039	0,259372	0,234961	0,22973
1 1 1 0 1	0,256898	0,254104	0,216731	0,212714	0,252008	0,240482	0,216032	0,211317
1 1 1 1 0	0,279861	0,275937	0,239756	0,23932	0,278553	0,271142	0,238012	0,236704
1 1 1 1 1	0,254453	0,257073	0,220573	0,218128	0,248865	0,252532	0,220224	0,21708
Average	0,225109	0,219132	0,185149	0,180288	0,214209	0,208541	0,182962	0,178499

Table A.3: Unscheduled Weight Percentages of GRASP

Experiment	Sno	S20	S50	S100
0 0 0 0 0	0,180013	0,151555	0,152879	0,154203
0 0 0 0 1	0,160999	0,141339	0,141073	0,14373
0 0 0 1 0	0,18266	0,158835	0,15685	0,152879
0 0 0 1 1	0,172157	0,144527	0,150106	0,146387
0 0 1 0 0	0,116479	0,103905	0,101257	0,099934
0 0 1 0 1	0,103613	0,091658	0,087673	0,089532
0 0 1 1 0	0,126406	0,107214	0,107214	0,101919
0 0 1 1 1	0,113974	0,103613	0,0983	0,095377
0 1 0 0 0	0,250165	0,231635	0,236267	0,236267
0 10 0 1	0,240967	0,23034	0,224495	0,223964
0 1 0 1 0	0,256784	0,239576	0,238253	0,238915
0 1 0 1 1	0,240436	0,234591	0,227152	0,229809
0 1 1 0 0	0,244209	0,231635	0,228326	0,236267
0 1 1 0 1	0,217322	0,213603	0,213603	0,214134
0 1 1 1 0	0,250827	0,238915	0,236267	0,235606
0 1 1 1 1	0,23034	0,22423	0,218385	0,217322
1 0 0 0 0	0,207062	0,195292	0,191805	0,19442
1 0 0 0 1	0,205903	0,183374	0,180929	0,182501
1 0 0 1 0	0,212293	0,198344	0,203575	0,202703
1 0 0 1 1	0,206776	0,200664	0,189487	0,195948
1 0 1 0 0	0,183522	0,177855	0,180035	0,180035
1 0 1 0 1	0,178309	0,175864	0,172721	0,169752
1 0 1 1 0	0,192241	0,189625	0,189625	0,188753
1 0 1 1 1	0,18285	0,180929	0,177436	0,173245
1 1 0 0 0	0,258065	0,246731	0,245859	0,2415
1 10 0 1	0,240133	0,232448	0,230527	0,224939
1 1 0 1 0	0,26286	0,247167	0,248474	0,248038
1 1 0 1 1	0,247992	0,235417	0,236989	0,233846
1 1 1 0 0	0,248474	0,242807	0,244115	0,244987
1 1 1 0 1	0,238212	0,238037	0,23891	0,236815
1 1 1 1 0	0,257193	0,251962	0,253705	0,251962
1 1 1 1 1	0,248865	0,244848	0,241705	0,240657
Average	0,208066	0,196517	0,195125	0,194573

Table A.4: Unscheduled Weight Percentages of SA

NN	NDH
weight=1-2-3	weight=1-2-3
0,245670914	0,235198111
weight=1-5-9	weight=1-5-9
0,244512714	0,23052721
duedateper=0	duedateper=0
0,240099522	0,23083256
duedateper=5	duedateper=5
0,250084106	0,23489276
highsetup	highsetup
0,266274019	0,254000358
lowsetup	lowsetup
0,22390961	0,211724962
not osc.	not osc.
0,204632314	0,191230588
osc	osc
0,285551314	0,274494733
noofjobs=76	noofjobs=76
0,233161397	0,217369032
noofjobs=115	noofjobs=115
0,257022231	0,248356289

Table A.5: Unscheduled Weight Percentages of NN and ATC at different experimental conditions

B1	B2	B3	B4	B5	B6	B7	B8
wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3
0,18114	0,18400	0,18166	0,18477	0,18151	0,18456	0,18120	0,18438
wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9
0,16605	0,16944	0,16591	0,16965	0,16649	0,16938	0,16526	0,16969
ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0
0,17202	0,17578	0,17335	0,17715	0,17254	0,17583	0,17262	0,17669
ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5
0,17517	0,17767	0,17422	0,17728	0,17546	0,17811	0,17385	0,17738
highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup
0,20220	0,20481	0,20363	0,20579	0,20232	0,20520	0,20234	0,20527
lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup
0,14500	0,14863	0,14394	0,14863	0,14568	0,14874	0,14413	0,14880
not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.
0,13441	0,13716	0,13470	0,13788	0,13516	0,13763	0,13425	0,13825
osc	osc	osc	osc	osc	osc	osc	osc
0,21278	0,21629	0,21287	0,21654	0,21284	0,21631	0,21221	0,21582
# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76
0,15662	0,15928	0,15672	0,16043	0,15681	0,15932	0,15560	0,15930
# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115
0,19057	0,19416	0,19084	0,19400	0,19119	0,19462	0,19086	0,19477

Table A.6: Unscheduled Weight Percentages of Beam Search at different experimental conditions

G1	G2	G3	G4	G5	G6	G7	G8
wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3
0,23415	0,22666	0,19191	0,18705	0,22303	0,21610	0,19024	0,18491
wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9
0,21606	0,21159	0,17838	0,17352	0,20538	0,20097	0,17568	0,17208
ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0
0,22253	0,21444	0,18215	0,17724	0,21162	0,20414	0,18013	0,17579
ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5
0,22768	0,22382	0,18814	0,18333	0,21679	0,21294	0,18579	0,18120
highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup
0,22897	0,22313	0,19932	0,19384	0,21725	0,21229	0,19662	0,19164
lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup
0,22124	0,21513	0,17096	0,16672	0,21116	0,20478	0,16930	0,16535
not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.
0,18880	0,18277	0,14614	0,14129	0,17624	0,17124	0,14360	0,13932
osc	osc	osc	osc	osc	osc	osc	osc
0,26141	0,25549	0,22415	0,21928	0,25217	0,24584	0,22232	0,21767
# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76
0,21293	0,20601	0,16791	0,16238	0,19930	0,19373	0,16539	0,16021
# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115
0,23728	0,23224	0,20238	0,19819	0,22911	0,22334	0,20052	0,19678

Table A.7: Unscheduled Weight Percentages of GRASP at different experimental conditions

SAno	SA20	SA50	SA100
weight=1-2-3	weight=1-2-3	weight=1-2-3	weight=1-2-3
0,21432834	0,20081571	0,20090661	0,200524186
weight=1-5-9	weight=1-5-9	weight=1-5-9	weight=1-5-9
0,20180301	0,19221765	0,18934316	0,188622297
duedateper=0	duedateper=0	duedateper=0	duedateper=0
0,20459049	0,19300488	0,19190464	0,1920612
duedateper=5	duedateper=5	duedateper=5	duedateper=5
0,21154086	0,20002848	0,19834513	0,197085283
highsetup	highsetup	highsetup	highsetup
0,22032901	0,20448968	0,20342001	0,203127963
lowsetup	lowsetup	lowsetup	lowsetup
0,19580235	0,18854368	0,18682976	0,18601852
not osc.	not osc.	not osc.	not osc.
0,17032872	0,15653708	0,15506024	0,154457385
osc	osc	osc	osc
0,24580264	0,23649628	0,23518952	0,234689098
noofjobs=76	noofjobs=76	noofjobs=76	noofjobs=76
0,19295959	0,17794809	0,17613121	0,176015279
noofjobs=115	noofjobs=115	noofjobs=115	noofjobs=115
0,22317177	0,21508527	0,21411855	0,213131204

Table A.8: Unscheduled Weight Percentages of SA at different experimental conditions

Appendix B

Computational Times

Experiment	NN	NDH
0 0 0 0 0	8,2	31,8
0 0 0 0 1	10	31,5
0 0 0 1 0	9,9	31,4
0 0 0 1 1	9,9	31,4
0 0 1 0 0	11,8	36,4
0 0 1 0 1	10	36,5
0 0 1 1 0	11,6	33,1
0 0 1 1 1	10	38,5
0 1 0 0 0	10	29,6
0 10 0 1	10	28,4
0 1 0 1 0	8,4	33,5
0 1 0 1 1	8,4	31,8
0 1 1 0 0	6,6	30,1
0 1 1 0 1	11,6	29,8
0 1 1 1 0	10	31,7
0 1 1 1 1	10	30,3
1 0 0 0 0	23,4	73,2
1 0 0 0 1	23,4	71,7
1 0 0 1 0	21,7	76,4
1 0 0 1 1	21,6	69,9
1 0 1 0 0	21,7	75,3
1 0 1 0 1	23,4	73,5
1 0 1 1 0	21,7	75,1
1 0 1 1 1	25,1	68,2
1 1 0 0 0	21,7	66,7
1 10 0 1	23,2	66,7
1 1 0 1 0	21,8	65,2
1 1 0 1 1	18,3	70,1
1 1 1 0 0	21,8	68,4
1 1 1 0 1	21,7	66,7
1 1 1 1 0	21,7	69,9
1 1 1 1 1	20	71,8
Average	15,89375	51,39375

Table B.1: Computational Times of NN and ATC

Experiment	B1	B2	B3	B4	B5	B6	B7	B8
0 0 0 0 0	9556,6	9606,8	11786,7	11804,9	14591,6	14111,7	17391,8	17549,8
0 0 0 0 1	9471,6	9491,9	11714,8	11661,6	14481,9	14001,5	17191,7	17240
0 0 0 1 0	9506,5	9516,6	11753,4	11733,5	14559,9	14036,5	17328,4	17385,2
0 0 0 1 1	9403,1	9408,4	11723,4	11663,3	14380,1	13871,7	17206,8	17263,6
0 0 1 0 0	10596,7	10635,3	13223,4	13166,5	16216,8	15660	19398,3	19456,7
0 0 1 0 1	10441,8	10395	12956,7	12918,2	15923,4	15448,4	19028,4	19179,9
0 0 1 1 0	10519,8	10579,7	13039,8	13046,9	16116,6	15573,3	19200	19313,3
0 0 1 1 1	10350,1	10351,7	12856,6	12866,7	15770,1	15216,6	18868,3	19003,5
0 1 0 0 0	8325,1	8338,3	10378,3	10336,5	12711,5	12293,4	15199,7	15351,7
0 10 0 1	8131,7	8185,2	10093,4	10016,8	12490,1	12055,3	14771,6	14850
0 1 0 1 0	8201,6	8208,5	10276,5	10201,6	12538,4	12091,6	15024,9	15143
0 1 0 1 1	8065,3	8098,5	10050,1	10063,5	12366,8	11921,7	14797	14880,1
0 1 1 0 0	8801,8	8786,6	10998,2	10869,9	13436,5	12996,8	16076,8	16183,3
0 1 1 0 1	8500,3	8496,4	10599,8	10536,6	13035,2	12680,2	15526,7	15738,4
0 1 1 1 0	8698,3	8726,6	10851,8	10849,9	13243,1	12855	15918,3	16038,4
0 1 1 1 1	8434,8	8456,5	10473,3	10468,5	12953,4	12478,2	15354,9	15510
1 0 0 0 0	31755	31590,1	39051,7	39096,7	47549,9	47600	58673,3	58629,9
1 0 0 0 1	31304,9	31173,4	38808,3	38713,4	47005,1	47043,4	58201,7	58238,1
1 0 0 1 0	31306,8	31335,1	38650,2	38871,7	46980,1	47023,3	58096,8	58363,3
1 0 0 1 1	31059,9	31035,1	38458,5	38521,9	46430	46745	57681,9	58041,7
1 0 1 0 0	33448,4	33240	41351,8	41418,4	50070	50009,9	62120,2	62111,6
1 0 1 0 1	32526,7	32430	39980	39893,3	48649,9	48729,9	60340,1	60401,8
1 0 1 1 0	33125,1	32911,7	40951,6	40874,8	49339,9	49405	61353,3	61393,2
1 0 1 1 1	32396,6	32276,6	40023,2	40093,3	48443,3	48668,3	59940,2	60250,3
1 1 0 0 0	29390	29390	36486,6	36561,7	44133,3	44275,1	54796,8	54936,7
1 10 0 1	29248,3	29354,9	35989,9	36046,7	43923,4	44071,7	54053,2	54516,8
1 1 0 1 0	28851,5	28926,6	35791,8	35956,6	43303,3	43800,2	53698,5	54055
1 1 0 1 1	28768,5	28819,9	35298,4	35473,4	43141,7	43326,7	53218,3	53640,1
1 1 1 0 0	30001,8	29898,4	36963,4	36976,6	44909,9	45068,4	55601,8	55836,7
1 1 1 0 1	28963,2	28576,6	35751,7	35463,5	43288,4	43303,3	53723,3	53578,4
1 1 1 1 0	29561,9	29660	36471,7	36395	44271,7	44494,8	54750,1	54968,3
1 1 1 1 1	28698,3	28564,9	35175,2	34968,3	43065,1	42958,2	52720,1	53086,7
Average	19919,13	19889,54	24624,38	24610,32	29978,76	29806,72	36789,16	36941,73

Table B.2: Computational Times of Beam Search

Experiment	G1	G2	G3	G4	G5	G6	G7	G8
0 0 0 0 0	5429,9	10785,1	5423,3	10716,9	7054,9	13006,7	11573,5	19958,2
0 0 0 0 1	5431,4	10773,2	5419,9	10723,3	7131,8	12606,5	12750	21715,2
0 0 0 1 0	5423,2	10771,7	5433,3	10753,5	7103,5	12366,9	11176,6	18266,8
0 0 0 1 1	5429,9	10766,8	5429,7	10731,5	7383,2	12973,3	9271,8	18168,4
0 0 1 0 0	5446,8	10776,7	5411,7	10708,3	7359,5	13916,8	14693,3	20850
0 0 1 0 1	5428,5	10793,3	5411,6	10715	7434	13450,1	13386,6	23880
0 0 1 1 0	5444,8	10793,4	5428,5	10708,4	7121,5	13628,4	13094,9	21273,5
0 0 1 1 1	5453,4	10781,6	5420	10693,6	7865,2	13851,8	14980,1	23566,4
0 1 0 0 0	5458,5	10791,6	5445	10733,1	7725	13205,1	10756,7	17876,6
0 10 0 1	5440	10755,1	5420	10704,9	7616,6	13451,5	9538,4	17796,8
0 1 0 1 0	5434,8	10768,2	5439,9	10731,5	7276,5	12955,1	10749,9	16305,1
0 1 0 1 1	5431,7	10796,4	5423,1	10701,4	6772,1	12826,6	9323,3	17181,6
0 1 1 0 0	5433,3	10760	5408,6	10718,4	6712	13903,5	11000	19201,6
0 1 1 0 1	5443,5	10751,4	5406,3	10711,6	7801,5	13470,1	13616,7	23530,1
0 1 1 1 0	5428,4	10776,7	5438,2	10714,9	6747,1	14106,7	13406,7	17523,5
0 1 1 1 1	5449,9	10784,9	5414,8	10701,6	8148,2	14585,1	11709,9	21181,8
1 0 0 0 0	12193,3	24445,1	12242,1	24653,6	16276,7	28325	25060	40640
1 0 0 0 1	12195,1	24473	12264,9	24673,4	15500,1	28340	24240,2	43498,4
1 0 0 1 0	12225	24468,1	12270,2	24770	15395,2	28370	22800,2	39248,3
1 0 0 1 1	12205,1	24469,9	12261,7	24729,9	16276,8	29636,7	22293,5	42221,8
1 0 1 0 0	12224,7	24468,2	12223,3	24636,7	19303,3	33011,6	28430,1	48136,7
1 0 1 0 1	12224,8	24478,5	12230,2	24659,9	19873,5	34515	33946,9	52058,3
1 0 1 1 0	12233,3	24446,8	12239,9	24691,6	20166,6	33471,4	25298,3	50386,8
1 0 1 1 1	12208,5	24500	12256,5	24708,4	19021,6	34960,1	29763,3	49531,5
1 1 0 0 0	12201,7	24461,7	12271,8	24751,6	16050	30601,7	24291,7	41555,1
1 10 0 1	12198,3	24475,1	12161,7	24755,1	16224,7	29628,4	25396,8	43128,3
1 1 0 1 0	12211,8	24463,2	12210,1	24760,1	15608,3	29586,6	23138,3	45693,2
1 1 0 1 1	12234,8	24468,4	12179,9	24755	15935,1	32126,8	19883,5	42946,7
1 1 1 0 0	12249,8	24501,7	12145,1	24713,4	19735,1	33079,8	30236,6	49891,7
1 1 1 0 1	12203,3	24461,7	12166,4	24738,5	20591,5	31673,2	30135,2	49054,9
1 1 1 1 0	12223,4	24424,9	12241,5	24740	18865	33053,3	28063,6	44583,3
1 1 1 1 1	12216,6	24490,1	12123,3	24718,2	20323,4	34194,9	26216,6	43528,3
Average	8826,797	17622,58	8820,703	17716,35	12574,98	22464,96	19069,48	32636,84

Table B.3: Computational Times of GRASP

Experiment	Sno	S20	S50	S100
0 0 0 0 0	33198,3	79645,1	76334,9	107501,8
0 0 0 0 1	35496,8	90438,4	81281,6	133303,5
0 0 0 1 0	28690,3	67303,3	92893,5	116431,7
0 0 0 1 1	32839,9	95151,7	88748,3	130135
0 0 1 0 0	30331,7	96553,2	81675	123571,7
0 0 1 0 1	91866,5	121535,1	150911,5	180909,7
0 0 1 1 0	40506,8	87088,4	74116,7	118283,4
0 0 1 1 1	101379,9	105986,5	111731,5	178529,9
0 1 0 0 0	36016,7	88453,4	111353,1	126253,5
0 10 0 1	90463,3	112231,6	97521,7	169053
0 1 0 1 0	36163,3	99823,7	94018,4	143700,1
0 1 0 1 1	38945	108784,8	137501,6	160731,7
0 1 1 0 0	50215	113218,1	113149,9	134961,6
0 1 1 0 1	46016,6	157148,5	228045	217626,5
0 1 1 1 0	36718,7	116011,6	136421,8	177340
0 1 1 1 1	62031,6	184248,3	183383,4	251198,5
1 0 0 0 0	129050,1	131115,2	226593,3	318398,2
1 0 0 0 1	124680	153365,2	277088,1	455298,3
1 0 0 1 0	131060,1	119035,1	233838,3	307615,1
1 0 0 1 1	98641,8	181603,6	268238,4	426900,2
1 0 1 0 0	117503,2	216156,7	291361,7	480359,9
1 0 1 0 1	245216,6	218456,8	328178,3	608301,6
1 0 1 1 0	131046,9	288034,9	300755,1	451008,4
1 0 1 1 1	140306,7	228681,7	335398,6	615146,7
1 1 0 0 0	94528,6	276976,6	241513,5	444719,9
1 10 0 1	131103,1	186976,7	316781,7	473681,6
1 1 0 1 0	140511,6	200671,7	221783,4	436888,3
1 1 0 1 1	222173,3	204304,9	234818,5	553954,9
1 1 1 0 0	99036,6	249343,4	348063,3	565931,9
1 1 1 0 1	141768,3	326189,8	352099,9	699810,1
1 1 1 1 0	189305,2	284356,7	390555,1	595446,7
1 1 1 1 1	285239,9	294658,3	376943,2	755383,4
Average	100376,64	165110,91	206346,82	333074,28

Table B.4: Computational Times of SA

NN	NDH
weight=1-2-3	weight=1-2-3
15,75	51,7375
weight=1-5-9	weight=1-5-9
16,0375	51,05
duedateper=0	duedateper=0
16,15625	51,01875
duedateper=5	duedateper=5
15,63125	51,76875
highsetup	highsetup
15,61875	50,58125
lowsetup	lowsetup
16,16875	52,20625
not osc.	not osc.
16,4625	53,36875
osc	osc
15,325	49,41875
noofjobs=76	noofjobs=76
9,775	32,2375
noofjobs=115	noofjobs=115
22,0125	70,55

Table B.5: Computational Times of NN and ATC at different experimental conditions

B1	B2	B3	B4	B5	B6	B7	B8
wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3
20102	20084	24876	24885	30248	30080	37164	37294
wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	weight=1-5-9
19735	19694	24372	24335	29709	29532	36414	36588
ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0
20028	19974	24758	24717	30151	29959	37005	37112
ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5
19809	19804	24490	24503	29806	29654	36572	36770
highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup
19521	19529	24144	24170	29411	29266	36083	36255
lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup
20316	20249	25104	25050	30545	30346	37495	37628
not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.
21048	20998	26020	26021	31656	31446	38876	38988
osc	osc	osc	osc	osc	osc	osc	osc
18790	18780	23228	23199	28300	28166	34702	34894
# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76
9187	9205	11423	11387	14050	13580	16767	16880
# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115
30650	30573	37825	37832	45906	46032	56810	57003

Table B.6: Computational Times of Beam Search at different experimental conditions

G1	G2	G3	G4	G5	G6	G7	G8
wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3	wt=1-2-3
8828	17618	8829	17718	12406	22286	18985	31961
wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9	wt=1-5-9
8824	17626	8811	17713	12743	22643	19153	33311
ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0	ddper=0
8825	17621	8815	17707	12649	22261	19940	33298
ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5	ddper=5
8828	17623	8825	17725	12500	22668	18198	31975
highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup	highsetup
8821	17620	8831	17727	11583	21250	17015	30387
lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup	lowsetup
8832	17624	8810	17704	13566	23679	21123	34886
not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.	not osc.
8824	17624	8835	17704	12516	22276	19547	33337
osc	osc	osc	osc	osc	osc	osc	osc
8828	17620	8805	17728	12633	22653	18591	31936
# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76	# job=76
5438	10776	5423	10716	7328	13394	11939	19892
# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115	# job=115
12215	24468	12218	24715	17821	31535	26199	45381

Table B.7: Computational Times of GRASP at different experimental conditions

SAno	SA20	SA50	SA100
weight=1-2-3	weight=1-2-3	weight=1-2-3	weight=1-2-3
82742,6938	157111,694	189651,688	290525,763
weight=1-5-9	weight=1-5-9	weight=1-5-9	weight=1-5-9
118010,581	173110,119	223041,956	375622,788
duedateper=0	duedateper=0	duedateper=0	duedateper=0
93530,7125	163612,738	207622,031	327480,175
duedateper=5	duedateper=5	duedateper=5	duedateper=5
107222,563	166609,075	205071,613	338668,375
highsetup	highsetup	highsetup	highsetup
87722,6375	137242,563	175019,269	281535,425
lowsetup	lowsetup	lowsetup	lowsetup
113030,638	192979,25	237674,375	384613,125
not osc.	not osc.	not osc.	not osc.
94488,475	142509,431	188696,55	296980,944
osc	osc	osc	osc
106264,8	187712,381	223997,094	369167,606
noofjobs=76	noofjobs=76	noofjobs=76	noofjobs=76
49430,025	107726,356	116192,994	154345,725
noofjobs=115	noofjobs=115	noofjobs=115	noofjobs=115
151323,25	222495,456	296500,65	511802,825

Table B.8: Computational Times of SA at different experimental conditions

Appendix C

Statistical Analysis

	B1	B2	B3	B4	B5	B6	B7	B8
B2	.9997	1						
B3	.9996	.9994	1					
B4	.9994	.9994	.9997	1				
B5	.9998	.9996	.9996	.9994	1			
B6	.9995	.9997	.9994	.9995	.9996	1		
B7	.9995	.9993	.9999	.9998	.9995	.9994	1	
B8	.9994	.9994	.9997	.9999	.9994	.9996	.9997	1

Table C.1: Correlation table of the beam search algorithms

	G1	G2	G3	G4	G5	G6	G7	G8
G2	.9992	1						
G3	.9984	.9986	1					
G4	.9985	.9987	.9996	1				
G5	.9995	.9990	.9984	.9985	1			
G6	.9996	.9995	.9985	.9986	.9990	1		
G7	.9985	.9987	.9999	.9997	.9986	.9987	1	
G8	.9986	.9988	.9997	.9999	.9986	.9987	.9997	1

Table C.2: Correlation table of the GRASP algorithms

	Sno	S20	S50	S100
S20	.9995	1		
S50	.9996	.9996	1	
S100	.9989	.9990	.9989	1

Table C.3: Correlation table of the simulated annealing algorithms

	Objec.	Value	Comp.	Time
Factors	F	p	F	p
F1	1992.300	.000	173.762	.000
F2	135.705	.000	1.501	.221
F3	31.503	.000	.351	.554
F4	2.493	.115	.320	.572
F5	9917.099	.000	.096	.757
F1-F2	1.941	.165	.132	.716
F1-F3	10.641	.001	.104	.747
F1-F4	.200	.655	.320	.572
F1-F5	383.069	.000	.022	.882
F2-F3	9.643	.002	.142	.706
F2-F4	.003	.954	.262	.609
F2-F5	24.284	.000	.022	.882
F3-F4	.003	.954	.589	.443
F3-F5	5.858	.016	.123	.727
F4-F5	.485	.487	.610	.435

Table C.4: F Values and Significance Levels (p) for ANOVA results of NN

	Objec.	Value	Comp.	Time
Factors	F	p	F	p
F1	1636.856	.000	2452.338	.000
F2	115.051	.000	26.067	.000
F3	30.525	.000	4.412	.000
F4	.311	.578	.940	.332
F5	8628.476	.000	.790	.375
F1-F2	3.411	.066	1.003	.317
F1-F3	5.112	.025	.418	.519
F1-F4	.833	.362	.067	.796
F1-F5	310.668	.000	.971	.325
F2-F3	6.708	.010	1.036	.310
F2-F4	.029	.865	3.697	.056
F2-F5	18.436	.000	.940	.333
F3-F4	.006	.941	.439	.508
F3-F5	8.992	.003	.017	.897
F4-F5	.015	.903	.038	.846

Table C.5: F Values and Significance Levels (p) for ANOVA results of NDH

	Objec.	Value	Comp.	Time
Factors	F	p	F	p
F1	2153.243	.000	36563.976	.000
F2	121.777	.000	380.739	.000
F3	69.772	.000	42.807	.000
F4	.001	.970	2.649	.105
F5	11826.231	.000	11.322	.001
F1-F2	.549	.460	35.895	.000
F1-F3	1.162	.282	0.018	.895
F1-F4	.002	.966	1.049	.307
F1-F5	406.554	.000	2.969	.086
F2-F3	6.434	.012	19.518	.000
F2-F4	.112	.738	.312	.577
F2-F5	20.908	.000	.402	.527
F3-F4	.003	.957	.013	.909
F3-F5	18.033	.000	2.993	.085
F4-F5	.018	.893	.157	.692

Table C.6: F Values and Significance Levels (p) for ANOVA results of B8 algorithm

	Objec.	Value	Comp.	Time
Factors	F	p	F	p
F1	2316.025	.000	1630676.8	.000
F2	135.003	.000	30.621	.000
F3	19.331	.000	15.089	.000
F4	.902	.343	3.475	.063
F5	12619.757	.000	11.011	.001
F1-F2	1.056	.305	35.452	.000
F1-F3	2.284	.132	2.770	.097
F1-F4	.009	.924	.002	.967
F1-F5	439.662	.000	1.866	.173
F2-F3	3.429	.065	.971	.325
F2-F4	.010	.919	.618	.433
F2-F5	24.289	.000	14.744	.000
F3-F4	.089	.765	3.545	.061
F3-F5	7.162	.008	.621	.431
F4-F5	.180	.672	1.455	.229

Table C.7: F Values and Significance Levels (p) for ANOVA results of G3 algorithm

	Objec.	Value	Comp.	Time
Factors	F	p	F	p
F1	2305.786	.000	459.830	.000
F2	144.112	.000	18.753	.000
F3	5.759	.017	38.237	.000
F4	.891	.346	.450	.503
F5	12756.531	.000	26.060	.000
F1-F2	2.049	.153	4.577	.033
F1-F3	2.318	.129	15.754	.000
F1-F4	.268	.605	.002	.962
F1-F5	438.978	.000	5.311	.022
F2-F3	6.827	.009	.231	.631
F2-F4	.007	.932	.318	.252
F2-F5	23.920	.000	.034	.854
F3-F4	.062	.0803	.096	.757
F3-F5	1.229	.269	1.806	.180
F4-F5	.243	.622	.111	.739

Table C.8: F Values and Significance Levels (p) for ANOVA results of S100 algorithm

	Objec. Value		Comp. Time	
Pairs	t value	p	t value	p
NN-NDH	-5.88	.000	-37.73	.000
NN-B7	-23.31	.000	-32.47	.000
NN-G8	-23.87	.000	-38.86	.000
NN-S100	-20.25	.000	-24.30	.000
NDH-B7	-21.51	.000	-32.46	.000
NDH-G8	-20.99	.000	-38.85	.000
NDH-S100	-16.05	.000	-24.30	.000
B7-G8	4.29	.000	7.00	.000
B7-S100	12.06	.000	-22.90	.000
G8-S100	14.40	.000	-22.78	.000

Table C.9: Pairwise comparison of the proposed algorithms

	Objec.	Value	Comp. Time	
Pairs	t value	p	t value	p
B1-B2	6.17	.000	2.14	.034
B3-B4	5.95	.000	.88	.380
B5-B6	5.89	.000	6.91	.000
B7-B8	6.69	.000	-6.38	.000
B1-B5	1.22	.223	-34.19	.000
B2-B6	.62	.537	-31.62	.000
B3-B7	-.105	.295	-31.52	.000
B4-B8	.21	.835	-31.87	.000
B1-B3	.02	.988	-33.38	.000
B2-B4	.26	.795	-32.41	.000
B5-B7	-1.29	.199	-29.26	.000
B6-B8	.26	.811	-32.64	.000

Table C.10: Pairwise comparison of the beam search algorithms

	Objec. Value		Comp. Time	
Pairs	t value	p	t value	p
G1-G5	-14.43	.000	-21.83	.000
G2-G6	-14.07	.000	-24.20	.000
G3-G7	-7.14	.000	-27.16	.000
G4-G8	-7.46	.000	-27.26	.000
G1-G2	-5.82	.000	-45.43	.000
G3-G4	-7.69	.000	-44.10	.000
G5-G6	-5.23	.000	-35.76	.000
G7-G8	-7.28	.000	-27.0	.000
G1-G3	-26.08	.000	1.55	.122
G2-G4	-25.99	.000	-9.81	.000
G5-G7	-22.00	.000	-19.48	.000
G6-G8	-22.00	.000	-21.06	.000

Table C.11: Pairwise comparison of the GRASP algorithms

	Objec. Value		Comp. Time	
Pairs	t value	p	t value	p
Sno-S20	-10.49	.000	-6.53	.000
Sno-S50	-11.65	.000	-10.47	.000
Sno-S100	-11.98	.000	-16.74	.000
S20-S50	-3.25	.001	-5.99	.000
S20-S100	-3.92	.000	-14.79	.000
S50-S100	-1.17	.243	-13.23	.000

Table C.12: Pairwise comparison of the simulated annealing algorithms

	Objec. Value		Comp. Time	
Pairs	t value	p	t value	p
NN-NDH	-4.56	.000	-26.81	.000
NN-B7	-16.13	.000	-22.61	.000
NN-G8	-18.32	.000	-27.67	.000
NN-S100	-17.23	.000	-18.97	.000
NDH-B7	-14.03	.000	-22.60	.000
NDH-G8	-15.76	.000	-27.66	.000
NDH-S100	-14.52	.000	-18.96	.000
B7-G8	-5.92	.000	6.94	.000
B7-S100	.92	.358	-17.93	.000
G8-S100	7.30	.000	-17.76	.000

Table C.13: Pairwise comparison of the proposed algorithms for the high reconfiguration times case

Vitae

Kemal Kilic was born on June 26, 1973 in Lefkose, Cyprus. He studied secondary school at Tarsus Amerikan Lisesi and high school at Kayseri Fen Lisesi. He attended to the Department of Electical and Electronics Engineering, Bogazici University, in 1990 and graduated from the same department in July 1995. In October 1995, he joined to the Department of Industrial Engineering at Bilkent University as a research assistant. Until August 1997, he worked with Asst. Prof. Selim Akturk for his graduate study at the same department.