AN ABSTRACT OF THE DISSERTATION OF

Shakib Shakeri for the degree of Doctor of Philosophy in Industrial Engineering presented on September 20, 2002. Title: A Mathematical Modeling Framework for Scheduling and Managing Multiple Concurrent Tasks.

Redacted for privacy

Abstract approved: _____

Rasaratnam Logendran

Occurrence of *human error* in highly complex systems, such as a cockpit, can be disastrous and/or overwhelmingly costly. Mismanagement of multiple concurrent tasks has been observed by researchers to be a type of *repetitive* human error in previous studies of accidents and incidents. This error may occur in the form of wrong selection of a strategy to attend to tasks, and/or wrong assessment of a task's priority at each moment.

The desire to prevent such errors forms two essential questions: 1) Is there any (near) optimal method of managing multiple concurrent tasks? 2) How optimally do human operators manage these tasks? To answer the first question, *operations research* as it is applied to *single machine scheduling* was used. The operator was assumed to be a single resource that attended to different tasks, one at a time. To answer the second question, a software environment was developed to measure the human's multitasking performance, which was then compared with the answer to question one.

In this research, the operator's *quality of performance* was maximized as opposed to the *number of tasks accomplished*, which was considered by previous researchers. A metaphor of 'Juggler and spinning plates' along with a graphic bar illustration was used to resemble an operator (a juggler) who manages several tasks (plates on vertical poles) concurrently.

Several mixed (binary) integer-linear programming models were developed discretely over time. One model was selected and solved by the means of tabu search heuristic method. In tabu search, the significance of different initial solution finding mechanisms and different applications of long-term memory was investigated. A conjecturing method, within the tabu search, was introduced for solving problems with very large planning horizons. In all cases, tabu search gave good quality solutions in a much shorter time than branch-and-bound.

Under five different scenarios, ten subjects were studied while managing multiple concurrent tasks in the software environment. None of the subjects could gain a score better than tabu search in any of the scenarios. Subjects' patterns of attendance to tasks were analyzed and compared against the pattern suggested by tabu search, and similarities/differences were identified.

A Mathematical Modeling Framework for Scheduling and Managing Multiple Concurrent Tasks

by

Shakib Shakeri

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of

the requirements for the

degree of

Doctor of Philosophy

Presented September 20, 2002

Commencement June 2003

Doctor of Philosophy dissertation of Shakib Shakeri presented on September 20, 2002

APPROVED:

Redacted for privacy

Major Professor, representing Industrial Engineering

Redacted for privacy

Head of the Department of Industrial and Manufacturing Engineering

Redacted for privacy

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Redacted for privacy

Shakib Shakeri, Author

ACKNOWLEDGMENTS

This dissertation was the culmination of my journey through the Oregon State University's Ph.D. program. It is my great privilege to acknowledge the following individuals and organizations that assisted me along this path.

I wish to thank my Major professor, Dr. Rasaratnam Logendran, for his always-steady guidance, and the countless hours he patiently spent in meetings with me over the past few years. He challenged me from the time this dissertation was only an idea, and continued to support and correct me throughout the entire process. I am greatly appreciative of his enthusiasm, effort, knowledge, and dedication.

I must also express my sincere gratitude to my Minor professor, Dr. Kenneth Funk. His insight initiated this interdisciplinary dissertation, and his guidance significantly helped me with every part that related to *human factors*. I was very fortunate to work with him closely as a student, teaching assistant, and research assistant, which also made me a student of his people skills, leadership, and professionalism.

I would like to thank the other members of my Ph.D. committee: Dr. Ping-Hung Hsieh, Dr. Dean Jensen, and Dr. John Sessions, the Graduate Council Representative. I am also thankful to Dr. Ashok Chandrashekar and Dr. Eldon Olsen who served in the committee before their departure from Corvallis. The constructive criticisms and comments of the committee members are much appreciated.

I am grateful to the following people and institutions for their financial support without which my Ph.D. program would have faced severe difficulties: Bob Baker

and the Central Web Services; Kenneth Funk, the Federal Aviation Administration (FAA), Research Integrations, Inc., National Aeronautics and Space Administration (NASA), and the Industrial and Manufacturing Engineering (IME) department; Ashok Chandrashekar and the College of Business; John Shea and the IME department; Milne Computer Center and the Student Computing Facilities. And finally my parents, who had an always-open bank that kept offering 0% APR loans even before I asked, and without questioning my credit.

I would like to thank my friend, Jirachai Buddhakulsomsiri (Sim), and Dr. David Birkes from the Department of Statistics for their very helpful assistance with parts of my data analysis and experimental design. Also, the theses of Adulwit Sonthinen, Fenny Subur, and Dr. Kurt Colvin helped me in preparation of my dissertation. Many thanks to the OSU Writing Center, which corrected my English, which was then recorrected by my professors, which was finally, incorrectly revised by me again. I owe special thanks to all of my friends and colleagues at Oregon State University, whose names are not listed here, for their enthusiasm, friendship, companionship, and fun spirit without which my journey would have been a lonely and boring one.

As I grow older, I realize how significant my parents' influence has been in my life. To be specific, if they did not encourage me to continue my higher education, it would have been very unlikely for me to think of a Ph.D. To be vague, however, their work started as early as my time $t = 0$, and my deepest indebtedness will grow until time $t = T$, the end of my *planning horizon.*

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

Page

LIST OF FIGURES

Figure Page

LIST OF FIGURES (Continued)

Figure Page

LIST OF TABLES

Table Page

LIST OF TABLES (Continued)

Table Page

LIST OF TABLES (Continued)

Table Page

LIST OF TABLES (Continued)

Table Page

LIST OF APPENDICES

Appendix Page

LIST OF APPENDICES (Continued)

Appendix Page

LIST OF APPENDICES (Continued)

Appendix Page

LIST OF APPENDIX FIGURES

Appendix Figure — Page

LIST OF APPENDIX FIGURES (Continued)

Appendix Figure — Page

LIST OF APPENDIX FIGURES (Continued)

Appendix Figure — Page

LIST OF APPENDIX TABLES

A Mathematical Modeling Framework for Scheduling and Managing Multiple Concurrent Tasks

CHAPTER 1: INTRODUCTION

Rapid growth of technology and the development of complex systems is changing the role of the human operator from being a direct moment-to-moment controller of a system to one of a supervisor or a monitor of multiple—fully automated or semiautomated—tasks. One of the main objectives of *automation* in complex systems is removing *human error*[1] by eliminating the *human* from the process. In most cases, automation has reduced the number of faults in a system associated with manual repetitive tasks. But a fair number of disastrous faults are still caused by human error while supervising automated systems. Boeing statistical summary shows that between 1987 and 1996, over 65% of the Hull Loss Commercial Jet Aircraft Accidents--in the US and worldwide--were primarily caused by the cockpit crew, and not by any other factor such as maintenance, weather, or the airplane (Boeing, 1997). A typical pilot error can be attending to tasks that do not have the highest priority among the other cockpit tasks. For instance, a captain who was distracted with a low-priority task on China Airline caused an incident with substantial damage to the aircraft (Chou, 1991). A study by Chou et al. (1996) shows that mismanagement of cockpit tasks occurred in 23% of the 324 accidents, and 49% of the 470 incidents reviewed. Compared to accidents, incidents are less severe occurrences with regard to death, injury or damage. "Incidents ... affect or could affect the safety of operations" (Federal Aviation Regulations, 1994). Evidence shows that before an accident happens, it occurs a few times as an incident but is overlooked (Chou et al., 1996).

---

[1] For more details on human error refer to the textbook by Reason (1990).

The main challenge for a pilot in managing cockpit tasks is the multitasking nature of it. A pilot has to repeatedly perform different tasks within the four broad categories of Aviation, Navigation, Communication, and System Management. For instance, detecting/fixing a unit's malfunction, communicating with the air traffic controller (ATC), keeping the proper altitude, and changing the heading are all relatively independent tasks, which have the potential to occur simultaneously. The more of these tasks that occur at the same time, the harder it is for a pilot to manage them, and the more likely it is to miss one. All the tasks in a multitasking environment are competing for a limited resource—the pilot's attention. There are several theories and models developed for representing a human's attention allocation including multitasking capability (Funk, 1998). Despite the differences between these models, all of them have agreed on the assumption that humans have a limited capacity for processing multitasks (Moray et al., 1991).

A multitask supervisory control scenario for a human operator is not limited to a pilot in the cockpit. In many other complex systems such as nuclear power plants, computer based industrial systems, a military tank crew, a commander in charge of several units, or a corporate manager, a common pattern of the monitor/decision maker responsible for several concurrent tasks is found. Hence, a normative model—standard of comparison—that provides an optimal or near-optimal algorithm for a human's attention allocation among multiple concurrent tasks, against which to compare and contrast actual human performance, will make a significant contribution to a variety of domains. Such comparisons can provide guidance to help human operators in complex, high-risk systems allocate attention more effectively.

CHAPTER 2: LITERATURE REVIEW

Theories and models of multitasking can be divided into psychological and engineering theories (Pew and Mavor, 1998). Engineering theories are focused on describing human behaviors, whereas psychological theories are more influenced by the mechanisms underlying behaviors. Pattipati and Kleinman (1991) summarize some of the work done in modeling a human's attention allocation. *Operations Research* as a known subject of combinatorial optimization is a valid tool for this purpose. Carbonell et al. (1968) for the first time applied *queuing theory* to model the visual scanning of an operator monitoring several tasks by looking at different displays. In their model, a human's visual attention resource was considered as a single server and the displays awaiting the server's attention were regarded as customers in a queue. Other researchers (Walden and Rouse, 1978; Chu and Rouse, 1979; Greenstein and Rouse, 1982) expanded this idea by assigning a service time probability distribution to the human attention resource as a server.

Tulga and Sheridan (1980) defined a paradigm with several queues, each having a group of similar tasks arriving randomly with random rewards, time requirements, and deadlines. The objective was to maximize the aggregate reward earned by working on different tasks. The decision made at each point in time was the solution of a deterministic combinatorial optimization problem (dynamic programming with branch and bound strategy). Pattipati et al. (1983) argue that one of the disadvantages of queuing theory approach is its difficulty to find an optimal sequencing strategy. Moreover, queuing theoretic emphasizes the stationary parameters of a system. These parameters such as mean waiting time (waiting time for a display to be read), or mean service time (task processing time) will help designers have a better understanding of the multitasking environment. However, the general interest/issue is in the moment-to-moment decisions made by the

human operator rather than parameters such as mean waiting time. Combinatorial approaches, on the other hand, have a deterministic view and cannot easily handle randomness associated with the nature of a multitasking system as well as queuing theory can. The main shortcoming of the queuing approach is that the operator has to follow a rule (First in-First served, or Last in-First served) and does not have the freedom to attend to the most urgent task.

Moray et al. (1991) used a different subject within operations research to model a human operator in charge of multiple concurrent tasks. They proposed a normative model based on *scheduling theory.* The optimal strategic decisions made by the model were then compared with those made by a human operator. Moray et al. (1991) considered the human as a *single machine* and cognitive/physical tasks that he needed to accomplish as *jobs*. For example, bringing a vehicle back to the centerline of the roadway is a *job.* The criterion in their model was to maximize the operator's *performance*. In their study, *performance* was measured by the number of tasks completed by their due date (a different interpretation of performance is described in Section 4.1). They assigned no *partial credit* to partially completed tasks (i.e., only fully completed tasks were valued). This means that their model could suggest ignoring a task due to not having a chance to be completed fully by its deadline. Nor was the interruption of tasks associated with any cost in their study.

Engineering theories are not limited to operations research. Other approaches used, in the manual control mode rather than the supervisory control mode explained above, have been based on *estimation* and *optimal control* theory (Pattipati and Kleinman, 1991). Estimation theory is concerned with predicting future states or detecting changes in parameters of a dynamic process, while control theories focus on *how* a human can achieve a desired state, in a manual control mode, optimally (Rouse, 1980). The emphasis of estimation and control theories in human-machine

interaction is different from the focus of this research. Consider the example where a driver should be within the appropriate lane and also maintain a certain speed. Control theory minimizes the series of actions that one takes in terms of choosing the appropriate steering wheel (or accelerator pedal) angle to achieve the desired state, while minimizing the deviation of the outcomes of these actions from the desired state. The emphasis of this research, however, is on how the operator's high-level attention is allocated to different tasks. In the driver example, whether a driver at any point in time should attend to the task of 'being in the appropriate lane,' or 'keeping the proper speed,' is what is addressed. It is assumed that as soon as the operator attends to a task (e.g., staying in the appropriate lane), the task status starts improving with a certain speed (optimally or not) towards the desired state. It is further assumed that tasks are independent and the driver can pay attention to at most one task at a time. The latter assumption is a reasonable assumption for the operator's monitoring, information processing and action selection skills as opposed to sensory-motor skills. Other researchers also have used the single resource assumption for the human's attention (Tulga and Sheridan, 1980; Moray et al., 1991; Greenstein and Rouse, 1982).

CHAPTER 3: PROBLEM STATEMENT

The objective of this research is to develop a normative model—as a standard of comparison—for a human operator to attend to the most appropriate task at each point in time, considering the consequences of choosing this task in a fixed time span. This model will be used with a simple experimental apparatus that allows the researcher to manipulate different variables of a multitasking environment and compare the (optimal) behavior of the model with that of the human operator.

Such normative models must be capable of measuring the performance of the decision-making process. The measure of performance, which is widely considered in previous research as maximizing the number of tasks accomplished (Chapter 2), does not seem to apply successfully to all multi-tasking environments. In a true multi-tasking environment, tasks do not simply leave the system after accomplishment, but they stay in the system and their status varies repeatedly between poor and desired status depending on how much attention they receive (Chapter 4). Unlike prior studies, the focus of the normative model developed in this research would be maximizing the operator's quality of performance in terms of how well the tasks are performed instead of the number of tasks accomplished. Such a normative model would also indicate exactly what task has to be attended at a specific point in time. Clearly, the focus here is different from the queuing theoretic approach where the emphasis is on estimating the stationary parameters of the system.

The approach that will be used in this research for developing the normative model is through mathematical modeling in *Operations Research*, as it is applied to *Scheduling*. However, before defining the parameters of the mathematical model, there is a need to somehow distinguish, define and quantify the elements of a multi-tasking environment in order to capture that environment to the best extent

possible. This quantification should address decision-making between tasks, criteria to evaluate the quality of the decision-maker's performance over time for each task and for all tasks that the decision-maker is responsible for as a whole, and the difference between tasks' characteristics and worth. Further, such quantifications should be applicable to a wide-variety of multi-tasking environments. For this purpose, a metaphor will be introduced to picture a model of the real-life multi-tasking environment, and within that metaphor some parameters essential to the description of a multi-tasking environment will be defined (Chapter 4).

After modeling a generic multi-tasking environment mathematically (Chapter 5), there is typically a need for solving that model with efficient heuristic algorithms in a timely fashion. Tabu search algorithm is the heuristic structure that will be used for this purpose (Chapter 6-7). On the other hand, an environment is needed to allow the researcher to measure and analyze human operator's decision-making performance. This environment will be developed in computer software (Chapter 8). Finally, the results obtained from the human operator's performance will be compared against the results obtained from solving the mathematical model. Similarities and differences of this comparison will be pointed out to guide operators about their strengths and weaknesses (Chapter 8).

In summary, the objectives of this research are as follows:

1. To define and quantify the elements and behavior of a generic multi-tasking environment—Chapter 4.
2. To develop a mathematical model with the use of the parameters and variables defined in (1) that aims at finding the best decision-making performance within the limitations of a multi-tasking environment—Chapter 5.
3. To develop an efficient heuristic algorithm that would solve the model proposed in (2)—Chapter 6-7.

4. To develop a task management environment that would measure and analyze the human operator's decision-making performance—Chapter 8.
5. To compare and contrast the human operator's performance identified in (4) with that of the mathematical model's found in (3) —Chapter 8.

CHAPTER 4: METHODOLOGY

In the following sections, first the definition of performance in previous studies (number of accomplishments) is reviewed and its shortcomings are addressed; second, a new definition of performance based on 'quality of performing the tasks' is proposed to improve its applicability to generic multi-tasking environments. Based on this new definition, a metaphor of 'juggler and spinning plates' is suggested in order to help understand and quantify the parameters and elements of a multi-tasking environment. Finally, some scenarios are discussed to show the applicability of this metaphor and advantages of looking at a multi-tasking environment from this perspective.

4.1 QUALITY OF PERFORMANCE VS NUMBER OF ACCOMPLISHMENTS

The aforementioned studies in Chapter 2 have always considered the operator either as a server who accomplishes different tasks waiting in the queue, or as a scheduler who schedules attention to tasks. The primary goal has been *accomplishing* as many tasks as possible to maximize the aggregated value. If the operator is considered as a server who is serving a queue of customers, there is a starting point and ending point for him to service each customer (service time). The service to a customer cannot be interrupted; otherwise the customer would not be satisfied. Also if the operator is perceived as a scheduler, he/she is looking for maximizing the overall value of the tasks accomplished. All of these studies were measuring the performance by the number of tasks accomplished. Evidently, these studies were also looking for maximizing the performance.

In this research, performance is defined in terms of *quality of performing* tasks instead of number of tasks accomplished. In generic multitasking environments, the tasks are required to be controlled continuously and usually exist all of the time.

Tasks cannot be assumed to have been *fully* served to depart the queue or system, and be counted as a task accomplished. The service that tasks receive is usually a temporary service with variable service time that can help them stay closer to their desired operating status. It is assumed that all tasks have a tendency to deviate from their desired status if not attended to. Therefore, the operator's implicit objective would be to minimize this cumulative deviation among different tasks that are not considered finished in a multitasking environment. The accumulation of these deviations is undesirable and too much of it in highly complex multi-tasking environments may lead to disastrous results. Chou (1991) quotes from the National Transportation Safety Board (NTSB) report of an accident,

> It is obvious that this accident, as well as others, was not the final consequence of a single error, but was the cumulative result of several minor deviations from normal operating procedure which triggered a sequence of events with disastrous results.

As indicated earlier, there are many multitasking environments in which no task is considered *finished.* For example, monitoring the speed while driving a car is an ongoing task. At no point in time, can the driver claim that keeping the proper speed is no longer a concern because he/she has adjusted the vehicle's speed (i.e., completed the task) once earlier on during the ride. Most likely, the driver deviates from the proper speed during the ride, but only when he/she deviates too much and too often from the perfect speed status, do we call him/her a poor driver with a low performance. This applies to other tasks during driving such as latitudinal control, fuel management, engine temperature monitoring, or even tuning the radio. Typically, an incident occurs when the operator deviates from the desired status in too many tasks to be able to correct them, or allows one critical task to deviate too much. As a result, those critical tasks that deviate beyond the correctable range trigger the accident. Thus, maximizing the performance in this study means maximizing the quality of performing the tasks.

4.2 JUGGLER AND SPINNING PLATES!

In order to better understand the environment discussed in this research, the metaphor of *juggler and spinning plates* is used. A simplistic view of any multitasking environment can be compared to a juggler who has several plates—each on a vertical stick—and tries to keep them spinning. At each point in time, the juggler has to decide as to which plate to attend. The juggler has to assure that no plate falls down and all the plates spin smoothly and do not wobble. Figure 4.1 shows this process in which for demonstration purposes it is assumed that all the plates (tasks) start at their 100% (perfect) *satisfaction level* (SL). i.e., the plates are spinning as smoothly as possible. This perfect SL can correspond to the task of controlling the speed in driving when the driver has just adjusted the vehicle's speed to comply with the posted speed. At each point in time a plate deviates from its perfect SL (and starts wobbling) by the fixed rate of 25% per time unit if not attended to. This rate is called the *deviation rate* (DR). It is also assumed that the rate a plate resumes its perfect satisfaction level is 25% per time unit if attended to. This rate is called the *correction rate* (CR). The plates may have different values such as a crystal plate versus a plastic plate, but in Figure 4.1, all plates have the same value.

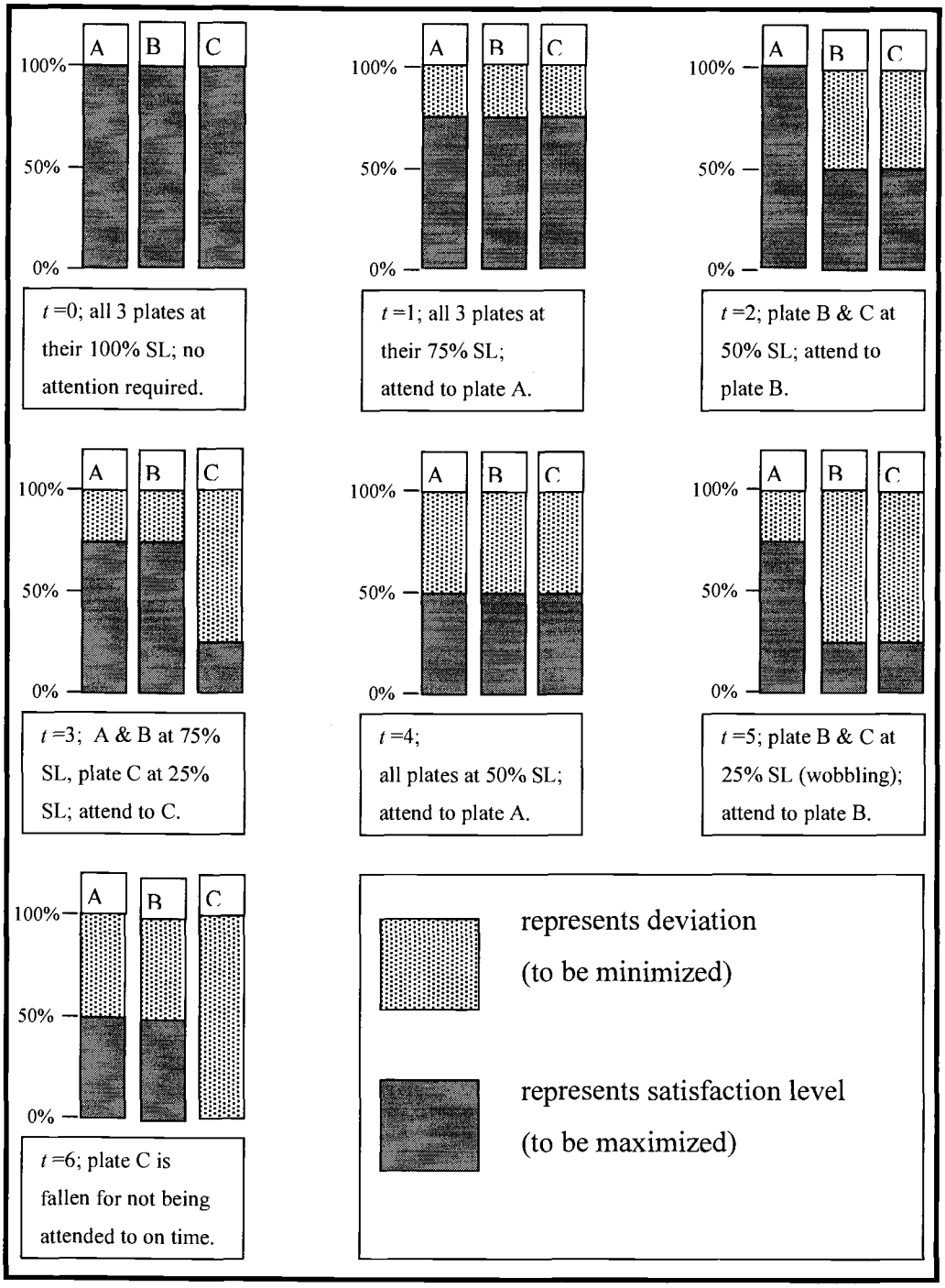


Figure 4.1 Attendance to three plates (tasks) from the juggler's (operator's) standpoint

4.3 VARIABLES TO MANIPULATE

Manipulating the following variables within the simplistic metaphor of spinning plates will add to the difficulty of the problem, but may improve the applicability of the model.

- The rate that a plate (task) deviates from the perfect status (deviation rate, DR)
- The rate that a plate (task) approaches its 100% Satisfaction Level or perfect status when attended to (correction rate, CR)
- The fact that the plates (tasks) have similar or different rates
- The cost of losing a plate (task), or its importance
- The cost of switching the attention between the plates (tasks)

4.4 APPLICATION OF THE METAPHOR

In the following sections, it is shown how one can explain some already known concepts or scenarios in a multi-tasking environment by using the plate metaphor or the bar graphs explained above.

4.4.1 Performance

Performance usually means, "How well is a task being performed over time?" Since a plate resembles a task in the plate metaphor, this question can be translated into, "How well is a plate spinning over time?" using the plate metaphor. In the bar graph, it would be, “How high is the SL kept over time?” The answer to the above questions could assess the quality of performance for a plate that has been wobbling too much for a period of time, and spinning very smoothly for some other period of time.

The question seeks the average performance over time, and not the status in an instance. An example for this view is when a person is asked, "Did you have a good

flight?" The flight might have been both smooth and rough, but overall how it is rated is the answer. A student's GPA (Grade Point Average) is another example in which the performance assessment is not based on a specific course. It is the overall performance of the student in all courses taken over time. However, different people might have different criteria for rating a student's performance. Some might consider the number of failed courses (the fewer the better), while some others might look for the number of courses with a grade of B or above (the more the better). Whatever the criteria is for assessing performance over time in a multi-tasking environment, it will be placed in the objective function of the mathematical model. It might be maximization of the average SL, minimization of the number of crashes, and so on.

4.4.2 Status

Before assessing the quality of performance over time, there is a need for a measure to assess the quality of performance at a specific point in time. The quality of performance at an instance in time is called *status* in this research. To compare with the questions asked in the previous section, the question to ask is "How well is the task being performed at this moment?" For instance, a driver glances at his/her vehicle's speedometer to assess the vehicle's speed status (under or over the speed limit). This is a judgment call based on the information gained at that instance. In the bar graphs, the *satisfaction level* (SL) of a task at a specific point in time, expressed in percentage, can be considered as a measurement unit for what is called as *status* here. In the plate metaphor, the measurement of how *well* or how *smooth* a plate is spinning is not easy, because there is no clear definition of *smoothness* in a spinning plate. A simple measurement unit for smoothness in spinning may be the *number of rotations per minute*. The lower this number, the harder the plate wobbles. In the mathematical model, the SL of a bar, between 0 and 1, will be used to indicate the status of a task at a specific point in time.

4.4.3 Importance, Weight or Value

The notions of *importance, weight* or *value* are used interchangeably in this research, and all differentiate the impact of different tasks on the *overall goal* even when they have the *same* amount of deviation from their perfect status. For example, suppose that a car is expected to drive in a straight highway whose lateral position with respect to the center of the lane is controlled via the steering wheel. When in highway, a reasonable angle for the steering wheel can range from - 60° to + 60° while the perfect angle is zero. If the driver holds the steering wheel at -30°, it is very likely that the deviation in the latitudinal control yields disastrous results (e.g., hitting a car in the adjacent lane, or getting off the highway). However, the deviation from the posted speed is not so critical, and one can drive 90 miles/hr in a 60 mile/hr highway with a relatively low chance of accident. If a reasonable speed range is assumed to be from zero to 120, this deviation is (90-60)/120=1/4th of the range. Note that the deviation in the latitudinal control had the same ratio (30/120) = 1/4th of the possible range.

As it was indicated earlier, importance is defined in terms of the contribution of a task to the overall goal. Thus, measurement of the importance should be a relative number with no specific unit. Assume that the overall goal is to minimize the summation of deviations of each task $(\sum_i \text{deviation}_i)$ in the system from its perfect status. Suppose that task (plate) A is twice as important as task (plate) B. This means, one unit of deviation in task A should be counted twice in the overall goal. To account for importance, the overall goal has to focus on minimizing the summation of deviation times importance of each task in the system $(\sum_i \text{deviation}_i \times \text{importance}_i)$. In such a case, *price* of a plate can perfectly substitute for the notion of importance. It should be noted that with such a definition for the overall goal, a plate that is spinning at 50% of its satisfaction

level, is costing the system (overall goal) with half of its price. This means that the cost of a plate depreciates gradually, and it does not lose its price suddenly only when it crashes.

The mathematical model in the next chapter will explicitly include the previously discussed concepts: correction rate (CR), deviation rate (DR), performance, status, and importance. The applications of the juggler metaphor can further be extended to include other concepts often faced in task management environments such as: deadline, urgency, priority, salience of task related stimuli, automation, etc. None of these concepts are explicitly included in the mathematical model due to the limited scope of this research. However, the conceptual application of the metaphor to explain these terms can be found in Appendix A.

CHAPTER 5: MODEL DEVELOPMENT

The objective is to develop a mathematical model as a standard of comparison for a human operator to attend to the most appropriate task at each point in time, considering the consequences of choosing this task in a fixed planning horizon. Many different behaviors and in turn different scoring mechanisms (objective functions) can be considered for the juggler paradigm. However, to keep the scope of this research manageable, only one of these scenarios is selected and the future chapters will all refer to that scenario. In the scenario selected (No. 4 in Table 5.1), the objective function will focus on maximizing the aggregated satisfaction level (SL) of the operator's performance on all tasks over the planning horizon. In this scenario, the operator will also be penalized for each time unit that a task is kept at zero SL. This scenario seems to represent a good majority of multitasking environments.

Different scenarios led to the development of different mathematical models in this research. These models are based on different behaviors at the minimum (zero) or maximum (100%) SLs. Table 5.1 and Figure 5.1 show a summary of these models. In the following sections, these scenarios are discussed in detail, and their mathematical models are presented. All models are formulated as a mixed (binary) integer-linear programming model. See the textbook by Hillier and Lieberman (1995) for further information on linear programming. Majority of these models, including No. 4 that is selected for the rest of this research, are based on discrete time unit and are computationally complex to solve. Appendix B shows why such models are NP-hard in the strong sense, and therefore need to be solved by efficient heuristic procedures explained in Chapter 6. Further, two polynomial time algorithm solutions are presented for two special cases in scenario 1 and scenario 3 of the following table.

Table 5.1 Summary of the mathematical models

No	Objective Function	Task behavior at	
		Zero SL	100% SL
1	Max total weighted average SL (simple objective function)	may be picked up later with no penalty	deteriorates if not attended to
2	Min total weighted flow time	may be picked up later with no penalty	task is considered accomplished and stays there without requiring further attention
3	Min total completion time	may be picked up later with no penalty	task is considered accomplished and stays there without requiring further attention
4	Max total weighted average SL (primary objective function)	penalized as long as it stays in zero SL, but may be picked up later	deteriorates if not attended to
5	Max total weighted average SL (modified objective function)	crashes and stays at zero SL	deteriorates if not attended to
6	Min total weighted no of crashes	crashes and stays at zero SL	deteriorates if not attended to
7	Max first crash time	crashes and stays at zero SL	deteriorates if not attended to

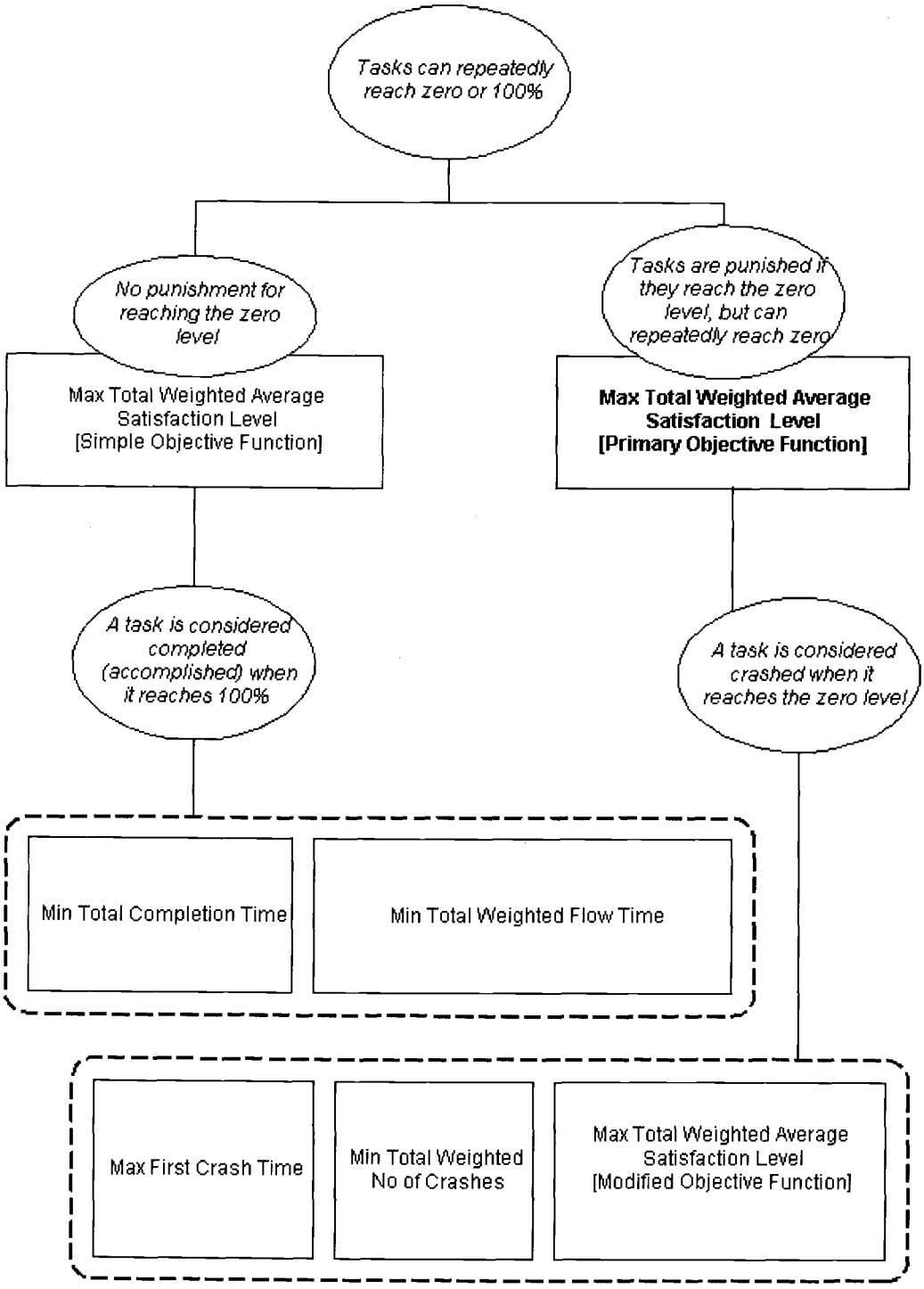


Figure 5.1 Overview of the mathematical models

5.1 MAXIMIZATION OF THE TOTAL WEIGHTED AVERAGE SL ACROSS TASKS (SIMPLE OBJECTIVE FUNCTION)

This model represents a scenario where tasks can repeatedly reach their perfect (100%) status or minimum (zero) SL with no penalty. When a task reaches zero level, it can be picked up any time later if attended to. Meantime, that task's contribution to the total score is zero, and there is no negative penalty. Also, if a task reaches its maximum level (100%) it contributes fully to the total score, but its SL will begin to drop the moment the attention is switched to a different task. This objective function is the core of all other objective functions discussed in this chapter. It captures the fundamental behaviors essential to the functionality of most generic multi-tasking environments. Since the scoring system does not change at the maximum (100%) or minimum (0%) SLs, it is also called *simple* objective function. If the maximum or minimum boundaries for SL are removed, there is an *optimal* rule for order of attendance to tasks. For further details refer to Appendix C.

5.1.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
w_i	weight (value) of task i in the overall objective.
$s_{i,t}$	status of task i at time t just before being attended to. $s_{i,0}$ = initial status; which is a known parameter between zero and 1 (100%).
$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$.
$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned.

M	a very large number.

5.1.2 The Model

Formula	#	Explanation
Max $Z = \frac{1}{(T+1)(\sum_{i=1}^{n} w_i)} \sum_{t=o}^{T} \sum_{i=1}^{n} (w_i)(s_{i,t})$	(0)	maximizes the weighted average satisfaction level (SL) among the tasks over the planning horizon T. $0 \le Z \le 1$
Subject to:		
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1}) CR_i - (1\text{-}x_{i,t-1}) DR_i$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t}) M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (%100).
$s_{i,t} \le (1 - z_{i,t}) M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$t = 1, 2, \ldots, T$		this range applies to constraints (2) – (6).
$i = 1, 2, \ldots, n$		this range applies to constraints (2) – (6).

5.2 MINIMIZATION OF THE TOTAL WEIGHTED FLOW TIME ASSUMING ATTAINING PERFECT SL (100%) IS COMPLETION OF A TASK

This is a scenario in which a task can actually get accomplished and left unattended for the rest of the planning horizon. It is assumed that all tasks are present in the

system at time zero. Therefore, flow time for a task in this context equals to the time that it gets accomplished and leaves the system. A task is fully accomplished the moment it reaches its maximum SL (100%). This scenario is similar to the classical machine-shop scheduling problems where a part gets processed on a machine and leaves the system. In this mathematical model, the planning horizon should be at least as large as the completion of the last job. Therefore, all tasks have to get accomplished (reach their maximum SL) one way or the other within the planning horizon to be able to fully evaluate this objective function.

If preemption is not allowed, a task's processing time will be equal to the time it is attended to until the time that it reaches its maximum SL (accomplished). What makes this problem different from typical machine-shop scheduling problems is that a task's processing time increases the longer it is unattended. In machine-shop scheduling, such jobs that take longer to process the later they are processed are called *deteriorating jobs*. Morton and Pentico (1993) discuss an example of deteriorating jobs where the later the hot steel slabs are processed, the more heat they require in order to get to the desirable temperature for rolling. Alidaee and Womer (1999) have a comprehensive review of theories for solving scheduling problems for deteriorating jobs on a single machine.

5.2.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
w_i	weight (value) of task i in the overall objective.
$s_{i,t}$	status of task i at time t just before being attended to $s_{i,0}$ = initial status, which is a known parameter between zero and 1 (100%).
$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$

$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
$y_{i,t}$	binary variable, Int ($s_{i,t}$), equals 1 when 100% is achieved and is zero otherwise.
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned; T should be large enough to allow for completing all tasks.
M	a very large number.

5.2.2 The Model

Formula	#	Explanation
$Max\ Z = \sum_{t=o}^{T} \sum_{i=1}^{n} (w_i)(y_{it})$ Subject to:	(0)	maximizes the weighted reward that is awarded to each task, at every time unit, since it is finished until the end of makespan. This in turn minimizes the weighted flow time, $Z \le (T+1)\ (\Sigma\ w_i)$.
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1})\ CR_i - (1\text{-}x_{i,t-1})\ DR_i + (y_{i,t-1})\ (DR_i)$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t})\ M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (100%).
$s_{i,t} \le (1 - z_{i,t})\ M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$y_{i,t-1} \le s_{i,t-1}$	(7)	$y_{i,t}$ = Int ($s_{i,t}$), starts giving a reward, at each time unit, after a task is finished until the end of the makespan.

t	=	1, 2, ..., T	this range applies to constraints (2) – (7).
i	=	1, 2, ..., n	this range applies to constraints (2) – (7).

Comments:

The algebraic summarization of constraints (2) – (6) can be represented as:

$$s_{i,t+1} = \text{Max}\ \{0, [\text{Min}\{1, s_{i,t} + (CR_i)\ (x_{i,t}) - (DR_i)\ (1-x_{i,t}) + (DR_i)\ (y_{i,t})\}]\ \}$$

5.3 MINIMIZATION OF THE TOTAL COMPLETION TIME ASSUMING ATTAINING PERFECT SL (100%) IS COMPLETION OF A TASK

The behavior of this system is very similar to the previous scenario. A task will be considered accomplished as soon as it reaches its maximum (100%) SL. A task will require no more attention when it is accomplished and is counted as one completed task. Tasks can repeatedly reach the zero level and resume a higher status with no penalty. The objective of this scenario is to minimize the total completion time of all tasks. A task's weight is of no importance for this objective function. For one special case, there is an *optimal* rule for order of attendance to tasks. The assumptions for this special case and the proof of the rule is discussed in Appendix D.

5.3.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
$s_{i,t}$	status of task i at time t just before being attended to $s_{i,0}$ = initial status, which is a known parameter between zero and 1 (100%).

$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$.
$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
$y_{i,t}$	binary variable, Int ($s_{i,t}$), equals 1 when 100% is achieved and is zero otherwise.
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned; T should be large enough to allow for completing all tasks.
M	a very large number.

5.3.2 The Model

Formula	#	Explanation
$Min\ Z = C$ Subject to:	(0)	minimizes the total completion time of all tasks. $Z \le T$
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1})\,CR_i - (1\text{-}x_{i,t-1})\,DR_i + (y_{i,t-1})\,(DR_i)$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t})\,M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (100%).
$s_{i,t} \le (1 - z_{i,t})\,M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$y_{i,t-1} \le s_{i,t-1}$	(7)	$y_{i,t}$ = Int ($s_{i,t}$), starts giving a reward, at each time unit, after a task is finished until the end of the makespan.
$C_i = T - \sum_{t} y_{i,t-1}$	(8)	completion time of a task equals to the makespan minus the rewards earned, at every time unit, since the time the task was finished.

C	$\geq C_i$	(9)	total completion time is the maximum completion time of all tasks.
t	= 1, 2, ..., T		this range applies to constraints (2) – (9).
i	= 1, 2, ..., n		this range applies to constraints (2) – (9).

Comments:

The algebraic summarization of constraints (2) – (6) can be represented as:

$$s_{i,t+1} = \text{Max}\ \{0, [\text{Min}\{1, s_{i,t} + (CR_i)\ (x_{i,t}) - (DR_i)\ (1-x_{i,t}) + (DR_i)\ (y_{i,t})\}]\ \}$$

5.4 MAXIMIZATION OF THE TOTAL WEIGHTED AVERAGE SL ASSUMING ATTAINING ZERO SL IS PENALIZED (PRIMARY OBJECTIVE FUNCTION)

Since this model is selected to be used and referred to in the following chapters, it is also called *primary* objective function. In this model, a task can reach its maximum (100%) or its minimum (zero) SL repeatedly over time. However, a penalty is assigned for every time unit that a task stays at zero SL. This will discourage the model from leaving some low weight tasks unattended for a long time in favor of some higher weight tasks. The degree of penalty is arbitrary and can change the behavior of the system depending on how large or small it is.

Too large a penalty encourages the model to prevent tasks from staying at a zero SL instead of keeping some other tasks at a very high level. On the other hand, too small a penalty may completely ignore some tasks that stay at a zero SL in favor of keeping some others at a very high level. Note: a zero penalty reduces this model back to the *simple* objective function in Section 5.1. The penalty coefficient is consistent across tasks in the objective function. Since this coefficient is also multiplied by the weight of the task, tasks with higher weight will be penalized more than lower weight tasks, which is what is expected. The penalty coefficient

used for the purpose of this research is 20% of the maximum weighted value that a task can get. **The great flexibility and wider application of this model compared to the other ones encouraged the researcher to use this model for the rest of this research. The model will be referred to as the *primary* objective function or simply the objective function in the next chapters.**

5.4.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
w_i	weight (value) of task i in the overall objective.
$s_{i,t}$	status of task i at time t just before being attended to. $0 \le s_{i,t} \le 1$ $s_{i,0}$ = initial status, which is a known parameter between zero and 1 (100%).
$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$.
$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
$y_{i,t}$	binary variable, [1 - Int (1- $s_{i,t-1}$)], equals zero when ZERO level is attained and is 1 otherwise.
k	arbitrary penalty coefficient activated when a task stays at a zero level. $k \ge 0$
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned.
M	a very large number.

5.4.2 The Model

Formula	#	Explanation
Max $Z = \frac{1}{(T+1)(\sum_{i=1}^{n} w_i)} \times \sum_{t=o}^{T}\sum_{i=1}^{n}(w_i)[s_{i,t} - k(1 - y_{i,t})]$ Subject to:	(0)	maximizes the weighted average satisfaction level (SL) among the tasks over the planning horizon T. It also penalizes a task if it is at the zero level. $-k \le Z \le 1$
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1})\,CR_i - (1\text{-}x_{i,t-1})\,DR_i$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t})\,M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (100%).
$s_{i,t} \le (1 - z_{i,t})\,M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$y_{i,t} \le M.\, s_{i,t-1}$	(7)	$y_{i,t} = 1 - \text{Int}(1 - s_{i,t-1})$, starts giving a flag, at each time unit as long as a task is at the zero level.
$t = 1, 2, \ldots, T$		this range applies to constraints (2) – (7).
$i = 1, 2, \ldots, n$		this range applies to constraints (2) – (7).

Comments:

The algebraic summarization of constraints (2) – (6) can be represented as:

$$s_{i,t+1} = \text{Max}\ \{0, [\text{Min}\{1, s_{i,t} + (CR_i)\ (x_{i,t}) - (DR_i)\ (1-x_{i,t})\}]\ \}$$

5.5 MAXIMIZATION OF THE TOTAL WEIGHTED AVERAGE SL ASSUMING ATTAINING ZERO SL IS TERMINATION (CRASH) OF A TASK (MODIFIED OBJECTIVE FUNCTION)

In this model a task can repeatedly reach and drop from the maximum (100%) SL, but as soon as it reaches the zero level it crashes and stays there. This model has exactly the same objective function as the simple objective function in Section 5.1, for which it is also called modified objective function. The only thing that differs between the two models is the constraints enforcing the crashing behavior. Figure 5.2 illustrates this difference. The motivation behind this model is that sometimes in real life a task is not worth pursuing when it crashes and cannot be attended to any longer. In general, such a behavior resembles tasks that have a deadline to be attended to, and after the deadline they are not worth pursuing. Refer to 'Deadline' in Appendix A.

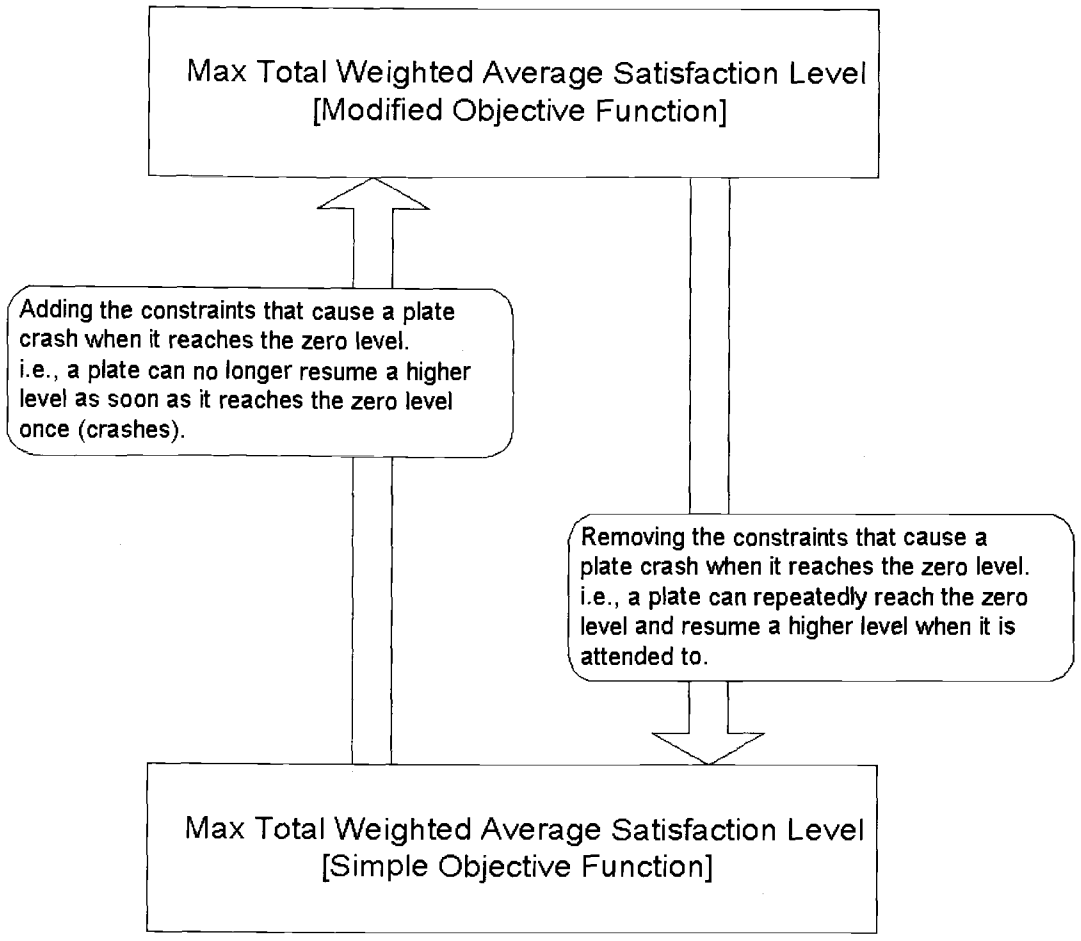


Figure 5.2 Comparison between the simple and modified objective functions in Sections 5.1 and 5.5

5.5.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
w_i	weight (value) of task i in the overall objective.
$s_{i,t}$	status of task i at time t just before being attended to $s_{i,0}$ = initial status, which is a known parameter between zero and 1 (100%).
$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$.

$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
$y_{i,t}$	binary variable, [1 - Int (1- $s_{i,t}$)], equals zero when ZERO level is attained and is 1 otherwise.
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned.
M	a very large number.

5.5.2 The Model

Formula	#	Explanation
Max $Z = \frac{1}{(T+1)(\sum_{i=1}^{n} w_i)} \sum_{t=o}^{T} \sum_{i=1}^{n} (w_i)(s_{i,t})$	(0)	maximizes the weighted average satisfaction level (SL) among the tasks over the planning horizon T. $0 \le Z \le 1$
Subject to:		
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1}) CR_i - (1\text{-}x_{i,t-1}) DR_i - (1 - y_{i,t-1}) (CR_i)$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t}) M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (100%).
$s_{i,t} \le (1 - z_{i,t}) M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$y_{i,t-1} \le M. s_{i,t-1}$	(7)	$y_{i,t}$ = 1 - Int (1 - $s_{i,t}$), starts giving a flag, at each time unit, after a task is crashed until the end of the makespan.

t	=	$1, 2, \ldots, T$	this range applies to constraints (2) – (7).
i	=	$1, 2, \ldots, n$	this range applies to constraints (2) – (7).

Comments:

The algebraic summarization of constraints (2) – (6) can be represented as:

$$s_{i,t+1} = \text{Max}\ \{0, [\text{Min}\{1, s_{i,t} + (CR_i)\ (x_{i,t}) - (DR_i)\ (1-x_{i,t}) - (CR_i)\ (1 - y_{i,t})\}]\ \}$$

5.6 MINIMIZATION OF THE TOTAL WEIGHTED NUMBER OF CRASHES ASSUMING ATTAINING ZERO SL IS TERMINATION (CRASH) OF A TASK

The objective function of this model is different from the previous models in that a task's satisfaction level per se does not contribute to the objective function. In this scenario also a task crashes when it hits the zero level and cannot increase its status any longer. The focus of this model, however, is to minimize the total weighted number of tasks that crash over the planning horizon. It is assumed that some tasks are likely to crash, but the goal is to minimize the number of these losses considering their weight (value). In this model, it is of no value how well a task's status is kept as long as it does not crash. The trade off takes place in choosing between saving a few higher weight tasks versus many lower weight ones.

5.6.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
w_i	weight (value) of task i in the overall objective.

$s_{i,t}$	status of task i at time t just before being attended to. $s_{i,0}$ = initial status, which is a known parameter between zero and 1 (100%).
$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$.
$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
y_i	binary variable, , [1 - Int (1- $s_{i,t}$)], equals zero when ZERO level is attained even once during the planning horizon and is 1 otherwise.
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned.
M	a very large number.

5.6.2 The Model

Formula	#	Explanation
$Min\ Z = \sum_{i=1}^{n} (w_i)(1 - y_i)$ Subject to:	(0)	minimizes the weighted # of crashes over the planning horizon, $0 \le Z \le (\Sigma\ w_i)$.
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1})\, CR_i - (1\text{-}x_{i,t-1})\, DR_i$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t})\, M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (100%).
$s_{i,t} \le (1 - z_{i,t})\, M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$y_i \le M.\ s_{i,t-1}$	(7)	y_i = 1 - Int (1 - $s_{i,t}$), signals the task that has crashed sometime during the makespan.

t	=	1, 2, ..., T	this range applies to constraints (2) – (7).
i	=	1, 2, ..., n	this range applies to constraints (2) – (7).

Comments:

The algebraic summarization of constraints (2) – (6) can be represented as:

$$s_{i,t+1} = \text{Max}\ \{0, [\text{Min}\{1, s_{i,t} + (CR_i)\ (x_{i,t}) - (DR_i)\ (1-x_{i,t})\ \}]\ \}$$

5.7 MAXIMIZATION OF THE FIRST CRASH TIME

The objective function of this model is similar to the last model in that a task's satisfaction level per se does not contribute to the objective function. However, weight of the tasks has no contribution in this model either. In this scenario, if a task hits the zero level, it will crash the whole system of tasks and it ends the task management operation. The focus of this model is to postpone the crash of any of the tasks, and in turn the crash of the system, as long as possible. This applies to situations in which all tasks are vital to the existence of the system and the operator cannot afford to lose any of them. In this case, the objective is to not lose any task as long as possible within the planning horizon.

5.7.1 Parameters and Variables

DR_i	deviation rate of task i if not attended to.
CR_i	correction rate of task i if attended to.
$s_{i,t}$	status of task i at time t just before being attended to $s_{i,0}$ = initial status, which is a known parameter between zero and 1 (100%).
$x_{i,t}$	binary variable: is task i attended to at time t? 0 (No) or 1 (Yes).
$A_{i,t}$	auxiliary variable to help find $s_{i,t}$

$z_{i,t}$	binary variable: $z_{i,t} = \begin{cases} 1, A_{i,t} \le 0 \\ 0, A_{i,t} > 0 \end{cases}$
y_t	binary variable, [1 - Int (1- $s_{i,t}$)], equals zero when ZERO level is attained for any task i at time t and is 1 otherwise.
i	= 1, 2, …, n independent tasks.
t	= 0, 1, 2, …, T units of time planned.
M	a very large number.

5.7.2 The Model

Formula	#	Explanation
$Min\, Z = \sum_{t=1}^{T} (t)(1 - y_t)$ Subject to:	(0)	minimizes the summation of points in time since a task is crashed until the end of makespan. This in turn minimizes the first crash time, $0 \le Z \le (T)$.
$\sum_{i=1}^{n} x_{i,t} \le 1 \qquad t = 0, 1, \ldots, T\text{-}1$	(1)	at most one task at any time t can be attended to.
$A_{i,t} = s_{i,t-1} + (x_{i,t-1})\, CR_i - (1\text{-}x_{i,t-1})\, DR_i - (1 - y_{t-1})(CR_i)$	(2)	current state of a task depends on its previous state and whether it was attended to in the last period.
$s_{i,t} \le A_{i,t} + (z_{i,t})\, M$	(3)	paired with constraint (6): maximum satisfaction level (SL) of a task is 1 (100%).
$s_{i,t} \le (1 - z_{i,t})\, M$	(4)	paired with constraint (5): minimum satisfaction level (SL) of a task is zero—cannot be negative.
$s_{i,t} \ge 0$	(5)	see the explanation for constraint (4).
$s_{i,t} \le 1$	(6)	see the explanation for constraint (3).
$y_{t-1} \le M.\, s_{i,t-1}$	(7)	y_t = 1 - Int (1 - $s_{i,t}$), signals that a task has crashed at time t.
t = 1, 2, …, T		this range applies to constraints (2) – (7).
i = 1, 2, …, n		this range applies to constraints (2) – (7).

Comments:

The algebraic summarization of constraints (2) – (6) can be represented as:

$$s_{i,t+1} = \text{Max}\{0, [\text{Min}\{1, s_{i,t} + (CR_i)(x_{i,t}) - (DR_i)(1-x_{i,t}) - (1-y_t)(CR_i)\}]\}$$

CHAPTER 6: TABU SEARCH-BASED HEURISTIC METHOD

6.1 INTRODUCTION

For every difficult combinatorial optimization problem, there is a need for an effective algorithm to find its optimal or near-optimal solution in a reasonable time. For such problems, implicit enumeration techniques such as branch-and-bound will not be effective. As the problem in this research is proved to be NP-hard, clearly there is a need for an effective heuristic algorithm. Morton and Pentico (1993) explain that the new heuristic search techniques, although not yet fully mature, have proven to be significantly helpful in the scheduling domain. One category of these heuristic methods uses *intensification/diversification* techniques. Within this category, the popular heuristics are: *tabu search*, *simulated annealing*, and *genetic algorithms*.

Tabu search, first introduced by Glover (1986), is the heuristic procedure selected and used to solve the problems addressed in this research. Tabu search explores the solution space to find an optimal or near-optimal solution. It has been applied successfully to a variety of applications, especially scheduling (Barnes et al., 1995). This method overcomes the shortcomings of its competitors in escaping the trap of local optimality. Unlike other methods that are memoryless (simulated annealing and other randomized approaches) or use rigid memory structures (branch-and-bound), tabu search uses flexible memory structure. The search history, recorded in the memory, along with the conditions embodied in tabu restrictions, constrains or frees the search process to reach better solutions more effectively. The long-term memory component of tabu search enables the search to intensify the search in the region that is historically found good, or diversify the search to less explored regions.

The fundamental principles of tabu search are discussed in detail in Glover (1989, 1990a, and 1990b). In the next section, the mechanism of the tabu search is described. Then, the initial solution finding methods and steps associated with the algorithm for the problem in hand is discussed. Finally, an application of the tabu search heuristic to an example problem is presented.

6.2 TABU SEARCH MECHANISM

The foundation of tabu search is built upon the following features (Glover, 1990b):

1. flexible memory structure: it allows exploitation of the evaluation criteria and historical search information more thoroughly. This can be contrasted to rigid memory structures (branch-and-bound) or memoryless structures (simulated annealing and other randomized approaches).
2. a memory structure control mechanism: this mechanism imposes or frees the constraints on the search process. Tabu restrictions and aspiration criteria are in fact the controls.
3. short term and long-term memories: the combination of these two memory functions allows for intensifying or diversifying the search. Intensification means more thoroughly searching neighborhoods that are historically found good. Diversification, on the other hand, searches those neighborhoods that have been relatively unattended. The former search digs deeper while the latter moves to new regions.

The hill-climbing like tabu algorithm progresses the search at each step to a better (higher evaluation) move. When it reaches the highest evaluation (peak of the hill), all local moves will lead to inferior solutions and hence none will be chosen. But this peak may be the local optimum and not the global one. Tabu search has the capability of not getting caught in the trap of local optima by moving the search to new regions until a (near) global optimum is reached. The search, however, does

not stop when (and if) it reaches the optimum and it has to be stopped through other criteria such as number of iterations or time elapsed.

The first step for tabu search is having an initial solution. The initial solution is an arbitrary solution that can be feasible or infeasible. It can be generated randomly or through systematic procedures. Intuitively, the better the starting point, the easier (faster) will the search get to (near) optimal solutions. This is especially of concern for larger problems. In this research, two methods for generating the initial solution are employed. These methods are discussed in detail in the next section.

Once an initial solution is determined, its neighborhood solutions can be explored by perturbing it. The value of each of these solutions is determined by the objective function, which in this research is maximizing the weighted satisfaction level over time. These solutions have to pass through a tabu filter whose goal is to escape the cyclic trap of local optima. The tabu filter is implemented through comparison of neighborhood solutions against a set of restricted moves listed in the tabu list (TL). This list is constructed based on the recent change in previous best solution that resulted in a better solution. The tabu list records these changes or moves in the order they were received. The size of this list is determined through experimentation, which usually has a logarithmic growth by increasing the problem size.

Tabu restrictions prevent the search from repeating the moves that are expected to reach a local optimum already attained. So after a set of neighborhood solutions is generated, the best local move among them is compared against the tabu list. If the move is restricted, it is normally ignored and the second best move is considered. There are cases, however, that a restricted move may have a better value than the best global value found so far, the aspiration level. Only in such a case, when aspiration criterion is met, a tabu restriction can be overwritten. The best move,

after filtering against the tabu list and aspiration criterion, is selected for future perturbation, and generation of a new neighborhood. This move is recorded into the candidate list (CL). Besides the previous two filters, tabu list and aspiration level, the best solution in each configuration has to be compared against the candidate list as well. This is to avoid perturbing a seed and exploring its neighborhood that was already explored. This process is repeated until the search is terminated.

Short-term memory, or the tabu list, is the core of tabu search process. Long-term memory, or the frequency matrix, can enhance the effectiveness of the short-term memory. They can focus further on searching the regions that were historically promising (intensification); or direct the search to neighborhoods that were rarely visited before (diversification). The information on all the previous moves required to exploit the long-term memory is recorded in the frequency matrix. After one complete set of search is performed, with the aid of long-term memory, a new complete search restarts. Although the starting seed is the same, the restrictions drawn from the long-term memory direct the search differently than before. The number of restarts is arbitrary and depends on how much time a researcher is willing to sacrifice for what quality of solutions. Evidently, the more restarts imposed, the better the quality of solutions, and the more time consuming the search will be.

6.3 INITIAL SOLUTION

Selecting an initial solution is the starting point for generating neighborhood solutions and exploring them. In general, a good starting solution is believed to expedite the search process to reach a better value. The question is what initial solution is considered good. The answer can only be found through experimentation and comparison. Also, because there is more than one problem investigated with different sizes and parameters, it is the initial solution *finding mechanism* that is compared against another, not the solution itself. The procedure

can be deterministic or random. In this research, two random procedures are used for generating initial solutions. One is simply a random order of tasks generated without considering any of the task parameters. The other one is also a random order, but the odds of a task appearing in that random order is different from the other tasks. The former method that tasks have equal chance of being selected in the order of attendance is called uniform random. The latter procedure, in which tasks have different chances or weights of being selected, is called weighted random.

6.3.1 Uniform Random

Imagine for each task to be attended there is a ball with the task number on it. Drop all the balls in a sack, and at every time unit within the planning horizon, blindly pick a ball from the sack to decide which task to attend to, record the number, and drop the ball back. That is similar to how the initial solutions were generated in this research with a uniform random method. No additional rationale or rule is considered in generating an order of attendance in this method. None of the parameters of the system such as correction rate, deviation rate, weights, planning horizon, etc. had any effect in this procedure. Thus for n number of tasks, each task has a $1/n$ chance to be selected at each time unit. This procedure is called uniform random.

6.3.2 Weighted Random

Imagine two big tanks of water. One tank has a very small hole from which water is dripping and a very big bowl for refilling the water. The other tank has a very big hole, but a very small bowl to be refilled with. The goal is to keep the level of water high in these two tanks. Naturally, more time is spent on the second tank (big hole and small refilling bowl) because it requires more attention. Occasionally, one also attends to the first tank (small hole and big refilling bowl), but it does not take much of his/her time to keep the level of water high in it. Now if it is told that

having a high level of water in one of these tanks is more important than the other one, it is intuitive that the more important tank will be attended to more frequently.

The above example is the idea behind weighted random procedure for generating initial solutions. Instead of holes and refilling bowls, deviation rates (DR) and correction rates (CR) are used. The importance of a task is determined by its weight or value (w). A ratio $w\frac{DR}{CR}$ has been developed based on the last three parameters (DR, CR, and w) to determine the chances of tasks being attended to during the planning horizon. The higher is this ratio for a task, the more likely is for that task to be attended to. Thus at each time unit task i is attended to with a probability of $\frac{w_i \frac{DR_i}{CR_i}}{\sum_i w_i \frac{DR_i}{CR_i}}$. Although a smarter procedure compared to uniform random, this method does not consider the satisfaction level of a task, which is in many cases the motivation behind an order of attendance. A weighted random procedure is merely a reflection of frequency of attendance to tasks in the long run. That is, the tasks that are selected in the initial solution may be very similar to the ones suggested by the optimal solution, but the order in which they are listed can be far different.

6.4 GENERATION OF NEIGHBORHOOD SOLUTIONS

After selecting an initial solution as a seed, one can go after generating a neighborhood of solutions whose values are to be explored. The neighborhood of solutions in this research is generated very simply. Pick a seed (initial solution) in which for each time unit in the planning horizon an order of attendance to tasks is predetermined. Start with the first time unit, exchange the first task with another, and leave the rest of the order of attendance unchanged. This will be a new solution in the neighborhood. For n number of tasks one can generate up to $(n-1)$ solutions

by changing the tasks only in the first time unit. This is also called a perturbation on the task. When all the solutions spawned from the first time unit are generated, move on to the second time unit. At this time repeat the same procedure (change the tasks in that time unit), but leave the order of attendance before and after time unit 2 unchanged. In general, a neighborhood solution is exactly the same as the initial seed, but differs in the task that is attended to only in one of the time units. To generate all the neighborhood solutions, continue the task perturbation for every time unit until the planning horizon is reached. For a planning horizon 0, 1, 2, ..., T and n number of tasks, with any seed one can generate $(n-1)^{T+1}$ solutions in the neighborhood.

Every one of the neighborhood solutions generated has to be evaluated and checked against the tabu list. If a solution exists in the tabu list, it should be disregarded unless its value is higher than the aspiration level, the best value found over all the neighborhood solutions so far. The best solution among the legitimate solutions remaining in the neighborhood will be admitted into the candidate list and used as the new seed for next generation of solutions.

6.5 STEPS OF TABU SEARCH

The research problem is focused on finding an optimal or near-optimal order of attendance to tasks at each time unit in the planning horizon. This order should maximize the weighted average satisfaction levels of tasks over time. The steps associated with the tabu search-based heuristic algorithm to solve this problem are presented next.

6.5.1 Step 1—Initial Solution

Randomly generate an initial solution for the order of tasks to be attended to at each time unit in the planning horizon. This solution can be generated through one of the two methods explained before.

6.5.2 Step 2—Neighborhood Solutions

A neighborhood of solutions should be generated using the initial solution as a seed. This neighborhood is created by perturbing the tasks at each time unit in the planning horizon.

6.5.3 Step 3—Evaluation of Solutions

Evaluate the weighted average satisfaction level over time for every solution generated. Check every solution against the tabu list and filter out solutions that are restricted. A restricted solution (move) may not be disregarded only if its value is higher than the aspiration level—the best value found among all the neighborhoods to this point. Select the best move of the current neighborhood solutions and update the following tabu search parameters:

6.5.3.1 Tabu List

Every time the best move in the neighborhood is selected, update the tabu list by recording the changed parameter of the initial seed that resulted in the best move. Therefore, the tabu list includes the parameter before the change, which produced the best seed of the neighborhood after being changed. This list is updated circularly in the order it was received. As the tabu list size is fixed in this research, if the list is filled to its size, the next item that is admitted to it replaces the oldest item in the list. Thus, it is a fixed-size storage with first-in-first-out rule. The tabu list size is found through experiments and it should be adjusted for the size of problem. For the purpose of this research, the following formula has been found to be appropriate for the tabu list size: $3+\left[\log_2(n\times T)\right]$ where n is the number of tasks, and T is the planning horizon.

Initial experimentation indicated no gain in using the variable-size tabu list instead of the fixed-size. Thus, it was not considered for further experimentation.

6.5.3.2 Candidate List

After generating the neighborhood from the initial seed, and filtering them against the tabu list, aspiration criterion and non-identical candidate, one legitimate solution is selected from that neighborhood as the best move. That solution should be recorded into the candidate list, and the number of entries into the candidate list should be increased by one. The new candidate list entry will serve as the new seed for the next generation of neighborhoods. The very first record of the candidate list is the same as the first initial solution.

6.5.3.3 Aspiration Level

The aspiration level is equal to the value of the best solution found since the very start of the search. Upon admitting a new solution to the candidate list, its value has to be compared against the aspiration level. If this value is better, aspiration level has to be updated with this new value.

6.5.3.4 Index List

The index list is a subset of the candidate list. It contains the local optima among the solutions in the candidate list. If a value of a candidate list solution is better than the solution immediately before and after, it will be admitted into the index list. A maximum size can be set for the index list to signal termination of the search process if reached. For the purpose of this research, the maximum size was set to the following empirical formula: $\left\lceil \frac{1}{2}\{\log_2(n \times T)\}^{2.1} \right\rceil$ where n is the number of tasks, and T is the planning horizon.

6.5.3.5 Number of Iterations

For every new candidate list solution (seed), a new set of neighborhood solutions is generated, and the number of entries in the candidate list is increased by one. This

also applies to the number of iterations, which needs to be increased by one. A maximum number of iterations could be set so that if all the other search termination conditions fail, this condition will ultimately terminate the search. The empirical formula used for the maximum number of iterations used in this study was: $\lfloor \{\log_2(n \times T)\}^{2.1} \rfloor$ where n is the number of tasks, and T is the planning horizon.

6.5.3.6 Number of Iterations Without Improvement

The number of iterations with no improvement would increase the same way as the previous section except that it increases by one only if there is no improvement in the objective function when moving from one solution to the next in the candidate list. The maximum number of iterations without improvement could be used to indicate the end of the search process if reached. For the purpose of this study, the following empirical formula was believed to determine a good maximum number.

Maximum number of iterations with no improvement = $\lceil \log_2(n \times T) \rceil$

Where n is the number of tasks, and T is the planning horizon

6.5.3.7 Long-Term Memory (LTM)

To diversify or intensify the search, a long-term memory frequency matrix is used. This matrix has two dimensions. One dimension is the task (n tasks) and the other dimension is the time (T planning horizon). Therefore, it is a (n * T) matrix in which cell (i, j) represents the number of times task i was attended to at time j by the solutions in the candidate list. Upon admitting a new solution into the candidate list, the frequency matrix also has to be updated with the information provided by the new solution. After the search process has stopped, the information in the frequency matrix (long-term memory) can be used to restart the search. The search restarts with the same seed initially used, but the information in the long-term memory directs it differently than the first time. Previous studies involving

applications of tabu search have shown that two restarts is a good tradeoff between the solution quality and computational efficiency (Logendran and Sonthinen, 1997).

At every restart, tabu list, candidate list, index list, number of iterations, and number of iterations with no improvement is reset to zero. In the following, two different uses of long-term memory are explained.

6.5.3.7.1 LTM_Max

The frequency matrix indicates how many times the best moves in the search process suggested that a certain task has to be attended to at a certain time. Thus, the maximum number in the matrix indicates that the majority of the good solutions agree on attending to that task at that point in time. In case of ties between two maximums, a column-wise strategy is used to pick the first maximum. LTM_Max is a method that uses this information and restarts the search with the same seed, but this time it fixes attendance to that particular task at that particular time. Thus, all the new solutions generated are generated around that *fixed* combination of task-time. This method is also called intensification as it intensifies the search around the solutions that were repeatedly found good by the previous search. After every restart, the frequency matrix is reset to zero.

6.5.3.7.2 LTM_Min

In contrast to LTM_Max, the minimum number in the same frequency matrix would indicate how few times (or none) the attendance to a certain task at a certain time was recommended by the previous search. This also means that any neighborhood around that combination of task-time is highly unexplored. In case of ties between two minimums, a column-wise strategy is used to pick the first minimum. LTM_Min is a method that employs this information and assures, for the

next restart of the search, that any solution generated revolves around this *fixed* task-time combination. Thus, it gives the search process an opportunity to explore the least explored areas. For this reason, it is also called diversification. The frequency matrix has to be reset to zero prior to every restart.

6.5.4 Step 4—Search Termination Conditions

The above process is repeated until one of the search termination conditions is met. The typical conditions for terminating the search process are:

- Maximum number of entries in the candidate list is reached—not used in this study.
- Maximum number of iterations is reached. To set this number, the empirical formula used for this research was: $\lfloor \{\log_2(n \times T)\}^{2.1} \rfloor$ where n is the number of tasks, and T is the planning horizon.
- Maximum number of consecutive iterations with no improvement in the objective function is reached. To set this number, the empirical formula used for this research was: $[\log_2(n \times T)]$ where n is the number of tasks, and T is the planning horizon.
- Maximum number of local optima or entries in the index list is reached. This number can be a fraction of the maximum number of iterations. For the purpose of this research, this number was set to: $\left\lceil \frac{1}{2} \{\log_2(n \times T)\}^{2.1} \right\rceil$ where n is the number of tasks, and T is the planning horizon.
- Maximum computation time is reached—not used in this study.
- Maximum number of restarts is reached if the long-term memory is used. For the purpose of this study, this number was set to 2.

One or more of the above conditions should be employed to stop a search process; otherwise the search would not stop even though it might have reached the global optimum.

6.6 APPLICATION OF TABU SEARCH TO A PROBLEM INSTANCE

In order to clarify the steps listed in Section 6.5, a small example is explained in the following paragraphs with the application of those steps. Consider a problem with 3 tasks, a planning horizon of 4 time units, initial satisfaction level of 0.983 and the following deviation rates, correction rates, and weights.

Table 6.1 Parameters of the problem instance

Number of Tasks:	3		
Planning Horizon:	4		
Initial SL:	0.983		
Task:	0	1	2
CR: (1 - 10) × DR	0.16	0.18	0.1
DR: (0.01 - 0.10)	0.08	0.02	0.02
W: (1 - 10)	1	3	10
w(DR/CR)	0.5	0.33	2

6.6.1 Step 1—First Iteration

The initial seed for this example was generated using the weighted random procedure. The task with a higher $w\frac{DR}{CR}$ has a higher chance of appearing in the order of attendance. As it is shown in Table 6.2, task 2 has the highest chance (71%) and task 1 has the least chance (12%) to appear in the initial seed.

Table 6.2 Likelihood of tasks appearing in the initial seed

	Task			Sum
	0	1	2	
w(DR/CR)	0.5	0.33	2	2.83
w(DR/CR)/total	18%	12%	71%	100%

The initial seed randomly generated for this example is: {0, 0, 2, 2}

6.6.2 Step 2—First Iteration

Using the initial seed {0, 0, 2, 2}, generate the neighborhood by perturbing on the tasks at each time unit, yet one time unit at a time.

Table 6.3 Neighborhood generated at the first iteration

Iteration	Value	Time			
1		0	1	2	3
0 (Initial)	968.25	0	0	2	2
1	973.21	1	0	2	2
2	977.44	2	0	2	2
3	972.15	0	1	2	2
4	972.97	0	2	2	2
5	959.54	0	0	0	2
6	963.85	0	0	1	2
7	967.68	0	0	2	0
8	969.55	0	0	2	1

6.6.3 Step 3—First Iteration

The best value selected among the solutions generated in the neighborhood is solution 2, {2, 0, 2, 2} with the value of 977.44. This solution is used to update the following parameters.

6.6.3.1 Tabu List—First Iteration

Recall that the initial solution was {0, 0, 2, 2}. The best solution generated in the neighborhood that the search procedure is going to move to is {2, 0, 2, 2}. Therefore the tabu list should be updated with {0, x, x, x} meaning that no solution of such kind will be allowed in the next iteration unless its value is superior to the aspiration level. If the formula proposed in Section 6.5.3.1 were to followed, the fixed size of the tabu list would be:

$$\text{Fixed tabu list size: } 3+\left[\log_2(n\times T)\right] = 3+[\log_2(4\times 3)] = 6$$

However, for the sake of simplicity and in order to show the circular entry of records in the tabu list, **the tabu list's size in this application is assumed equal to 1.**

6.6.3.2 Candidate List—First Iteration

The best solution of the neighborhood will be recorded in the candidate list. The candidate list already has one record that is the initial solution {0, 0, 2, 2}. The new record added will be {2, 0, 2, 2}. Note that the new record is not tabu, and it has not existed in the candidate list before.

Increase the number of the candidates in the candidate list by one, which will be equal to 2. Because this number is less than 100 (arbitrary maximum), the search can continue.

6.6.3.3 Aspiration Level—First Iteration

The value (968.25) of the initial solution {0, 0, 2, 2} was the initial value for the aspiration level. This value must be updated with the value (977.44) of the new seed in the candidate list {2, 0, 2, 2} because it is superior.

6.6.3.4 Index List—First Iteration

The initial solution {0, 0, 2, 2} by convention is always admitted to the index list. The new solution {2, 0, 2, 2} with the value of 977.74 is superior to the previous (initial) solution {0, 0, 2, 2} with the value of 968.25. Therefore the new solution has a potential to be admitted into the index list if the next new solution's value is inferior to 977.74. At this moment, total number of records in the index list is equal to one.

6.6.4 Step 4—First Iteration

The conditions for terminating the search are:

- The maximum number of iterations would be:
  $\lfloor \{\log_2(n \times T)\}^{2.1} \rfloor = \lfloor \{\log_2(3 \times 4)\}^{2.1} \rfloor = 14$ However, for the sake of simplicity in this example, this number is set to 4. At this iteration, the number of iterations equals to 1, so the search can continue.
- Maximum number of consecutive iterations with no improvement would be:
  $\lfloor \log_2(n \times T) \rfloor = \lfloor \log_2(3 \times 4) \rfloor = 3$ At this iteration, the number of iterations with no improvement equals to zero, so the search can continue.
- Maximum entries in the index list is:
  $\left\lceil \frac{1}{2}\{\log_2(n \times T)\}^{2.1} \right\rceil = \left\lceil \frac{1}{2}\{\log_2(3 \times 4)\}^{2.1} \right\rceil = 7$ At this iteration, there is only one entry in the index list so the search can continue.
- Maximum number of restarts is 2. At this iteration, the number of restarts is zero so the search can continue.

6.6.5 Step 2—Second Iteration

The new seed {2, 0, 2, 2} is the last selected entry in the candidate list. The new neighborhood is generated based on this seed in the following table. The first neighborhood solution is crossed out because it violates the tabu restriction {0, x, x, x} and its value was inferior to the aspiration level (977.44).

Table 6.4 Neighborhood generated at the second iteration

Iteration 2	Value	Time			
		0	1	2	3
0	977.44	2	0	2	2
1	968.25	0	0	2	2
2	973.21	1	0	2	2
3	977.18	2	1	2	2
4	972.71	2	2	2	2
5	974.01	2	0	0	2
6	978.32	2	0	1	2
7	976.87	2	0	2	0
8	978.74	2	0	2	1

6.6.6 Step 3—Second Iteration

The best solution in this neighborhood is the last solution {2, 0, 2, 1} with the value of 978.74. This solution is used to update the following parameters.

6.6.6.1 Tabu List—Second Iteration

The new solution that the search is going to move to is: {2, 0, 2, 1}. Recall that the seed used for generating this neighborhood was {2, 0, 2, 2}. Therefore, the new tabu restriction would be {x, x, x, 2}. As the size of the tabu list is selected to be

equal to 1 in this problem, the new entry in the tabu list will replace {0, x, x, x}, the first record.

6.6.6.2 Candidate List—Second Iteration

The new entry in the candidate list will be the best solution of the last neighborhood generated that is {2, 0, 2, 1}. The number of records in the candidate list should be increased by one, which will now be equal to 3. This number is far below the maximum of 100 entries, so the search can continue.

6.6.6.3 Aspiration Level—Second Iteration

The value of the aspiration level should be updated if it is inferior to the best value found in the last neighborhood. As 978.74 of the last candidate list is superior to the aspiration level (977.44), the value of the aspiration level has to be updated with 978.74.

6.6.6.4 Index List—Second Iteration

The value of the new candidate in the candidate list (978.74) is superior to its previous candidate (977.44). Therefore, the previous solution lost its chance to be a local optimum and the new one becomes a potential local optimum if the value for its next entry is inferior.

6.6.7 Step 4—Second Iteration

The conditions for terminating the search are:

- The maximum number of iterations is 4. At this iteration, the number of iterations equals to 2, so the search can continue.
- Maximum number of consecutive iterations with no improvement is 3. At this iteration, the number of iterations with no improvement equals to zero, so the search can continue.

- Maximum entries in the index list is 7. At this iteration, there is only one entry in the index list so the search can continue.
- Maximum number of restarts is 2. At this iteration, the number of restarts is zero so the search can continue.

6.6.8 Repeat the Cycle

The following tables show how the above procedures will be implemented for the next two iterations.

Table 6.5 Neighborhood generated at the third iteration

Iteration 3	Value	Time			
		0	1	2	3
0	978.74	2	0	2	1
1	969.55	0	0	2	1
2	972.92	1	0	2	1
3	976.04	2	1	2	1
4	974.01	2	2	2	1
5	969.6	2	0	0	1
6	970.61	2	0	1	1
7	976.87	2	0	2	0
8	977.44	2	0	2	2

Solution 8 is rejected, as it is a tabu move and/or it already existed in the candidate list.

New candidate: {2, 0, 2, 0}, (976.87)

New tabu list: {x, x, x, 1}

Updated candidate list: {0, 0, 2, 2}, {2, 0, 2, 2}, {2, 0, 2, 1}, {2, 0, 2, 0}

Updated aspiration level: 978.74 > 976.87, so no update needed.

Updated Index list: {0, 0, 2, 2}; {2, 0, 2, 1} because 977.44 < 978.74 > 976.87.

Table 6.6 Neighborhood generated at the fourth iteration

Iteration 4	Value	Time			
		0	1	2	3
0	976.87	2	0	2	0
1	967.68	0	0	2	0
2	972.64	1	0	2	0
3	977.75	2	1	2	0
4	973.28	2	2	2	0
5	966.58	2	0	0	0
6	972.04	2	0	1	0
7	978.74	2	0	2	1
8	977.44	2	0	2	2

In the above table, solution 7 is rejected, as it is a tabu move and its value is no greater than the aspiration level (978.74). Solution 8 is also rejected, as it already existed in the candidate list.

New candidate: {2, 1, 2, 0}, (977.75)

New tabu list: {x, 0, x, x}

Updated candidate list: {0, 0, 2, 2}, {2, 0, 2, 2}, {2, 0, 2, 1}, {2, 0, 2, 0}, {2, 1, 2, 0}

Updated aspiration level: 978.74 > 977.75, so no update needed.

Updated index list: {0, 0, 2, 2}, {2, 0, 2, 1}, no new local optimum is added.

The following table shows a summary of solutions generated and their corresponding parameters updated.

Table 6.7 Summary of solutions for the first four iterations

Iteration	CL		tabu	IL	AL	
	Value	Solution		Solution	Value	Solution
0 (initial)	968.25	{0, 0, 2, 2}		{0, 0, 2, 2}	968.25	{0, 0, 2, 2}
1	977.44	{2, 0, 2, 2}	{0, x, x, x}		977.44	{2, 0, 2, 2}
2	978.74	{2, 0, 2, 1}	{x, x, x, 2}		978.74	{2, 0, 2, 1}
3	976.87	{2, 0, 2, 0}	{x, x, x, 1}	{2, 0, 2, 1}	978.74	{2, 0, 2, 1}
4	977.75	{2, 1, 2, 0}	{x, 1, x, x}		978.74	{2, 0, 2, 1}

The following table shows how the maximum frequency matrix is updated based on the selected candidate lists in Table 6.7.

Table 6.8 Updating steps of the maximum frequency matrix

CL	Task	Time			
		0	1	2	3
{0, 0, 2, 2} (initial seed) Reset table to zero	0	0	0	0	0
	1	0	0	0	0
	2	0	0	0	0
{2, 0, 2, 2}	0	0	1	0	0
	1	0	0	0	0
	2	1	0	1	1

Table 6.8 (Continued) Updating steps of the maximum frequency matrix

CL	Task	Time			
		0	1	2	3
{2, 0, 2, 1}	0	0	2	0	0
	1	0	0	0	1
	2	2	0	2	1
{2, 0, 2, 0}	0	0	3	0	1
	1	0	0	0	1
	2	3	0	3	1
{2, 1, 2, 0}	0	0	3	0	2
	1	0	1	0	1
	2	4	0	4	1

6.6.9 Long-Term Memory (LTM)

Because the search is over, new restarts could be used with a new seed modified by the information given through the long-term memory.

6.6.9.1 LTM_Max

The maximum number in the last frequency matrix is 4 that is repeated for both combination of (task 2-time 0) and combination of (task 2-time 2). This indicates that these combinations were repeated in the candidate lists the most number of times. A column-wise strategy is used to pick the first minimum (task 2-time 0). Following LTM_Max rule, a whole new restart for the search process can be performed based on a new seed. The new seed will be a modified solution of the initial seed of the last iteration {2, 0, 2, 2} except that the underlined 2 shows that this element will be fixed throughout all new generation of neighborhoods.

6.6.9.2 LTM_Min

The process will be the same as LTM_Max except that the *minimum* number in the matrix is selected to be fixed. In this example, there are several cells with zero magnitude that show none of those task-time combinations have been suggested by the candidate lists. The first cell with the smallest number (column-wise) is picked and will be fixed. So the new seed will be {0, 0, 2, 2} and the underlined 0 shows that this element will remain fixed throughout the generation of solutions.

6.6.10New Restart

For this example, an LTM_Min strategy is used which results in {0, 0, 2, 2} as the initial seed. Therefore, the new neighborhood should be generated and evaluated by perturbing on the tasks at each time unit, yet one time unit at a time. The following tables show the process of generating these solutions. Note: tabu list, candidate list, index list, number of iterations, and number of iterations with no improvement is reset to zero.

The solutions generated after the first restart is listed in Table 6.9. Those solutions that are crossed in that table were either tabu moves for which the solution values were inferior to the aspiration level, or they were solutions that already existed in the candidate list.

Table 6.9 Neighborhood solutions generated after the first restart

Iteration	Solution	Value	Time			
			0	1	2	3
1	Initial	968.25	0	0	2	2
	1	972.15	0	1	2	2
	2	972.97	0	2	2	2
	3	959.54	0	0	0	2
	4	963.85	0	0	1	2
	5	967.68	0	0	2	0
	6	969.55	0	0	2	1
2	0	972.97	0	2	2	2
	1	968.25	0	0	2	2
	2	972.15	0	1	2	2
	3	974.68	0	2	0	2
	4	976.71	0	2	1	2
	5	973.54	0	2	2	0
	6	974.27	0	2	2	1
3	0	976.71	0	2	1	2
	1	963.85	0	0	1	2
	2	962.87	0	1	1	2
	3	974.68	0	2	0	2
	4	972.97	0	2	2	2
	5	974.42	0	2	1	0
	6	971.85	0	2	1	1
4	0	974.68	0	2	0	2
	1	959.54	0	0	0	2
	2	965.72	0	1	0	2
	3	976.71	0	2	1	2
	4	972.97	0	2	2	2
	5	970.11	0	2	0	0
	6	973.12	0	2	0	1

Table 6.10 shows a summary of updated parameters for the candidate list solutions, and Table 6.11 shows the updated frequency matrix after all the iterations of the first restart.

Table 6.10 Summary of solutions for the four iterations after the first restart

Iteration	CL		tabu	IL	AL	
	Value	Solution		Solution	Value	Solution
0 (initial)	968.25	{0, 0, 2, 2}		{0, 0, 2, 2}	978.74	{2, 0, 2, 1}
1	972.97	{0, 2, 2, 2}	{x, 0, x, x}		978.74	{2, 0, 2, 1}
2	976.71	{0, 2, 1, 2}	{x, x, 2, x}		978.74	{2, 0, 2, 1}
3	974.68	{0, 2, 0, 2}	{x, x, 1, x}	{0, 2, 1, 2}	978.74	{2, 0, 2, 1}
4	973.12	{0, 2, 0, 1}	{x, x, x, 2}		978.74	{2, 0, 2, 1}

Table 6.11 Updated frequency matrix after all iterations of the first restart

Task	Time			
	0	1	2	3
0	4	3	2	2
1	0	1	1	2
2	4	4	5	4

The LTM_Min for the next restart would suggest the seed {1, 0, 2, 2}. The following table shows a summary of the updated parameters after evaluating the solutions generated from the new seed.

Table 6.12 Summary of solutions for the four iterations after the second restart

Iteration	CL		tabu	IL	AL	
	Value	Solution		Solution	Value	Solution
0 (initial)	973.21	{1, 0, 2, 2}		{1, 0, 2, 2}	978.74	{2, 0, 2, 1}
1	973.77	{1, 2, 2, 2}	{x, 0, x, x}		978.74	{2, 0, 2, 1}
2	977.77	{1, 2, 0, 2}	{x, x, 2, x}		978.74	{2, 0, 2, 1}
3	974.62	{1, 2, 0, 1}	{x, x, x, 2}	{1, 2, 0, 2}	978.74	{2, 0, 2, 1}
4	973.44	{1, 2, 0, 0}	{x, x, x, 1}		978.74	{2, 0, 2, 1}

Table 6.13 Updated frequency matrix after all iterations of the second restart

Task	Time			
	0	1	2	3
0	4	3	5	3
1	4	1	1	3
2	4	8	6	6

Because there were only two restarts required in this example, the search ends at this point. .

Table 6.14 shows that the best solution found is {2, 0, 2, 1} with the value of 978.74. The computer time for the example illustrated above was a fraction of a second. Using Hyper Lingo 4.0 (1998) software for solving this problem through implicit enumeration techniques proves that the solution found by tabu search is optimal. Tabu search heuristics, with the same steps that were explained in this chapter, are used in the next chapter to solve problems with a variety of difficulties. The structure of these problems, tabu search results and its computational

efficiency compared to the branch-and-bound used in Lingo are discussed in the next chapter.

Table 6.14 Summary of the best solutions found at different search steps

Search Number	The best solution in IL	Value
Initial solution configuration	{2, 0, 2, 1}	978.74
First long-term memory restart	{0, 2, 1, 2}	976.71
Second long-term memory restart	{1, 2, 0, 2}	977.77

6.7 PSEUDO-CODE AND FLOW CHART OF TABU SEARCH-BASED HEURISTIC

The following psuedo-code is adapted and modified from Subur (2000).

```
Determine the tabu search parameters
Generate the initial solution
Set the Aspiration Level Solution (ALS) to the initial solution
Set the Aspiration Level (AL) to the total weighted average SL of the initial solution
Initialize the Long-term Memory Matrix (LTM) to zero
Do
{
        Add the initial solution to the list of restart solutions
        Initialize the Tabu List (TL)
        Initialize the number of Iteration (IT)
        Initialize the number of Iteration without improvement (IT_IMP)
        Initialize the Candidate List (CL) and the Index List (IL)
        Admit the initial solution to the Candidate List (CL) and the Index List (IL)
        Evaluate the total weighted average SL of the initial solution
        Set the current seed to the initial solution
        Do
        {
                Number of Iterations (IT) is increased by 1
                Generate the neighborhood solutions by perturbing on the tasks of the current seed
                at each time unit, yet one time unit at a time.
                For each neighborhood solution generated from the current seed
                {
                        Evaluate the total weighted average SL
```

```
                If (move ∈ TL and AL is not satisfied)
                        Exclude the solution that results from the move
        }
        The best solution ← ∅
        Do
        {
                Identify the neighborhood solution that has the maximum total weighted
                average SL
                If (the neighborhood solution ∉ CL)
                {
                        The best solution ← the neighborhood solution
                        The best move ← the move that results in the neighborhood
                        solution
                }

        } while (the best solution = ∅)
        The next seed ← the best solution
        CL ← the best solution
        TL ← the best move
        If (the total weighted average SL of the best solution > AL)
                Update AL and ALS
        If (the current seed = local optimum)
        {
                IL ← the current seed
                Entries into IL is increased by 1
        }
        If (the next seed > the current seed)
                Iteration without improvement (IT_IMP) ← 0
        Else
                Iteration without improvement (IT_IMP) is increased by 1

        Update LTM matrix
        The current seed ← the next seed

}while (IT and IT_IMP and entries into IL have not reached the specified numbers)

Do
{
        Identify the next new restart solution by suing the LTM matrix (either
        intensification or diversification)

} while the new restart ∈ list of restart solutions
Next initial solution ← new restart solution

} while (the number of restart has not reached the specified number)
Terminate the search
Return ALS with the value of AL as the best solution found so far
```

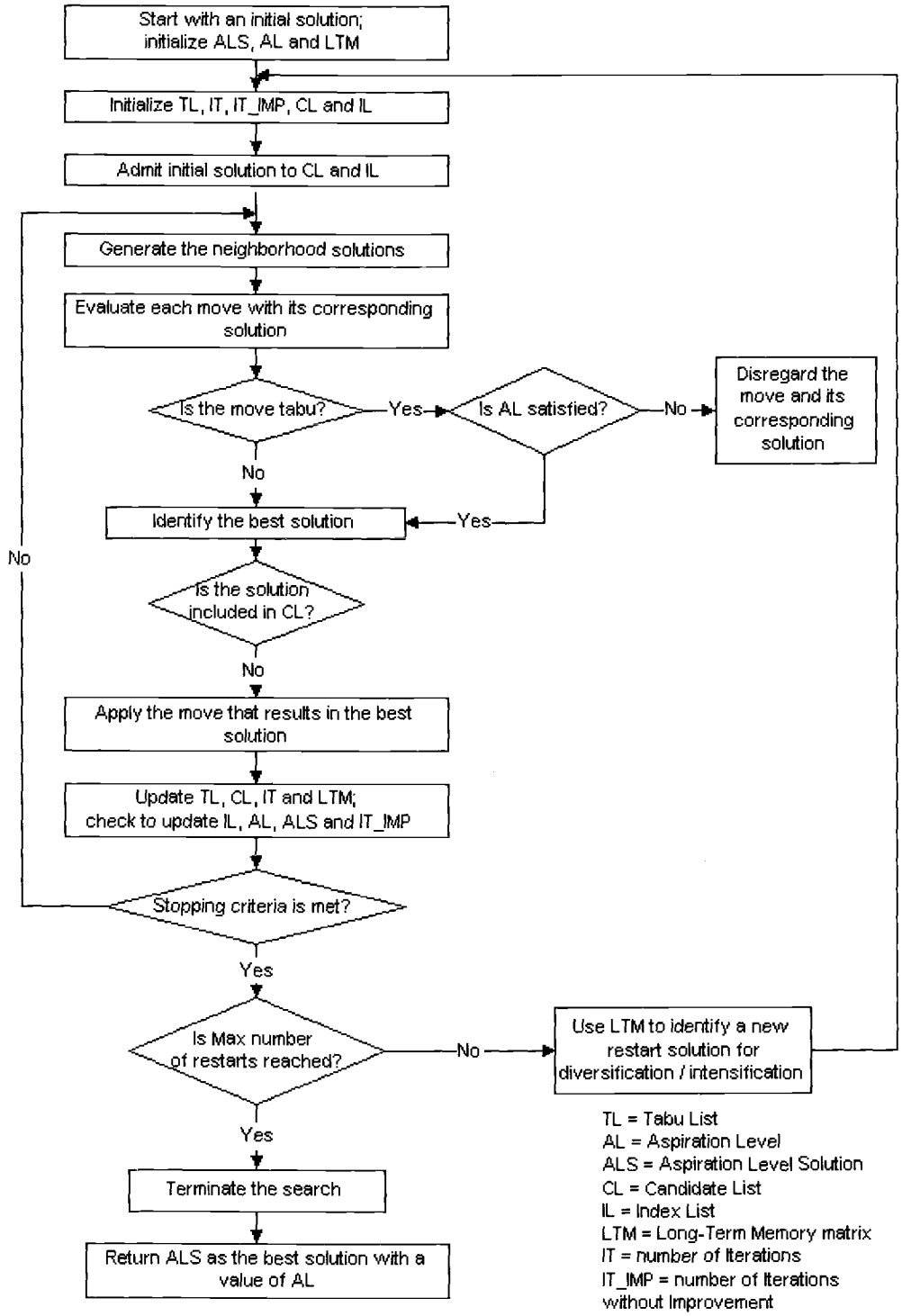


Figure 6.1 Flow chart of tabu search-based heuristic adapted and modified from Subur (2000)

CHAPTER 7: PERFORMANCE OF TABU SEARCH-BASED HEURISTIC

7.1 INTRODUCTION

The most desirable solution to any OR problem is the optimal solution. However, optimal solutions are not always practical to find. In many cases, it is actually impossible to find one within a reasonable time. The main obstacle on the way of finding optimality is the duration of time that it takes. It may take from several hours to several days or even weeks to find an optimal solution for a computationally complex problem (NP hard). Even small increases in the size of such problems may increase the time required for solving them by dozens of folds. Heuristic algorithms, such as tabu search, are used to overcome this problem. They will not guarantee an optimal solution, but they are expected to identify near-optimal solutions in a much shorter time. Therefore, the performance of such algorithms is measured with the quality of their solutions and their timeliness.

Measuring the time that an algorithm takes is not a difficult task. Measuring the solution quality, however, can be complicated. If the optimal solution could be found in a timely manner, the difference between this solution and the one found by the heuristic algorithm would determine the algorithm's precision. However, in cases where there is no known optimal solution, evaluating a good heuristic solution would take more than a simple subtraction. An upper bound (for maximization) has to be found to measure the proximity of the heuristic solution against it. An optimal solution must lie between the upper bound and the heuristic solution. Therefore, the difference between the two (without knowing the optimal) usually is a very good measure for the quality of a heuristic solution. In the following section, the performance (computational time and solution quality) of tabu search heuristic applied to this research is measured.

7.2 TABU SEARCH PERFORMANCE

To investigate tabu search performance, problems were divided into three different sizes of small, medium, and large. Refer to Section 7.3 for a complete description of these problems. In the following sections, the quality of the solutions generated by tabu search for this research, and the computational time efficiency of those solutions are discussed.

The computer used for the purpose of the study in this chapter was a Pentium 4, 1.7 GHz, 256 MB RAM, with Windows 2000 operating system. Tabu search was coded in the Microsoft Visual Basic 6.0 programming language. In order to find the optimal solutions using the branch-and-bound enumeration method, Hyper Lingo 4.0 (1998) computer software, hereafter called Lingo, was used. Any computational time reported in this chapter could significantly vary if performed on a computer with different hardware/software specifications.

7.2.1 Small Size Problems

The results showed that the performance of the tabu search-based heuristic is exemplary. For the small size problems where optimal solutions could be found by Lingo in a timely manner, the tabu search heuristic also found the optimal solution in a fraction of a second on all problems attempted. For the medium and large size problems, however, Lingo could not find optimal solutions in a timely fashion. Therefore, there was no easy target against which to evaluate the performance of tabu search. For such problems, good upper bounds had to be found to measure how well the tabu search-based heuristic has performed.

7.2.2 Medium Size Problems

Unlike small size problems for which an optimal solution could be found in a matter of seconds, medium size problems could take several hours long. Most

initial experimental cases took so long that it seemed unreasonable to wait for the optimal solution. An arbitrary time length of 2.5 hours (researcher's patience) was determined as a cut-off point for Lingo to find an optimal solution for four experimental problems. All four problems were carefully constructed in the lower end of the medium size problems. The lower end of medium size problems was selected because it was expected that Lingo could find an optimal solution within a reasonable computation time. The initial experiments showed that the closer the initial satisfaction level is to zero, the harder the problem becomes. This is caused by the introduction of new binary variables, in the mathematical model, to penalize zero satisfaction levels. Therefore, the experiment consisted of problems with initial satisfaction levels both above and below 50% to represent a more general case.

The following table shows the range of parameters for the medium size problems. Abiding by the following ranges, four distinct problems were selected that represented lower end of the range available for number of tasks and planning horizon. Also abiding by the range, two of the satisfaction levels were selected above 50% and two below.

Table 7.1 Specifications of medium size problems

Medium Size Problems	
Parameter	**A Number in the Interval of**
# of tasks	[5, 8], integer value
Planning horizon (time units)	[5, 10], integer value
Initial satisfaction level for all tasks	[0, 1], up to 2 decimal points

Table 7.2 shows how the data was generated for the parameters of each task in all problems. Table 7.3 shows the parameters generated for all four problems.

Table 7.2 The rule underlying generating parameters of each task for medium size problems

Parameter	A Number Generated from a Random Uniform Distribution in the Interval of
Weight	[1, 10], integer value
Deviation rate	[0.01, 0.10] up to 2 decimal points
Correction rate	[1 to 10] multiplied by the corresponding deviation rate

Lingo could not find an optimal solution for any of these problems within a 2.5-hour time limit. Each one of these problems were further relaxed by Lagrangian relaxation techniques and solved again by Lingo. The concept and implementation details of Lagrangian relaxation are discussed in the next section. For all the problems, Lingo found a solution within the same time limit and in fact much faster most of the time. These relaxed solutions (upper bounds) on average were less than 11% from the best solution found by the tabu algorithm (Table 7.4). If all the tasks start at perfect status (100% SL), and stay at perfect status throughout the experiment (i.e., deviation rate is zero), the objective function will be equal to 1. This is the intuitive upper bound. If the intuitive upper bound of *'1'* were used instead of the upper bounds obtained by Lagrangian relaxation, the tabu results would have resulted in a 31% average deviation. Therefore, upper bounds found by Lagrangian relaxation are almost three times superior to the intuitive bound, and that makes the time spent on implementing Lagranginan relaxation all the more

Table 7.3 Parameters generated for the four sample problems in medium size

		Problem Number						
Problem No:		1	2	3	4			
Tasks: [5, 8]		5	7	5	6			
Planning Horizon: [5, 10]		7	5	6	6			
Initial SL: [0,1]		0.90	0.80	0.35	0.40			
		Task						
		1	2	3	4	5	6	7
Problem No								
1	CR (1-10)×DR	0.48	0.09	0.21	0.15	0.14		
	DR (0.01-0.10)	0.08	0.03	0.07	0.03	0.02		
	Weight (1-10)	1	4	8	3	7		
2	CR (1-10)×DR	0.24	0.9	0.16	0.3	0.42	0.5	0.36
	DR (0.01-0.10)	0.04	0.09	0.02	0.1	0.06	0.05	0.04
	Weight (1-10)	3	6	6	1	4	4	10
3	CR (1-10)×DR	0.24	0.24	1	0.09	0.25		
	DR (0.01-0.10)	0.03	0.08	0.1	0.01	0.05		
	Weight (1-10)	6	9	9	6	10		
4	CR (1-10)×DR	0.08	0.1	0.36	0.18	0.56	0.28	
	DR (0.01-0.10)	0.08	0.01	0.09	0.02	0.08	0.04	
	Weight (1-10)	1	5	7	3	9	10	

worthwhile. A comparison summary between the upper bounds and optimal solutions can be found in Table 7.4.

Unlike tabu search, Lingo uses an implicit enumeration algorithm to find the optimal solution. This is a systematic evaluation of all possible solutions without explicitly evaluating all of them often called as branch-and-bound (Greenberg, 2002). One important research question is how well the tabu search-based heuristic performs compared to the implicit enumeration algorithm applied by Lingo to the original problem before relaxation. To answer this key question, there was a need to let Lingo run in order to identify an optimal solution. The initial cut-off time of 2.5 hours for Lingo was extended to more than 3 times for a total duration of 8 hours in the hope of finding an optimal solution. Preliminary experimentation also indicated that the 8-hour time limit would be reasonable for the lower end, medium size problems. In particular for the above four instances, Lingo consistently found an optimal solution in less than 8 hours. The average time taken to solve a problem optimally was 302 minutes. It took an average of 45 minutes for the relaxed version of these problems to find the optimal solution that is the upper bound in this case. This duration was 6.7 times shorter than the time taken for original problems to find a solution optimally before relaxing.

The computation time required for the tabu search-based algorithm is almost negligible. It took an average of 1.5 seconds for tabu search to find the best solution. In the absence of an optimal solution, this solution is only 10.63% from the upper bound found by relaxation. After letting Lingo run for 8 hours to find an optimal solution, it is determined that the tabu search-based solutions are in fact only less than 2% from the optimal. Thus, in the previous examples, tabu search is clearly a better alternative than implicit enumeration as it gives a result within 2% precision, but more than 9000 times faster! The following table provides a

summary of the above comparison. In the next section, Lagrangian relaxation technique used in this research is discussed in detail.

Table 7.4 Relative comparison between optimal, upper bounds, and tabu solutions

		Problem				
		1	2	3	4	Average
Deviation	(Opt. – Tabu) / Opt. %	0.24	0	5.62	0	**1.47**
	(Upp. - Tabu) / Upp. %	4.29	6.8	14.69	16.74	**10.63**
	(1 – Tabu) %	9.02	17.7	54.47	42.77	**30.99**
	(Opt. - Upp.) / Opt. %	-4.24	-7.29	-10.63	-20.1	**-10.57**
Time (Sec)	Tabu	1	2	1	2	**1.5**
Lingo Time (Min)	Optimal	448.17	172.92	212.73	372.5	**301.58**
	Relaxed	11.97	25.87	18.17	123.9	**44.98**
Time Ratio	Opt. / Upp.	37.45	6.68	11.71	3.01	**6.71**

7.2.2.1 Lagrangian Relaxation: Application

Lagrangian relaxation is a theoretical concept that is widely used in practice for finding upper (lower) bounds to typically complex integer-programming problems (Fisher 1985, Fisher 1981). The concept is based upon replacing a set of difficult constraints by a penalty term in the objective function. This penalty or punishment is accounted for by assigning a unique multiplier to the amount of violation of each of those constraints and their dual variables. Major challenges for this technique are: 1) which constraints to choose? 2) how to compute good multipliers?

Generally, the constraints selected for relaxing should make the problem significantly easier without losing the essential properties of the original model. The answer to question 2 is a general-purpose procedure called 'subgradient

method.' This method involves a series of iterations that use the solution to the relaxed problem to find new penalty multipliers, and uses the new multipliers to define a new relaxed problem. The cycle is repeated until the solution to the relaxed problem converges to a number and becomes highly insensitive to the change of multipliers. The closer is this number to the best solution found by alternative heuristics, in this case tabu search, the more desirable is the upper bound. Sometimes when an upper bound is within a predetermined proximity (e.g., 5%) of the heuristic result, the search for a better upper bound stops.

The choice of constraints, the starting values for multipliers, and the step size in iterations of the subgradient procedure are as much art as they are science. A bad choice of values for any of the above items can result in upper bounds that are exceedingly far from the heuristic solutions or good upper bounds that take immensely long time. It usually takes several trials before finding the right balance.

For the above four problems, constraint 1 (Refer to the model in Chapter 5.) was the one that was finally selected for relaxation. This constraint enforces that only one task at a time could be attended to. So violation of this constraint could mean attending to more than one task at any point in time. Relaxing constraint 1 resulted in identifying good upper bounds within a reasonable time. Relaxation of constraint 3 did make the problem significantly easier, but the upper bounds identified were too far off. This poor result is attributed to M (a very large number) as one of the multipliers in constraint 3. Relaxing this constraint means punishing it for the amount of violation with its dual variable in the objective function. A penalty term with a very large multiplier (M in this case), suppresses the rest of the multipliers in the objective function. This, in consequence, leads to poor results.

The equality constraint 2 was relaxed in three different fashions: two of them involved substituting the equality constraint with two inequalities, relaxing one

inequality (greater than or less than) and leaving the other one in the model. The last method was to relax the equality constraint treating it as a strict equality constraint. The only difference when relaxing an equality constraint (versus inequality) is the penalty multipliers do not have to be non-negative integers. None of the three different trials were helpful. These relaxations either changed the structure of the model completely and resulted in solutions of no value or took a very long time to find an acceptable upper bound. Thus, constraint 1 was the only promising constraint for relaxation. It should be noted that this constraint is in fact a matrix of constraints with T (planning horizon) rows.

As noted earlier, the relaxation of constraint 1 resulted in upper bounds that gave the tabu solutions an average deviation of 10.63%. For the four problems, the average percentage deviation between the upper bounds found by relaxing constraint 1 and the optimal solution is 10.57%. The deviation comparisons are shown in the Table 7.4.

7.2.2.2 Lagrangian Relaxation: Technical Details

Throughout this section, bold letters are used to indicate matrices or vectors. Consider the following problem:

$Z = \max \mathbf{cx}$

Subject to:

$\mathbf{A x} \leq \mathbf{b}$

$\mathbf{D x} \leq \mathbf{e}$

$\mathbf{x} \geq 0$ and integral

where x is n*1, b is m*1, e is k*1 and all other matrices have comfortable dimensions.

If we were to relax constraints $Ax \leq b$, the above problem would result in:

$Z_D(\mathbf{u}) = \max \mathbf{cx} + \mathbf{u}\,(\mathbf{b} - \mathbf{Ax})$

Subject to:

$\mathbf{Dx} \leq \mathbf{e}$

$\mathbf{x} \geq 0$ and integral

where $\mathbf{u}$ is an m vector of non-negative multipliers.

Any answer to $Z_D(\mathbf{u})$ will be an upper bound to Z. Ideally, the most desirable u should solve the following problem:

$Z_D = \min Z_D(\mathbf{u}),\ \mathbf{u} \geq 0.$

In order to find the tightest upper bound, a set of iterations should be followed through. A very tight upper bound may require completion of many cycles of the algorithm and in turn be time consuming. A compromise should be made between precision and the time that it takes.

The following are the steps for a generic Lagrangian relaxation algorithm.

1) $\mathbf{u}^0 = 0,\ k = 0.$
2) Assign an arbitrary number to step size (t_k).
3) $\mathbf{u}^{k+1} = \max\{0,\ \mathbf{u}^k - \mathbf{t}_k(\mathbf{b} - \mathbf{A}.\boldsymbol{x}^k)\}$. Revise multipliers for the set of relaxed constraints.
4) Solve the relaxed problem ($LR_{\mathbf{u}}^k$) with new multipliers (u) and find the new upper bound $Z_D(\mathbf{u})$.
5) If the iteration limit is reached, or the new upper bound is lower than the best answer found so far for the primal problem (Z*), STOP and quit the algorithm.

6) If the solution to relaxed problem ($\mathbf{x}^k$) is feasible in primal problem, update the best answer found so far for primal problem (Z*).
7) Increase k by 1.
8) Go to step 2.

For further details of the above procedure, refer to Fisher (1985).

There are methods suggested in literature for choosing an appropriate step size and changing it between iterations in the above procedure. However by experiment, it was found for the above four problems, not changing the step size across the iterations gives the best result. The initial selection of the step size for each problem was arbitrary and based on trial and error. No change was made to the step size in subsequent iterations. This can be seen in Table 7.5.

The third step of the above algorithm refers to the constraints in the original model that were decided to be relaxed. In the four examples discussed before, constraint (1) was relaxed. Thus, the third step of the above algorithm would translate to:

$$\mathbf{u}^{k+1} = \max\{0, \mathbf{u}^k - \mathbf{t}_k(1 - \sum_{i=1}^{n} x_{i,t})\}, t = 0, 1, \ldots, T\text{-}1$$

where n = number of tasks, T = planning horizon, $\mathbf{t}_k$ = step-size for iteration k,
t = index for time, and $\mathbf{u}^k$ = T vector of multipliers for iteration k of the set of constraints 1.

Table 7.5 shows the performance of Lagrangian relaxation compared to optimal for the four problems experimented.

Table 7.5 Relative comparison between optimal and upper bound solutions found by Lagrangian relaxation in different steps

Problem	Constraint 1 Lagrangian Relaxation				
	Solution	Time (min)	t	Objective Function	Deviation from Optimal (Optimal – Lagrangian)/Optimal
1	Optimal	448.17		0.9120	0%
	Lagrangian	0		0.9873	-8.26%
		6.63	0.005	0.9518	-4.37%
		5.30	0.005	0.9506	**-4.24%**
		2.55	0.005	0.9531	-4.51%
2	Optimal	172.92		0.8230	0%
	Lagrangian	0		0.9655	-17.31%
		25.87	0.005	0.8830	**-7.29%**
		38.12	0.005	0.8766	-6.50%
3	Optimal	212.73		0.4824	0%
	Lagrangian	9.13		0.8134	-68.60%
		1.72	0.010	0.5936	-23.05%
		7.32	0.010	0.5337	**-10.63%**
4	Optimal	372.5		0.5723	0%
	Lagrangian	37.93		0.8440	-47.48%
		25.97	0.010	0.6873	**-20.10%**
		57.23	0.010	0.6312	-10.29%
		21.85	0.010	0.6138	-7.25%

The main motive for any relaxation is not finding the optimal solution in a timely manner. The only reason that the optimal solutions are mentioned in the above table is merely to evaluate how well the Lagrangian relaxation has performed, and how much time it has saved compared to finding the optimal.

7.2.3 Large Size Problems

As it was mentioned earlier, the performance of tabu search is measured in terms of its quality and its timeliness. To measure the quality of a heuristic solution, knowing the optimal solution or in its absence an upper bound is vital. For this category of problems, it was practically impossible to find an optimal solution in a timely fashion. Even an attempt to find an upper bound would take unreasonably long time for these problems. A relaxed version of the original problem, whose solution would be an upper bound, is still a very difficult problem to solve in this category. Therefore, there is no optimal solution or upper bound available to measure the quality of tabu search-based heuristic solutions for large size problems. However, all different combinations of tabu algorithm and initial solutions were compared against each other to identify which combination yields the best quality solution. Similar comparisons were performed on small and medium size problems. The timeliness of tabu heuristics, on the other hand, is measurable and proves to be very good. It took less than a minute for tabu search to give its best solution for all problems experimented in the large size.

7.3 DESIGN OF EXPERIMENT

7.3.1 Treatments

To have a better understanding of the tabu search performance and the factors impacting it, problems experimented with were divided into three different sizes. These different sizes define a clear distinction for computational complexity of problems under investigation. The following table shows the structure for the small, medium and large size problems.

Table 7.6 Specifications of small, medium, and large problem sizes

Problem Size	Optimal Solution Through Implicit Enumeration (Lingo)	A Number Generated from a Random Uniform Distribution in the Interval of	
		# of Tasks (Integer)	Planning Horizon (Integer Time Units)
Small	Found in a few seconds	[2, 3]	[2, 4]
Medium	Found in a few hours for some cases	[5, 8]	[5, 10]
Large	Impractical to find in a reasonable time	[12, 15]	[11, 20]

For each one of the above categories of problems, a few research questions have to be addressed. In particular, it is of interest to know if any of the following treatments has any effect on the quality of the best solution found by tabu search and its timeliness.

1) The significance of the two different initial solution generation methods: uniform random and weighted random.
    a. H_0: There is no difference in the score gained by tabu search using uniform random versus weighted random initial solution finding mechanisms.
    b. H_1: There is a difference in the score gained by tabu search using uniform random versus weighted random initial solution finding mechanisms.

2) The significance of the three different tabu search-based heuristics: Fixed-no-long-term, Fixed-LTM-Max, and Fixed-LTM-Min.
    a. H_0: There is no difference in the score gained by tabu search using Fixed-no-long-term, Fixed-LTM-Max, or Fixed-LTM-Min.
    b. H_1: At least one of the three methods produce different results than the other two.

3) The significance of the interaction between the above two treatments which constitutes for a total of $3 \times 2 = 6$ levels.
    a. H_0: There is no interaction between initial solution finding mechanisms and tabu search methods.
    b. H_1: There is interaction between initial solution finding mechanisms and tabu search methods.

To address the above research questions, a special multi-factorial experimental design, called split-plot design, is used. Initial experimentations implied no apparent difference in computation time between the above treatments. Therefore, only one performance measure is used: quality of the best solution found by tabu search heuristic.

For this experimental design, two factors are defined. They are 1) initial solution generation method and 2) tabu search-based heuristic. The first factor has two different levels: a. uniform random and b. weighted random. Uniform random means assigning a random schedule of attending to tasks for the initial solution. This schedule is obtained by generating an integer number (task number) from a random uniform distribution at each time unit. The second level, weighted random, uses a ratio $w_i(\frac{DR_i}{CR_i})$ to determine the chances of a task being selected at each time unit. The random number is generated so that at any time unit a task *i* has a chance

of $\frac{w_i(\frac{DR_i}{CR_i})}{\sum_j w_j(\frac{DR_j}{CR_j})}$ to be attended to. The rationale behind this method of generating initial solutions is explained in detail in Section 6.3.2.

The second factor, the tabu search-based heuristic method, has three different levels. All three levels use a fixed tabu list size. However, if long-term memory is activated, they differ in how it is used. The first one does not use the long-term memory at all; the second one uses the long-term memory based on maximal frequency. This means that the heuristic deepens the search around the solutions that have been suggested by tabu the most number of times. This type of search is also called intensification. The third level employs the long-term memory based on minimal frequency. In this search, the heuristic diversifies the search by searching around the solutions that have been looked at by tabu the least number of times.

Initial experimentation showed no significance for variable tabu list size. Therefore, treatment levels for tabu search method did not include 'variable tabu list size' in the treatments summarized in Table 7.7.

Table 7.7 Treatments and their levels in the design of experiment

Factor or Treatment	Treatment Levels
Initial solution generation method	Uniform random
	Weighted random
Tabu search method	Fixed tabu list size and no long-term memory
	Fixed tabu list size and long-term memory based on maximal frequency Max_LTM
	Fixed tabu list size and long-term memory based on minimal frequency Min_LTM

7.3.2 Sample Size

Different formulas are available for finding the appropriate sample size in the design of experiments textbooks. These formulas are all based on the difference that needs to be detected, the variation in the data, and the risk that can be tolerated (Hicks and Turner, 1999). When a researcher deals with simpler cases, these formulas are clear such as the ones for comparing the means of two populations with equal variance, a completely randomized single-factor experiment, or a two-factor factorial design. However, there is no simple formula to find the sample size for more complicated designs such as a split-plot design used in this research.

Hicks and Turner (1999) state that the choice of sample size in practice is usually arbitrary. When it is not possible to determine the sample size accurately, one is advised to take as large a sample as possible to detect smaller differences in means with less risk. Although two or three replications (blocks) is recommended for similar problems in the textbooks (Hicks and Turner, 1999; Montgomery, 2001), a conservative sample size of ten was selected for the purpose of this experiment. This was due to negligible costs associated with time and/or effort to take samples on the computer.

7.3.3 Split-Plot Design

To conduct this experiment, ten problem instances within each problem size were created. The procedure for creating such instances is described in Appendix E. Each problem instance is different in structure that is characterized by the number of tasks, planning horizon, initial satisfaction level, correction rates, deviation rates, and weights. This can cause significant variation in results of the treatment factors among problem instances. Blocking is a technique used to reduce the variation between problem instances. Each problem instance is considered a block. Thus, the significance of the treatment factors, if any, can be attributed to the treatments and not to the difference among problem instances.

For each block, all the six (two levels of initial solution and three levels of tabu search) combinations of both factors are tested. If the six treatments were independent or experimented randomly, the design would have been a randomized complete block design. However, to remove the influence of the differences between the initial solutions produced by the same initial solution finding mechanism, a refined design called split-plot is used. First, an initial solution is generated by one of the two procedures (levels) and then three levels of tabu search are applied to that initial solution. This is different from generating three different initial solutions for three different tabu search heuristics, which is a requirement of a randomized complete block design. Thus, the design in this research is a special form of block design called split-plot design. For this purpose, initial solution is the whole plot treatment and tabu search heuristic is the subplot treatment. The subplot treatment is usually the factor of maximum interest to the researcher, which is also the case in this design. The following picture shows the sketch of this split plot design. For further details on block designs and split designs, refer to the textbook by Montgomery (2001).

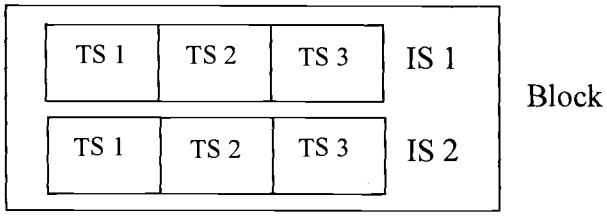


Figure 7.1 Split-plot design with Initial Solution (IS) as the whole plot and Tabu Search (TS) method as subplot

7.4 EXPERIMENTAL RESULTS AND ANALYSIS

The small, medium, and large size problems generated for the purpose of this experiment are shown in Appendix E.

7.4.1 ANOVA Tables for Small, Medium, and Large Size Problems

The following sections contain the ANOVA tables (Table 7.8-Table 7.10) investigating the significance of tabu search methods (three levels) and initial solution finding mechanisms (two levels) and their interaction in the quality of the final best solution found by the tabu search heuristic. None of the ANOVA tables for small or medium size problems suggests any significance of the primary effect of treatments or their interactions. The ANOVA table for large size problems (Table 7.10), however, indicates that there is a significant difference between the levels of tabu search methods.

Table 7.8 The ANOVA table for small size problems

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Whole plot:					
block	9	3163904.78	351544.976	.	.
seed	1	0	0	.	.
block × seed (whole plot error)	9	0	0	.	.

Table 7.8 (Continued) The ANOVA table for small size problems

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Sub plot:					
tabu	2	0	0	.	.
block × tabu	18	0	0	.	.
tabu × seed	2	0	0	.	.
block × tabu × seed (subplot error)	18	0	0	.	.
Total (corrected)	59	3163904.78			

Table 7.9 The ANOVA table for medium size problems

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Whole plot:					
block	9	3153806.56	350422.951	.	.
seed	1	92.405	92.405	1.5419	0.24572
block × seed (whole plot error)	9	539.368	59.93	.	.
Sub plot:					
tabu	2	1.22	0.61	2.2932	0.12966
block × tabu	18	4.793	0.266	.	.
tabu × seed	2	1.387	0.693	2.2871	0.13030
block × tabu × seed (subplot error)	18	5.453	0.303	.	.
Total (corrected)	59	3154451.19			

Table 7.10 The ANOVA table for large size problems

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Whole plot:					
block	9	2234603.32	248289.258	.	.
seed	1	104.017	104.017	1.3199	0.2802
block*seed (whole plot error)	9	709.258	78.806	.	.
Sub plot:					
tabu	2	67.264	33.632	3.5978	**0.0485**
block × tabu	18	168.265	9.348	.	.
tabu × seed	2	8.368	4.184	1.6414	0.2214
block × tabu × seed (subplot error)	18	45.878	2.549	.	.
Total (corrected)	59	2235706.37			

The Student Newman Keuls (SNK) Multiple-Range Test

Among the small, medium, and large size problems, only 'tabu' factor in the large size problems (Table 7.10) had significant difference between its levels, P-value = 0.0485. Therefore, the new question to answer is which of the three levels of tabu methods is significantly different from the others. To answer this question, a Student Newman Keuls (SNK) multiple-range test is used. This test has smaller type I (alpha) error than the error in Duncan's test. For this reason, the power of this test is generally lower than Duncan's, and it is harder in the SNK test to declare significant difference between a pair of means. In contrast, the SNK test is less conservative than Tukey procedure and has the potential to find more differences

conservative than Tukey procedure and has the potential to find more differences between treatments (Weber and Skillings, 2000). The following calculations were performed by the SAS statistical software (Release 8.02). For further information on the steps of the SNK test and the application of the SAS software to this test, refer to the textbook by Hicks and Turner (1999).

The SNK criteria: $SNK(k,\alpha_E) = q_{\alpha,k,\nu}\sqrt{\frac{s^2}{r}}$

where

$q_{\alpha,k,\nu}$ is the Studentized range statistic

k is the number of means, in this case 3 for three levels of tabu methods.

ν is the degree of freedom for the estimate of the experimental error. In this case, $block \times tabu$ is used as the experimental error for assessing the significance of $tabu$. Therefore, it is 18.

s^2 is the experimental error, in this case 9.348.

r is the number of observations used to calculate each mean, in this case 60/3 = 20.

α_E usually an experimental error rate equal to 0.05.

First, means of each level has to be ranked as follows.

Table 7.11 Ranking of different tabu search methods based on their mean

Level	Tabu Method Mean
1	406.2435
3	407.5515
2	408.837

Then, using the equation provided earlier, the critical value in Table 7.12 is populated. Using the critical value, the comparison results between different tabu methods are shown in Table 7.13.

Table 7.12 The SNK test critical value for different tabu methods

k	2	3
$q_{0.05,k,18}$	2.97	3.61
$SNK(k,0.05)$	2.0305	2.4680

Table 7.13 The SNK test comparison between different tabu methods

Tabu Method	**Tabu Method Mean**	**Tabu Method**			k	$SNK(k,0.05)$
		1	3	2		
		406.2435	407.5515	408.8370		
1	406.2435		1.3080	2.5935*	3	2.4680
3	407.5515			1.2855	2	2.0305
2	408.8370					

Note: dark and light gray cells distinguish different levels of comparison. Lighter cells are comparisons between means that are adjacent in ranking (i.e. comparing Tabu 1 to Tabu 3, and Tabu 3 to Tabu 2). The darker cell is the comparison between means that are 2 ranks apart (i.e. comparing Tabu 1 to Tabu 2).

It can be seen from the table that Tabu 1 and Tabu 2 means are 2.5935 units different, which exceeds the critical value of $SNK(3,0.05) = 2.4680$, noted by an

asterisk. Therefore, it can be concluded that there is significant difference between the mean of Tabu 1 (Fixed tabu list size, no long-term) and the mean of Tabu 2 (Fixed tabu list size, LTM_Max) with P-value < 0.05. To ensure the reliability of the test, the normal probability plot of the residuals in the large size is provided in Figure 7.2. Since the plot is quite normal, the assumption of normality for the residuals is not violated and the final conclusion is valid.

Since the differences in means do not exceed their critical value for the other comparisons, no significant difference between them can be concluded. For this reason a normal probability plot for the small or medium size was not needed. Thus, the only conclusion made with 95% confidence is that the 'Fixed size, LTM_Max' method performed better than the 'Fixed size, no-long-term' method in the large size categories.

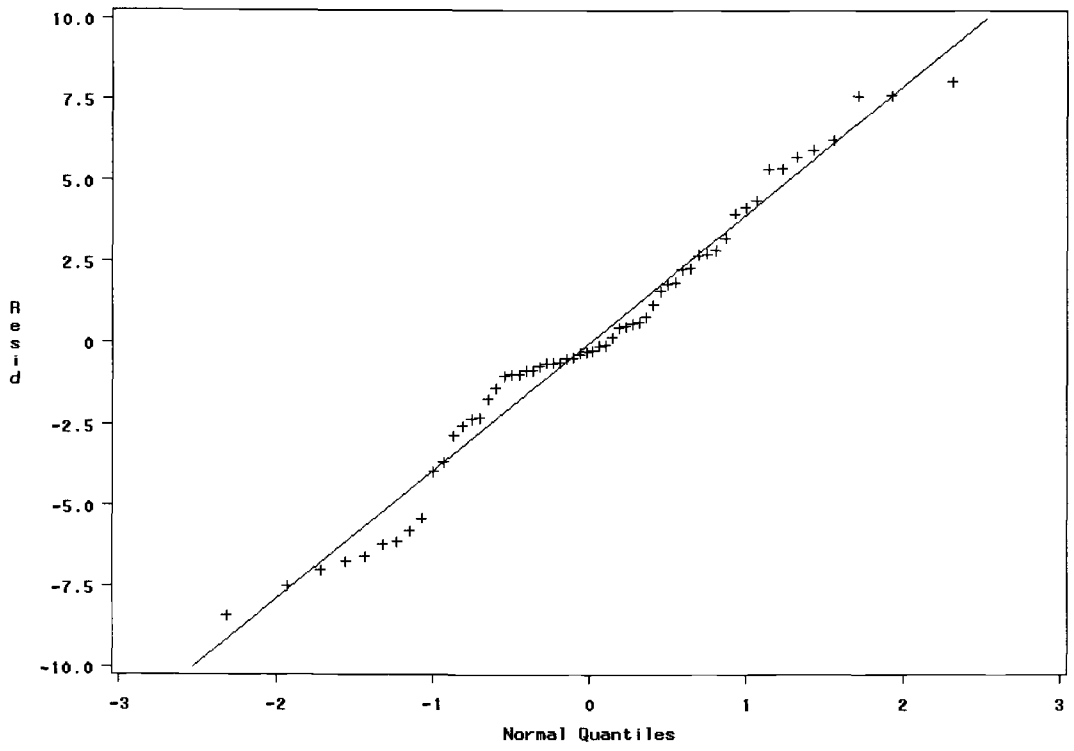


Figure 7.2 Normal probability plot for the residuals in the large size

7.4.2 Very Large Size Problems

Previously three categories of small, medium, and large size problems were described based on their difficulty in finding an optimal solution or an upper bound. In small size problems, optimal solutions were found in no time; in medium size problems, they took several hours; and in large problems, it was not even attempted to find the optimal solution since it could have taken days or weeks. In none of the three problem sizes, however, finding the best solution with tabu search was an issue. Tabu search gave its best solution in no time for small size problems, and in a few seconds for large size problems. This will change in the category of

very large size problems in which tabu search could take hours, if not days or weeks to find the best solution.

There is no limit to increasing the size of the problems in this research by increasing the number of tasks and/or planning horizon. Unless there is a motivation for doing so, one can engage himself in a never ending race. The motivation behind very large size problems, in this case, was a need to solve problems for a relatively very large planning horizon, and compare the solution so obtained with human performance. The experiments with human subjects that will be discussed in the next chapter involve a person playing with a computer game consisting of six tasks for five minutes. The game allows a person to switch his/her attention every tenth of a second if he/she wants to, so the time unit will be tenth of a second in that environment. Therefore, five minutes will be equivalent to 3000 time units. Even to compare the subject's score with the near-optimal score for only ten seconds, one has to solve a mathematical model with a planning horizon of 100 time units, which is a very large problem. Therefore, the category of very large size problems was developed to address a very specific need.

In this category, number of tasks is fixed at 6, but the planning horizon should be at least 100 time units. Therefore, what makes the problem computationally demanding is not the number of tasks, because a six-task problem falls in the category of medium size problems (Table 7.6). It is the planning horizon that is far higher than the largest planning horizon previously considered, which is 20 time units in the large size category (Table 7.6). The following table shows characteristics of very large size problems that have been attempted in this experimentation.

Table 7.14 Specifications of very large size problems

Very Large Size Problems	
Parameter	**Number**
Number of tasks	6
Planning horizon (time units)	Larger than 100
Initial satisfaction level for all tasks	0.50

If the parameters of the tabu search is properly set to allow a reasonable search for the best solution, it will take several hours or even days to solve a problem with T = 3000 time unit planning horizon to compare with five minutes of human performance. One alternative way around this would be to rescale the time unit to one second instead of tenth of a second. If this is done successfully, the problem can be solved for T = 300 time units to capture a five-minute period, which is lot easier and faster. In order to do so, parameters of the system should be adjusted with the assumption that a task can be attended to every second instead of 0.10 of a second. Table 7.15 shows how the parameters of a problem can be transformed to scale down the planning horizon.

When the transformed problem is solved with tabu search-based heuristics, the order of attendance can be conjectured to the actual game, having in mind that when a task is attended to it has to be for at least one second. That is, if the transformed problem suggests that a task should be attended for one time unit (one second), in the actual game environment this task has to be attended for ten time units (10×0.10 of a second). This conjecturing technique proves to be very effective in estimating a near-optimal order of attendance. The following tables consider smaller size problems to show that the conjectured solution is very close to the solution of the actual problem if it was solved directly. Moreover, it shows

that the time taken for finding a conjectured solution is far less than solving the problem directly. On average, a conjectured solution took 0.6% of the time it took to solve the same problem directly using tabu search.

Table 7.15 Original and transformed parameters for the conjecturing technique

Task	Actual Problem with T = 3000			Transformed Problem with T = 300		
	CR	DR	Weight	CR	DR	Weight
1	9	2	6	90	20	6
2	5	1	5	50	10	5
3	8	3	10	80	30	10
4	8	3	6	80	30	6
5	9	4	4	90	40	4
6	13	2	2	130	20	2

When the transformed problem is solved with tabu search-based heuristics, the order of attendance can be conjectured to the actual game, having in mind that when a task is attended to it has to be for at least one second. That is, if the transformed problem suggests that a task should be attended for one time unit (one second), in the actual game environment this task has to be attended for ten time units (0.10 of a second). This conjecturing technique proves to be very effective in estimating a near-optimal order of attendance.

Table 7.16 considers smaller size problems to show that the conjectured solution is very close to the solution of the actual problem if it was solved directly. Moreover, it shows that the time taken for finding a conjectured solution is far less than solving the problem directly. On average, a conjectured solution took 0.6% of the time it took to solve the same problem directly using tabu search.

Table 7.16 Comparison between transformed, conjectured, and actual solutions

Scenario	Transformed T = 20	Time Taken	Conjectured T = 200	Actual T = 200	Time Taken
1	400	0:00:06	400	400	0:09:56
2	458.57	0:00:08	458.58	459.52	1:00:25
3	422.77	0:00:10	423.48	426.88	1:08:50
4	489.29	0:00:10	489.59	501.37	1:03:20
5	524.32	0:00:16	524.96	522.07	0:48:08
Scenario	**Transformed T = 25**	**Time Taken**	**Conjectured T = 250**	**Actual T = 250**	**Time Taken**
1	375.12	0:00:12	374.23	375	0:15:35
2	448.2	0:00:15	448.26	448.75	1:39:30
3	399.61	0:00:31	400.74	401.98	1:39:02
4	481.91	0:01:42	482.76	482.06	1:12:03
5	521.55	0:00:15	521.41	514.73	1:19:32
Scenario	**Transformed T = 30**	**Time Taken**	**Conjectured T = 300**	**Actual T = 300**	**Time Taken**
1	350	0:00:13	349.99	350	0:22:31
2	436.82	0:02:30	436.69	437.05	2:39:59
3	376.45	0:00:18	376.79	369.58	1:53:17
4	474.57	0:00:31	473.86	478.14	2:23:56
5	524.83	0:00:24	524.97	527.28	3:19:40

Although from Figure 7.3 it seems evident that the conjectured solutions are almost as good as direct solutions, the following statistical analysis is performed to prove it.

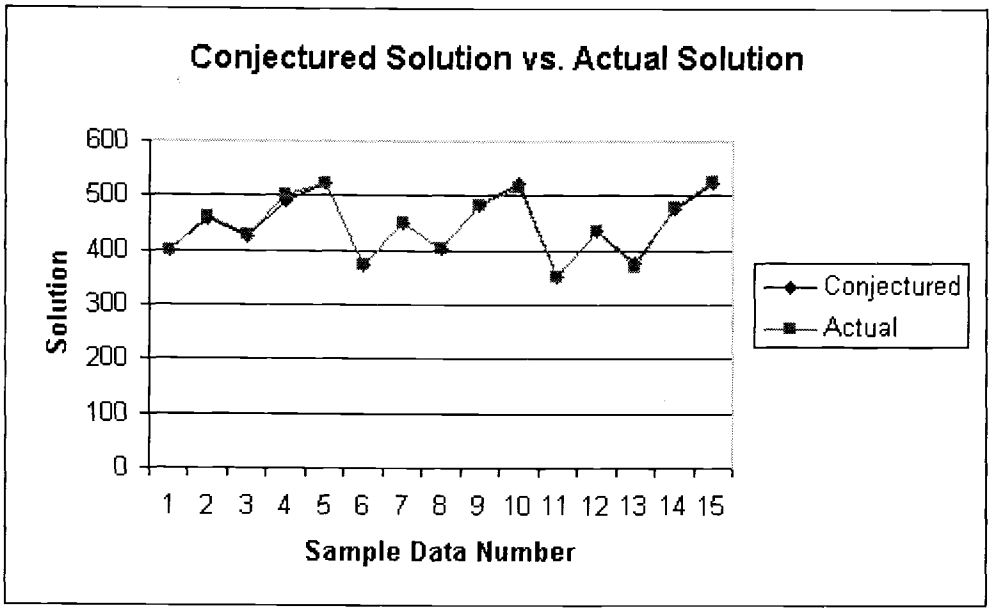


Figure 7.3 Comparison between conjectured and actual solutions

Statistical Analysis

To find out if there is a difference between the conjectured results and direct results, a paired t-test is performed for all the data combined (15 sample pairs). The hypotheses for this test are:

H_0: The mean difference between a conjectured score and a score gained directly by tabu search, for all data combined, is zero or

$$\mu_{\text{difference}} = 0$$

H_1: The mean of the score difference between conjectured and direct method is not zero or

$$\mu_{\text{difference}} \neq 0$$

The mean and the median for the combined sample data is:

Sample mean = - 0.54

Sample median = - 0.49

Then the following is calculated for the pair-wise t-test:

Computed t-Statistic = - 0.4689

P-value = 0.646

The t-statistic and P-value indicate that with a 95% confidence there is no significant difference between the pairs. The t-test is based on normality assumption. Since the normality assumption is not very strong in this case, two non-parametric tests are also performed, which are less sensitive to presence of outliers and assumption of normality. These tests are: Sign and Signed Rank test. For further information on these tests refer to the textbook by Ramsey and Schafer (2002). The hypotheses for both of these tests are:

H_0: The difference between a conjectured score and a score gained directly, for all data combined, has a zero median or

$$\text{Median}_{\text{difference}} = 0$$

H_1: The median of the score difference between conjectured and direct method is not zero or

$$\text{Median}_{\text{difference}} \neq 0$$

Sign test is based on counting the values above and below the hypothesized median. The following is obtained for Sign test:

Large sample test statistic = 1.336

P-value = 0.181

Signed Ranked test, on the other hand, is based on comparing the average ranks of values above and below the hypothesized median. The following is calculated for Signed Rank test:

Large sample test statistic = 0.937

P-value = 0.348

The P-values of the non-parametric tests are also bigger than 0.05. Therefore, none of the three tests could reject the null hypothesis with 95% confidence. Thus, it can be concluded with 95% confidence that there is no difference between the sample pairs of data, or conjectured results are as good as results gained directly.

7.5 DISCUSSION

The objective of this chapter was to evaluate the performance of tabu search, and to investigate the influences of different tabu search methods, and different initial solution finding mechanisms in finding a good solution. For this purpose, problems were divided in three different sizes: small, medium, and large.

To compare the performance of tabu search against the optimal, there was a need to find optimal solutions for these problems. In small size problems, optimal solutions could be found in no time; only for some of the problems in the medium size, optimal solutions could be found in less than 8 hours; and for large size problems, it could have taken days or weeks to find optimal solutions, so it was not attempted.

Tabu search gave optimal solutions in no time in small size problems. For a few problems in the medium size range, although optimal answers existed, upper bounds were found by Lagrangian relaxation techniques. This was a good test to evaluate the tightness of the gap between the upper bound, optimal, and tabu search solution. For the problems tested, tabu search gave solutions within 2% of the

optimal, but more than 9000 times faster than optimal. Tabu search also gave solutions within 11% of the upper bound, but more than 1300 times faster than finding the upper bound. All these indicate that tabu search had a great performance in the medium size problems. Since it was not practical to find an optimal or upper bound for large size problems in a reasonable time, the quality of tabu search solutions for this category of problems is unknown. However, tabu search proved to be very time efficient for large size problems also, and gave its best solution in less than a minute.

To investigate the effect of different tabu search methods and different initial solution finding mechanisms on tabu solutions, a split-plot experiment was designed. The results proved that only in large size problems, a 'Fixed tabu size, LTM-Max' method performs significantly better than a 'Fixed tabu size, no long-term.' None of the other tabu methods or initial solution finding mechanisms, in no other scenario, proved to be significantly different.

The next chapter explains a situation in which human subjects have to play a game that has the same constraints and objective functions as the mathematical model, but with very large planning horizons. To compare the performance of the subject with a near-optimal solution, there was a need to solve these problems with tabu search. The relatively very large planning horizon of these problems makes them also very difficult to solve. Unlike small, medium, and large size problems in which tabu search could give its best solution in a fraction of a second up to a minute, in these problems, tabu search might take hours or days. A conjecturing method is offered that scales the problem down to a shorter planning horizon with adjusted parameters, solves it, and rescales the solution back up to the original planning horizon. For the sample data, this conjecturing technique proved to give equally as good results as if it was solved directly, but in a remarkable 0.6% of the time.

The next chapter will discuss an environment to measure and analyze human performance in a multi-tasking environment subject to the constraints and objective functions of the mathematical model discussed so far. Human performance will be compared to the near-optimal solution found by tabu search for that environment.

CHAPTER 8: PERFORMANCE OF HUMAN VERSUS TABU SEARCH IN MULTIPLE TASK MANAGEMENT

8.1 INTRODUCTION

In Chapter 4, a generic multi-tasking environment was analyzed and some of its parameters were defined. Using these parameters, this environment was mathematically modeled (Chapter 5), and solved by tabu search (Chapter 6-7). The solution found through this mathematical model for any set of parameters would be a near-optimal solution (standard of comparison) against which human performance could be compared. In this chapter, a task management environment is explained that allows the researcher to record and measure the performance of the human as an operator of this environment. A set of experiments were conducted for this purpose and the subject's performance were reviewed (replayed), analyzed through graphs, and compared with the corresponding near-optimal solution. The objective was to find out how far/close a human subject's performance was to the near-optimal solution, and to find out if there was any pattern of attention allocation that humans use. In the following sections, there is a detailed description of the task management environment, the structure of the experiments, their results and conclusions.

8.2 METHOD

The details of the experiment are discussed in the following paragraphs. This includes a description of participants and equipment (task management environment), how the experiment was performed, and its statistical design.

8.2.1 Participants

Ten subjects, two females and eight males but not selected based on gender, participated in this research. Subjects were invited to participate voluntarily through a verbal conversation. All subjects were informed that they were not getting paid/awarded for their participation. Due to the ease of accessibility, they were all OSU college students who were comfortable with the use of a mouse in a computer, the minimum qualification required. They all had a minimum of 8 hours per week experience in working with a computer with an average of 26 hours per week. Their age range was 21 to 32 years with an average of 25 years old. Only half of them drove a car with an average of 4 driving hours per week. The detailed specifications of the participants can be found in Appendix F.

8.2.2 Equipment (Task Management Environment)

8.2.2.1 Hardware

The experiment took place in an ordinary room with no distractions or noise beyond the level accepted in a library's quiet area. The essential devices used were an office desk, two chairs, desktop computer, monitor, mouse, pen and paper. The computer used was a Pentium II/300 MHz, 64 MB RAM, with Windows NT 4.0 operating system.

8.2.2.2 Software

A computer program named *Tardast*[1] (Persian for Juggler) was utilized for the purpose of this experiment. The software was coded in the Microsoft Visual Basic 6.0 programming language. A simplistic view of any multitasking environment could be compared to a *Tardast* (user of this software) who has several tasks on

[1] The later customized versions of Tardast used at Oregon State University are named TME (Task Management Environment).

hand and tries to perform satisfactorily in all those tasks. This software served as a low-fidelity multitasking experimental environment in which the behavior of the tasks (deterioration/correction rates from/to their satisfactory status), importance of the tasks, and the duration of the experiment could be manipulated. This software was used to better understand how human operators allocate their attention among multiple concurrent tasks.

The software interface represented six tasks, which were shown as bars (Figure 8.1). Should the subject not have attended to a task, its status deteriorated from the satisfactory level with a certain adjustable rate DR. On the other hand, while the subject attended to it, its status improved towards the desired state by a different adjustable rate CR. Subjects could attend to a task by simply depressing a button underneath each task using their mouse. Software updated the status of the tasks every tenth of a second. Therefore, theoretically a subject could switch between the tasks as quickly as tenth of a second although most of the time it was longer due to human psychomotor limitations. The software computed a numerical score depending on how well the subject had kept the average status of the tasks over time, how long tasks were penalized at the zero SL, and whether or not he/she had ignored the more important tasks. For further information on how this score is calculated, refer to the primary objective function in Section 5.4. The computer also recorded what task the subject had attended to at any point in time. Subjects' attendance to tasks were replayed on the computer, reviewed and studied in the form of graphs by utilizing a Microsoft Excel 2000 spreadsheet. This helped to discover the pattern of attendance to tasks (if any) used by the subject.

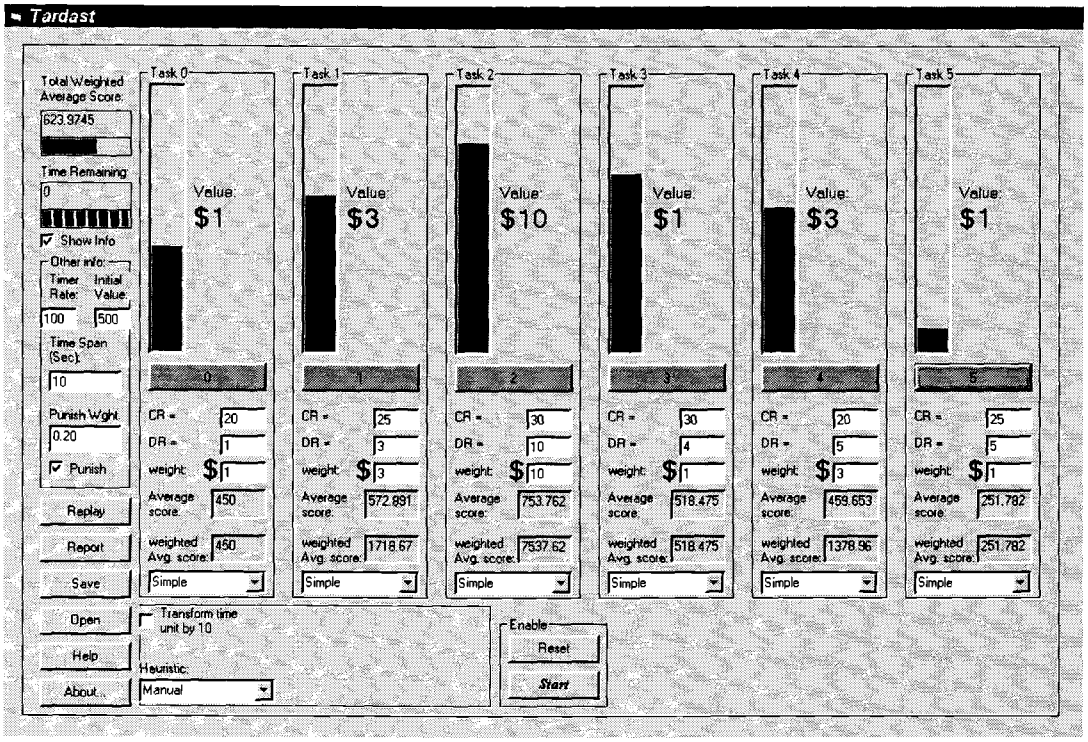


Figure 8.1 Tardast interface

8.2.3 Experimental Procedure

At the start of the experiment, the subject was assigned a unique, randomly generated identification number. This number was used for the rest of the experiment to record associated data and avoid disclosing the subject's real identity in future reports of this research. The experiment started after the subject read and voluntarily signed the informed consent document (Appendix G). The subject was then asked to read a one-page instruction sheet on the mechanics of the experiment and the game, significance of this project and its overall goal (Appendix H). Then, some further instructions were given to the subject verbally by the experimenter. For the sake of consistency, the experimenter used a one-page memory aid to recite these instructions (Appendix I).

Subjects had to play the computer game (Section 8.2.2.2) in five different scenarios. The scenarios differed from each other in the CR, DR, and weight (value) of the tasks (Section 8.2.4). The length of the game at each scenario was five minutes for which the subject's attendance to tasks for the entire game was recorded. However, subjects had to have a maximum of ten practice trial plays per scenario, each one minute long, per scenario before doing the actual five-minute run for which the data was collected. After the data was collected for all the scenarios, a brief questionnaire was given to the subject. This questionnaire inquired gender, age, computer experience (hours per week), driving experience (hours per week), and whether the subject used any particular strategy when playing the games in these different scenarios. The questionnaire can be found in Appendix J. Subjects were thanked and dismissed by the experimenter after finishing the questionnaire. The experiment lasted a maximum of 2 hours 15 minutes. Figure 8.2 shows the breakdown of the elements of the experiment and their sequence, and Table 8.1 shows the time allocation for these elements within the experiment.

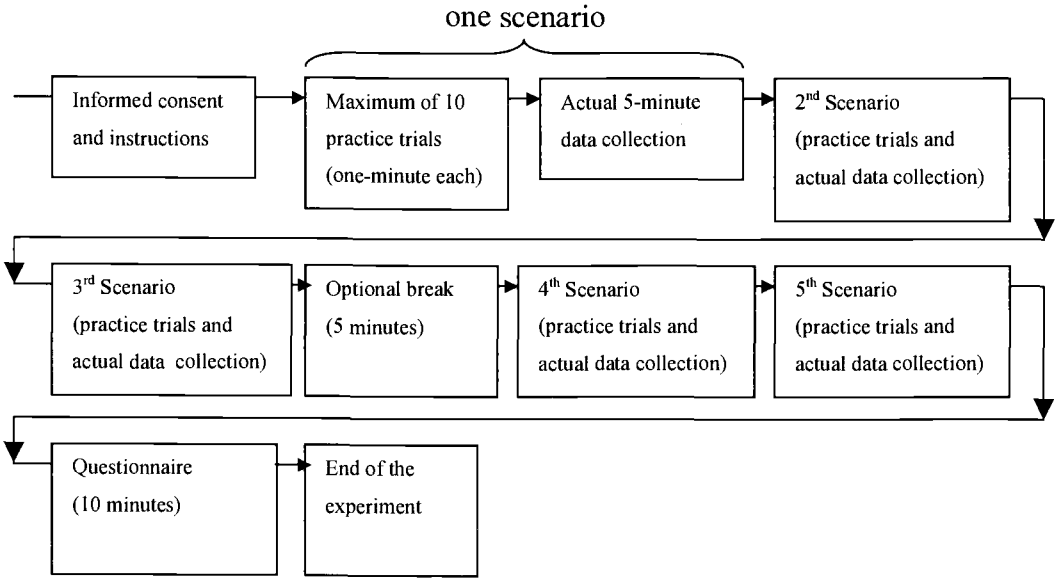


Figure 8.2 Experimental procedure for human subjects

Table 8.1 Time breakdown of the experimental elements for human subjects

Experimental Element	Time (minute)
Informed consent and pre-instructions	15
10 one-minute practice trials (per scenario)	10
Actual data collection (per scenario)	5
Total practice trials plus actual data collection (one scenario)	15
Total practice trials plus actual data collections (all 5 scenarios)	$15 \times 5 = 75$
Questionnaire	10
Break	5
Allowance (record time and switch time between scenarios)	25
Total length of the experiment per subject	**130**

The trials were only performed to let the subject familiarize himself/herself with the environment of the scenario and discover his/her own best strategy for playing the game in that scenario. Only the score (not the pattern of attendance) of the practice trials were recorded to demonstrate to the subject and experimenter how fast/much the subject was improving, how better one strategy was compared to a strategy previously chosen, and how many trials were left out of ten. If the subject's three highest scores were within 5% of each other and if the subject expressed verbally that a particular strategy would be his/her best choice, then the practice trials were brought to an end at the subject's request and the actual five-minute data collection started. Subjects were allowed a maximum of ten practice trials, which were all used by most of the subjects in the majority of the scenarios.

8.2.4 Scenarios

All five scenarios in this experiment were common in that tasks started with an initial SL = 0.50 (50%) and that tasks were penalized for 20% of their value for every time unit that they stayed at a zero SL. They all differed, however, in the combination of CR, DR, and weight. The primary intent of these scenarios was to discover how close to the near-optimal order of attendance a human subject would get by intuition. Intuition may be sufficient for the simple experimental scenarios, but by increasing the variability of the factors (CR, DR, and weight), it might have its shortcomings. If the complexity of a scenario confuses the subject so much that his/her intuition would not be sufficient for discovering the near-optimal order, it is desirable to know what factor(s)— CR, DR, or weight—is (are) the most influential in increasing this complexity and confusion.

Scenario 1 was a placebo experiment and was equivalent to a sugar pill in medical experiments. In this scenario, all tasks were identical and the subject's order of attendance had an insignificant effect on the final score. Subjects were expected to gain the closest score to the near-optimal one found by the mathematical model. This scenario was intended to help discover any bias or particular behavior in subjects' decision-making solely influenced by the environment of the experiment (hardware and software) and limitations/capabilities of humans. An example could be attending to tasks on the right side of the screen versus the left side. In this scenario, if any consistent patterns were discovered among subjects, it was to be later considered (excluded) when investigating the rest of the scenarios. It also helped to discover if a subject was exceptionally superior/inferior to the rest of subjects with regard to the use of the mouse regardless of good/bad decision-making.

In Scenario 2, tasks were identical except that most of them had different DRs. This scenario sought the subject's reaction and performance when DRs were different.

Scenarios 3 and 4 were similar to Scenario 2, but instead of DR, their CR or weight was different. The last scenario (5) was the most complicated one, in which all tasks had different CRs, DRs and weights. It was of interest to know how well subjects handled this scenario considering its complexity.
Table 8.2 shows a summary of the scenarios' designs and their underlying motivation relative to CR, DR, and weight.

Table 8.2 Design of the scenarios and their underlying motivation

Scenario	Parameter Across Tasks			Motivation
	CRs	DRs	Weights	
1	Identical	Identical	Identical	• Does the subject attend to tasks equally, or is he/she biased towards something? • How close can he/she get to the optimal score? This will be solely due to psychomotor limitations (how fast can he/she move/click the mouse) and not due to bad decision-making. • Are all subjects behaving similarly?
2	Identical	Different	Identical	Does the subject follow the optimal order or seem to ignore the differences in DR?
3	Different	Identical	Identical	Does the subject follow the optimal order or seem to ignore the differences in CR?
4	Identical	Identical	Different	Does the subject follow the optimal order or seem to ignore the differences in weight?
5	Different	Different	Different	• How close can the subject get to the optimal score? • How complex does the subject find this scenario?

In the following table, the arbitrary parameters used for a task within a scenario are shown. The task's parameters within a scenario did not change between subjects throughout the experiment. However, the order of scenarios to which subjects were exposed was random (e.g., 2-3-5-1-4) to even out the possible learning effect when taking the experiment.

Table 8.3 Parameters used for tasks in different scenarios

Scenario	Task	0	1	2	3	4	5
1	CR	9	9	9	9	9	9
	DR	3	3	3	3	3	3
	w	5	5	5	5	5	5
2	CR	9	9	9	9	9	9
	DR	2	1	3	3	4	2
	w	5	5	5	5	5	5
3	CR	9	5	8	8	9	13
	DR	3	3	3	3	3	3
	w	5	5	5	5	5	5
4	CR	9	9	9	9	9	9
	DR	3	3	3	3	3	3
	w	6	5	10	6	4	2
5	CR	9	5	8	8	9	13
	DR	2	1	3	3	4	2
	w	6	5	10	6	4	2

Table 8.4 Approximate average across tasks for parameters used in Table 8.3

Parameters	Approximate Average Across Tasks
CR	9
DR	3
Weight	5

Although the scenarios in Table 8.3 might seem to have very different parameters, they follow a simple rule. The average CR, DR, and weight for every scenario is

almost the same to preserve the number of tasks that are likely to be unattended in each scenario. Table 8.4 shows the approximate average for all scenarios:

The average ratio of $\frac{CR}{DR} = n$ -1 is a good estimate of how many tasks (n) can be kept at a high satisfaction level in the long run (Appendix K). For the parameters of the above problem, the average $\frac{CR}{DR} = \frac{9}{3} = 3$ means that a maximum of four out of six tasks are likely to stay in a high status and the remaining two will be mostly unattended. These parameters were designed intentionally to discover which tasks subjects of the experiment would ignore.

8.2.5 Experimental Design

This experiment measured the subjects' score after playing with the software for five minutes in each of the five scenarios. On the other hand, for each scenario, tabu search was run a few times and the best score of those trials were picked as a standard of comparison for that scenario. All subjects could not attend to any task for about 5-15% of the time due to being slow in handling the mouse, which lowered their score. On the other hand, tabu search does not have such a shortage, which makes it hard to have a fair comparison. Therefore, subjects' data was thoroughly gone through and *repaired*. It means that any time that they did not attend to any task, the gap in the data was replaced with the next task to which they actually attended. Then, the subject's score after the data repair in each scenario was compared to the score gained by the tabu search. The following two tests were conducted:

1) A two sample pair-wise comparison combining all the scenarios to find out if there was any difference between the mean of subjects' scores and tabu-search's scores.

2) A two sample pair-wise comparison for each scenario to find out if there was any difference between the mean of subjects' scores and tabu-search's scores in that scenario.

8.2.5.1 Dependent Variable

The dependent variable was the subject's repaired score gained after playing with the software for five minutes.

8.2.5.2 Independent Variable

The independent variables were:

- *Scenario*: A fixed effect variable with five levels.
- *Subject*: A random effect variable with ten levels.

8.3 STATISTICAL RESULTS

The statistical results in both combined and per-scenario cases indicate with 95% confidence that there is a significant difference between the repaired results obtained by subjects and the corresponding tabu search results. Since the mean and median of the difference (subject's repaired score minus tabu's score) was negative in all cases, and since subjects could never attain a score higher than tabu's, it could be concluded with 95% confidence that tabu search has a better performance than the subjects. See Appendix L for further details on the statistical tests.

8.4 BEHAVIORAL RESULTS

In this section, in order to study the behavior of the subject more objectively **the original data before repair is used to draw the graphs in Appendix M**. If the repaired data were used, the "none" column in the graph for frequency of attendance would be zero. In all scenarios, the subjects' pattern of attendance is considered and the best, worst, and mean performance is discussed. Also, scores of

all the subjects are shown by a bar graph at the end of each section to give an overview of the overall performance in each scenario to the reader.

8.4.1 Scenario 1: Identical Tasks

In this scenario, tasks were identical and had no difference in parameters. The following table shows the parameters used for tasks in this scenario.

Table 8.5 Parameters used for tasks in Scenario 1

Scenario	Task	0	1	2	3	4	5
1	CR	9	9	9	9	9	9
	DR	3	3	3	3	3	3
	w	5	5	5	5	5	5

8.4.1.1 Best

Figure M.1 in Appendix M shows the best participant performance in this scenario. Three of the tasks were concurrently brought to 100% SL in the first minute, and maintained at that level for the rest of the time span. This resulted in an above 90% average SL for these three tasks (3, 4, and 5) and a near zero SL for the rest. Tasks 0, 1, and 2 were not attended to at all. The subject did not attend to any task for 15% of the time due to being slow in handling the mouse.

8.4.1.2 Worst

Figure M.2 in Appendix M illustrates the performance that resulted in the lowest score in this scenario. The subject in this case attended to only four of the tasks (0, 1, 2, 3 and 4) sequentially for most of the time (first three minutes), and completely ignored the rest of the tasks (4 and 5). In the last two minutes of the time span, the

subject attempted to have three (and later down to two) of the tasks at a fairly high SL, and kept the rest of the tasks (out of the four attended to) just high enough to not be penalized. Overall, this subject had an average SL of 30-45% in four of the tasks and almost zero average in the rest. This subject did not maintain a high SL in any of the tasks long enough to improve his score.

8.4.1.3 Tabu

Figure M.3 in Appendix M depicts the order of attendance suggested by the tabu search. Tasks 0, 2, and 4 were attended to and kept at a very high status for most of the time span. After the third minute, task 0 got less attended to in favor of task 3. Task 1 was shortly attended to in the first half, and task 2 was occasionally attended to in the second half of the time span.

8.4.1.4 Mean

Figure M.4 in Appendix M shows the average performance of all the participants in this scenario. Tasks 3 and 4 had the highest frequency of attendance and tasks 0 and 1 the lowest. Consequently, tasks 3 and 4 had the highest average SL and tasks 0 and 1 the lowest.

The following Figure shows the original and repaired scores of all the subjects in this scenario.

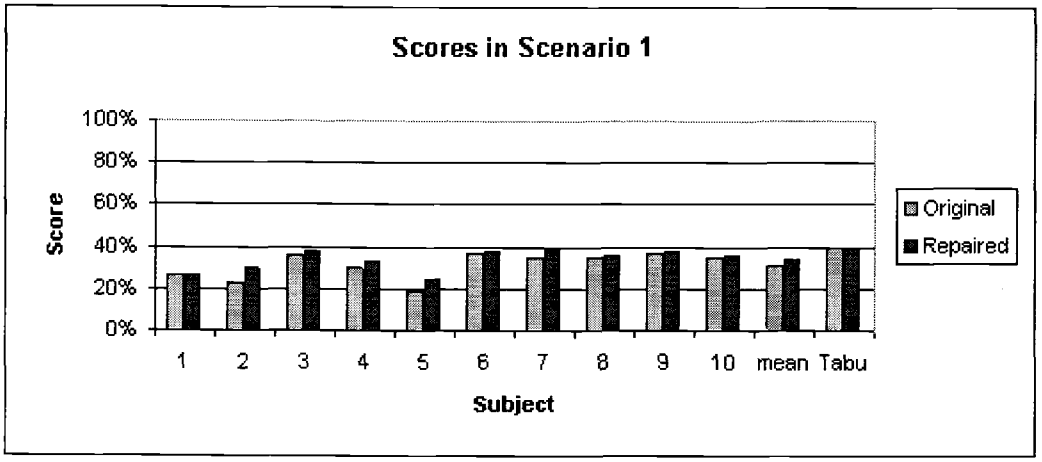


Figure 8.3 Subjects' original scores and repaired scores in Scenario 1

8.4.2 Scenario 2: Different DRs

In this scenario, tasks had identical parameters except for their DR. These tasks in their increasing order of DRs were: 1, 0 (or 5), 5 (or 0), 2 (or 3), 3 (or 2), and 4. The following table shows the parameters used for tasks in this scenario.

Table 8.6 Parameters used for tasks in Scenario 2

Scenario	Task	0	1	2	3	4	5
2	CR	9	9	9	9	9	9
	DR	2	1	3	3	4	2
	w	5	5	5	5	5	5

8.4.2.1 Best

Figure M.5 in Appendix M shows the best participant performance in this scenario among the subjects. This subject attended to the two lowest DR tasks (1 and 0) first and brought them to a high SL; then attended to the next lowest DR task (5) and

after these three tasks were at a fairly high status, the next lowest DR task (2) was attended to. For a good portion of time subject maintained all these four tasks (0, 1, 2, and 5) at a high SL. In the meanwhile, the subject occasionally attended to the least DR task (4). The subject could have gained an even better score if he attended to task 3 instead of task 4 due to its lower DR. Although tasks 2 and 3 were identical, task 3 was not attended to at all, while task 2 had the highest attendance rate. Overall, the subject maintained a high SL (70-90%) in the four lowest DR tasks, and an over 30% SL in the fifth task (4). The 6^{th} task (3) was not attended to at all although it did not have the highest DR. This subject did not attend to any task for 3% of the time due to being slow in handling the mouse.

8.4.2.2 Worst

The performance with the lowest score is shown in Figure M.6 in Appendix M. The subject maintained the highest SLs of 60% and 45% for the two lowest DR tasks (1 and 5). However, due to being overly concerned with the penalization of the rest of the tasks, he attempted to not let any of them reach a zero SL. This resulted in a poor average SL for these tasks.

8.4.2.3 Tabu

The attendance order suggested by the tabu search can be seen in Figure M.7 in Appendix M. Initially, the three lowest DR tasks (0, 1, and 5) were brought up to a high SL. Then, gradually the model attended to the next two low DR tasks (3 and 2), and brought them to a high status. For about 100 seconds the model kept all these five tasks at a high status concurrently. In the last 50 seconds, because the model realized the time span was about to end and the two lowest DR tasks (0 and 1) would not reach a zero SL, the model stopped attending to them in favor of avoiding task 4 from being penalized. The model kept a good average SL for all tasks with a range of 20-80% SL. Task 0 and 1 had the highest average SL (80%).

However, task 0 had to be attended to twice as often to attain the same average as task 1 due to the fact its DR was also twice the DR of task 1.

8.4.2.4 Mean

The overall average performance of all participants for this scenario is depicted in Figure M.8 in Appendix M. As illustrated, the participants had a good understanding of the differences in DR and attained a higher SL for the task with the lower DR. Identical tasks (0 and 5), and (2 and 3) were treated almost equally by the participants. In this scenario, the participants did not attend to any task for less than 5% of the time.

The following Figure shows the original and repaired scores of all the subjects in this scenario.

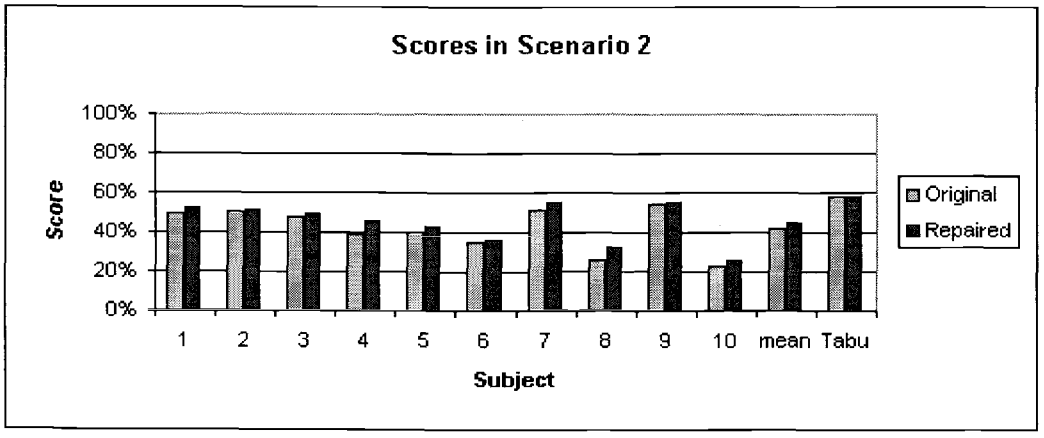


Figure 8.4 Subjects' original scores and repaired scores in Scenario 2

8.4.3 Scenario 3: Different CRs

This is a scenario in which tasks had equal DRs and equal weights. They only differed in their CRs. These tasks in their decreasing order of CRs were: 5, 0 (or 4),

4 (or 0), 2 (or 3), 3 (or 2), and 1. The following table shows the parameters used for tasks in this scenario.

Table 8.7 Parameters used for tasks in Scenario 3

Scenario	Task	0	1	2	3	4	5
3	CR	9	5	8	8	9	13
	DR	3	3	3	3	3	3
	w	5	5	5	5	5	5

8.4.3.1 Best

The graphs for the best participant performance in this Scenario are illustrated in Figure M.9 in Appendix M. The subject realized that the best strategy would be to only attend to four tasks with the highest CR. Therefore, tasks 1 and 2 were never attended to. The subject initially attended to the two highest CR tasks (5 and 0) and brought them near to their 100% SL quickly and kept them stable before attending to the next two high CR tasks (4 and 3). Then, the subject tried to maintain a high SL for task 3 in the first half of the planning horizon, and for task 4 in the second half. Overall, these two tasks maintained an average SL of almost 60%. The bar graphs show that task 3 required more attendance than task 4 to keep the same average SL. This is due to slightly lower CR of task 3 compared to task 4. Among the four tasks attended to, task 5 had the lowest attendance at 20%, but the highest average SL again because of its very high CR. This subject had good understanding of the differences in CR between the tasks and attended to them accordingly. The subject did not attend to any task for 4% of the time due to being slow in handling the mouse.

8.4.3.2 Worst

The performance with the poorest score in this scenario is illustrated in Figure M.10 in Appendix M. It is observed from the data that the subject had a fairly sequential order of attendance to tasks with the primary intention of not letting any task reach a zero SL. This subject spent the most time on task 3 and 4 (third and second high CR tasks). Overall, he maintained a poor average SL of 20-30% for the first two high CR tasks (4 and 5) and a poorer average SL of less than 20% in the remaining tasks. This subject tried to manage too many tasks and was overly concerned with the penalization of tasks. The subject seems to have had a moderate understanding of the differences in CR between the tasks, but he did not have a clear strategy with regard to these differences.

8.4.3.3 Tabu

The attendance order suggested by the tabu search in this scenario is depicted in Figure M.11 in Appendix M. Tasks 5, 4 and 3 (first, second, and fourth high CRs) were brought to near 100% SL for the first minute of the experiment. Afterwards, task 4 received less attention in favor of bringing task 0 (second high CR) to near 100% SL. After the second minute, task 4 gradually resumed its status also and all four tasks (0, 3, 4, and 5) were juggled to maintain a high and fairly stable SL. The highest CR task held a high SL average of nearly 90%; the next three high CRs maintained an average of 60-85% SL. The two least CR tasks (tasks 1 and 2) were almost unattended.

8.4.3.4 Mean

Figure M.12 in Appendix M illustrates the average performance of all participants. They all seem to have had a good understanding of the differences in CR and planned their attendance accordingly. The highest CR task (5) maintained the highest average SL (64%). The next two high CR tasks (0 and 4) maintained the next high averages (almost 45%). The least high CR task (1) almost did not receive

any attention. Across all participants, no task was attended to for about 5% of the time.

The following Figure shows the original and repaired scores of all the subjects in this scenario.

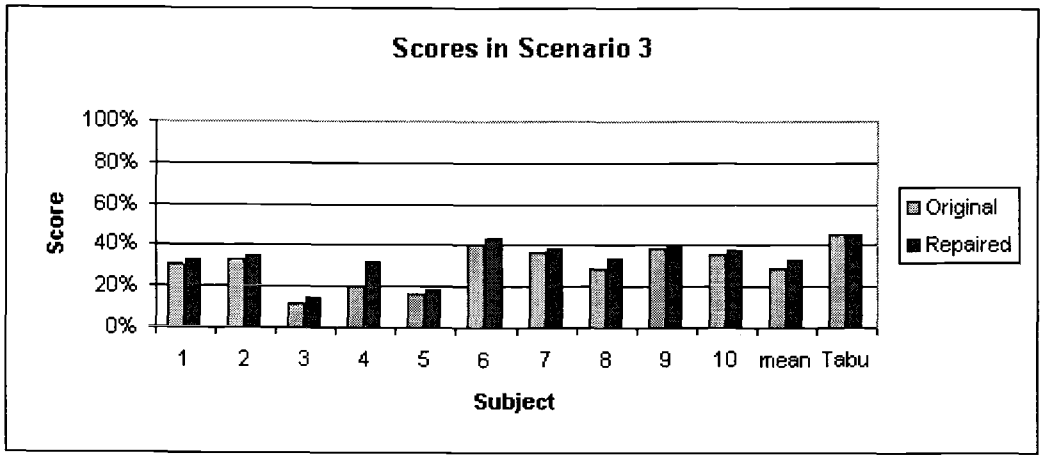


Figure 8.5 Subjects' original scores and repaired scores in Scenario 3

8.4.4 Scenario 4: Different Weights

In this scenario, tasks had identical parameters except for their weight. These tasks in their decreasing order of weight were: 2, 0 (or 3), 3 (or 0), 1, 4, and 5. The following table shows the parameters used for tasks in this scenario.

Table 8.8 Parameters used for tasks in Scenario 4

Scenario	Task	0	1	2	3	4	5
4	CR	9	9	9	9	9	9
	DR	3	3	3	3	3	3
	w	6	5	10	6	4	2

8.4.4.1 Best

Figure M.13 in Appendix M shows the performance of the subject with the best score in this scenario. The subject equally focused on tasks 2, 3, and 0 throughout the experiment. In the first minute, he juggled between tasks 2, 3, and 0 sequentially to attain their 100% SL. After reaching 100% SL, he kept near this status by consistently juggling between the same tasks. This strategy resulted in an overall average of over 90% SL for tasks 2, 3, and 0, and an average of near zero SL for the rest of the tasks. The subject clearly favored the three more valuable tasks and completely ignored the other three. This subject did not attend to any task for 11% of the time due to being slow in handling the mouse.

8.4.4.2 Worst

The worst performance in this scenario is depicted in Figure M.14 in Appendix M. This subject tried to attain a good SL in four and sometimes five tasks, which resulted in a poor score. He managed to keep an average of near 76% SL for the most valuable task, but kept a poor average of 16% or less for the rest of the tasks. Besides the most valuable task (2) that had an attendance rate of 27%, each of the other three valuable tasks (0, 3, and 1) had a decent attendance rate of 20-23%. The reason these three tasks had such a low average SL, considering the good attendance rate, is that the focus was to avoid letting them reach zero SL and be penalized. The subject acknowledged the differences in weight between the tasks

and attended to them accordingly. However, he tried to handle too many tasks being overly concerned with avoiding penalization of tasks.

8.4.4.3 Tabu

In this scenario, the solution suggested by tabu search after a few trials is demonstrated in Figure M.15 in Appendix M. Matching with the best performance previously discussed, the three most valuable tasks were the focus of attention for the first 40 seconds until they reached a stable state near their 100% SL. Afterwards, they were kept in their steady state as much as possible with an attendance rate of over 25%. When in the steady state, the next valuable task (1) occasionally was attended to with an overall attendance rate of almost 20%. The overall averages of the three most valuable tasks' SL were slightly below 90% as opposed to slightly above 90% in the best subject performance. However, the next valuable task (1) had a near 20% SL average in this case, while it was not attended to at all with an almost zero average SL in the best subject performance case.

8.4.4.4 Mean

The participants' performance in this scenario on average (Figure M.16 in Appendix M) indicates that they had a good understanding of the differences in weight between the tasks. The three most valuable tasks were attended to the most, each with a rate of 25-30%. The next frequency of attendance belonged to the fourth valuable task (1) and the other two tasks were ignored. Thus, the overall attendance was similar to the one suggested by the tabu search. The most valuable task (2) had an average SL of 85%; the next two valuable tasks (0 and 3) had an average SL of over 60%; the fourth valuable task (1) had an average of slightly above 10%; and the rest of the tasks had almost a zero average. Subjects had an average rate of 7% for not attending to any task in this scenario.

The following Figure shows the original and repaired scores of all the subjects in this scenario.

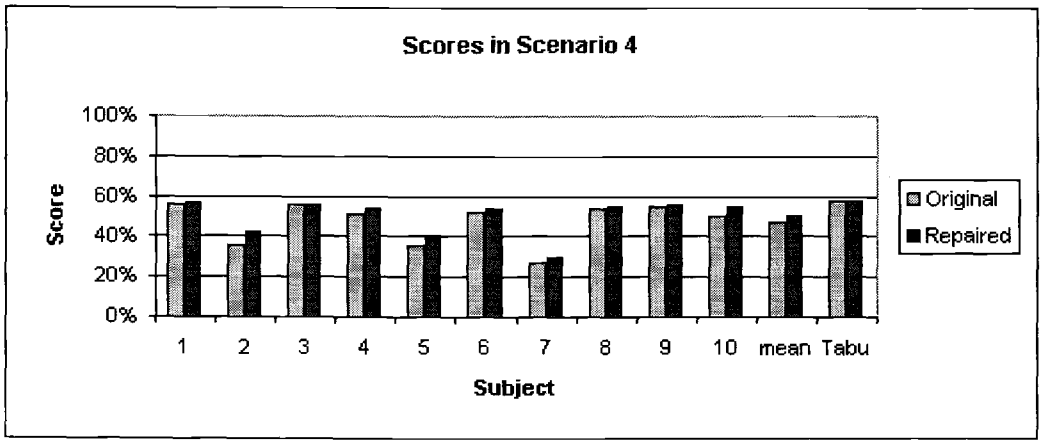


Figure 8.6 Subjects' original scores and repaired scores in Scenario 4

8.4.5 Scenario 5: Different CRs, DRs, and Weights

This was the most complicated scenario in which all of the tasks' parameters (CR, DR, and weight) differed from each other. Tasks' DRs were identical to the case in Scenario 2; their CRs were identical to Scenario 3; and their weights were identical to Scenario 4. The following table shows the parameters used for tasks in this scenario.

Table 8.9 Parameters used for tasks in Scenario 5

Scenario	Task	0	1	2	3	4	5
5	CR	9	5	8	8	9	13
	DR	2	1	3	3	4	2
	w	6	5	10	6	4	2

8.4.5.1 Best

The best participant performance in this scenario is illustrated in Figure M.17 in Appendix M. This performance was almost the same as the best performance in Scenario 4. That is the differences in CR and DR between the two scenarios did not affect anything and only the three most valuable tasks were attended to and the rest were ignored. This subject did not attend to any task for 10% of the time due to being slow in handling the mouse.

8.4.5.2 Worst

This performance, depicted in Figure M.18 in Appendix M, has resulted in the poorest score because the subject has attempted to manage all six tasks. Except the two most valuable tasks (2 and 3), the rest of the tasks were either over or under attended considering their value and contribution to the final score. The most valuable task had an average SL of over 30%, while the rest of the tasks had an average of 10-15% SL. This poor result could have improved if some low-value tasks had attained a lower average in favor of the high-value tasks attaining a higher average. The subject's attending to too many tasks, being overly considerate of getting penalized, caused his poor performance.

8.4.5.3 Tabu

Figure M.19 in Appendix M shows the solution suggested by the tabu search after a few trials. This strategy initially attended to tasks 0, 1, and 2 with the primary intention of achieving near 100% SL for 0 and 2, and meanwhile keeping task 1 at 50% SL. This was a very reasonable strategy as task 2 had the highest value. Task 0 and 3 equally had the second highest value, but task 0 was preferred as it had a better CR and a lower DR. After these two tasks (0 and 2) reached a fairly steady state near 100% SL, the tabu strategy also attended to task 1. Although this task (1) had both a lower value and CR compared to task 3, it seems to be preferred because it had a lower DR.

After task 1 also reached a near 100% SL along with the other two (0 and 2), the fourth task (3) was attended to around the 50th second. This task had the highest value between the remaining three, which all had a zero SL by then. By the second minute, all four tasks reached a steady state near their 100% SL. The two least valuable tasks were rarely attended to. The least valuable task (2) was also rarely attended to seemingly because of its very high CR and low DR. Tasks (0, 1, and 2) had an overall average SL of above 90%; task 3 had an over 70% average SL. This indicates that this strategy successfully managed the four highest value tasks.

8.4.5.4 Mean

The overall performance of the participants in this scenario (Figure M.20 in Appendix M) indicates that they had a good understanding of the differences between the scenarios. The three most valuable tasks had the highest rate of attendance, 20-30% each. They also held an average SL of 55-75%. The fourth valuable task (1) had the fourth rate of attendance (10%) and also the fourth highest SL (over 20%). The participants' performance had good similarity with the one suggested by the tabu search, except that they were less careful about task 1. Although task 1 was the third valuable task, it had the least DR, and so it was very easy to keep at a high SL. In this scenario, the participants did not attend to any task for 6% of the time due to not handling the mouse properly.

The following Figure shows the original and repaired scores of all the subjects in this scenario.

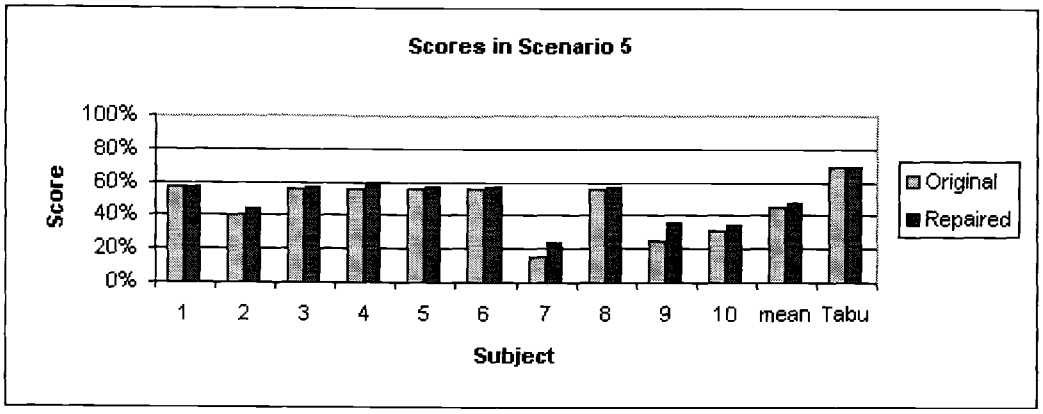


Figure 8.7 Subjects' original scores and repaired scores in Scenario 5

8.4.6 Questionnaire

Subjects were given a brief questionnaire after finishing all five scenarios. The questionnaire included questions about their specifications (Section 8.2.1), any particular strategy for task management if used (discussed below in Section 8.5.2) and any comments that they had about the way the experiment was carried out. Highlights of these comments are:

- They enjoyed the experiment: "Cool experiment!" "It's an interesting game!" "...[it] was entertaining."
- Problems with the mouse: "Using [the] mouse is not convenient." "My finger hurts!" "I was trying to push my finger [in difficult scenarios...] to achieve a grand score." "The [right] mouse button got in the way occasionally when I accidentally pressed it and lost the [attention on the task temporarily.]"
- Training procedure: "Training should be [several] three-minute [trials] instead of [ten] one- minute [practice trials]."
- Interface of the software: "How about making [height of] the buttons larger at the expense of the [height] of [the] bars?"

8.5 DISCUSSION

8.5.1 Statistical

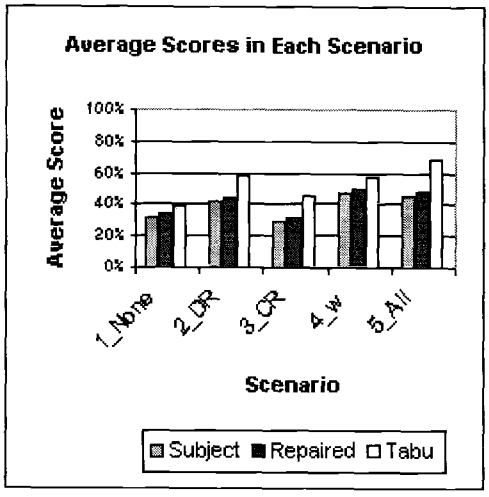
In all the experiments conducted, subjects' performances were inferior to tabu-search's. In this comparison, subjects were given two bonuses compared to tabu search to compensate for the time lost by the subject for not attending to any task.

1) Tabu search was intentionally limited to switch between the tasks one second at a time at its quickest switching rate. However, subjects could switch between the tasks as fast as every tenth of a second if they opted to.
2) In all the experiments conducted, subjects did not attend to any task for 1-15% of the time with an average of 6%. In the statistical comparison, subjects' data was repaired. That means any time that the subject did not attend to any task, the gap in the data was filled with the task that he/she next attended to. The score of this new data was calculated and then compared with tabu search in the statistical comparison.

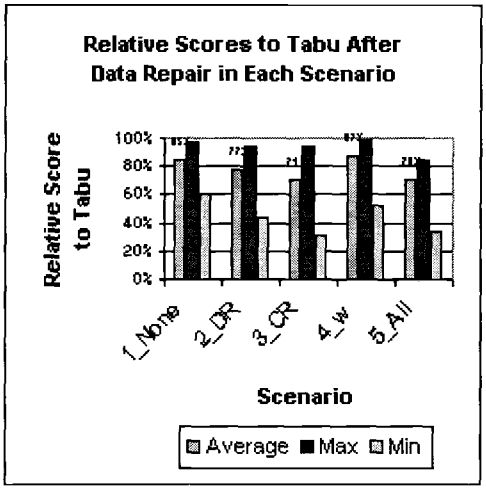
In spite of the above two bonuses given to the subjects, neither in all scenarios combined, nor in any scenario considered independently, could subjects beat the score gained by the tabu search. In some cases, however, subjects gained good scores that were within 3% of the tabu solution (Figure 8.8). The maximum best score gained among the subjects in each scenario was no worse than 15% of the tabu solution in any of the scenarios.

Figure 8.8 shows that on average subjects had the best performance in Scenario 1 (placebo) and Scenario 4 (different weights), in which they averaged around 15% of the tabu solution. This result was expected in Scenario 1 as all the tasks were identical and the order of attendance had the least significance. However, for Scenario 4, it means that subjects had a good reaction to the differences in weight

between the tasks. This also might be because the weight of the tasks were explicitly displayed to the subject all the time as opposed to CR or DR, which they had to sense by playing with the software.



(a) Subjects' average original scores, repaired scores, and corresponding tabu scores



(b) Subjects' relative mean, maximum, and minimum scores to tabu scores after the data repair

Figure 8.8 Summary of subjects' scores in each scenario

In the same figure, it can be seen that subjects in Scenario 3 (different CRs) and Scenario 5 (with all parameters being different) held the largest relative difference to tabu. This result in Scenario 5 means that the complexity of the scenario due to different parameters in CR, DR, and weight enlarged the gap between the subject's performance and the near-optimal one. The result in Scenario 3 can be justified with the fact that subjects could sense a task's CR only when they attended to that task. However, subjects had a better sense of the differences in DR (Scenario 2) because they could compare the DRs just by looking at the adjacent tasks dropping.

This relative comparison for CRs was not as easy in Scenario 3 because only one task at a time could be attended to, and no two tasks could be improved at the same time to be compared. The fact that a task's DR was always visible while its CR was visible only when attended might explain the 6% better average in Scenario 2 versus Scenario 3 in Figure 8.8 (b). A task's weight, on the other hand, was given to the subject explicitly on the screen, which might explain the better performance in Scenario 4 versus Scenarios 2 and 3.

8.5.2 Behavioral

Looking at the subjects' performance graphs, based on the original data before the data repair, and reading their answers to question 6 in the questionnaire on whether they followed a particular task management strategy, several common patterns were recognized. These findings are discussed in the following bulleted paragraphs.

- Subjects who were overly concerned with the penalization of the tasks and attempted to handle too many tasks (more than four) could not perform very well. These subjects had a tendency to over estimate the effect of 'failure' or in this case penalization of the tasks. Almost all subjects learned to let one or more tasks go (less than 5% attendance rate) and in consequence gained better scores as they advanced through the experiment. See Section 8.5.3.2 for further details.

- The general rule of priority is: tasks with higher weight, lower DR, and higher CR should be kept at the higher average SL. The majority of subjects had the above approach partially in mind. The better they followed this approach, the better score they received.

- Subjects who tested and explored all the tasks at each scenario in their practice trials had a better understanding of the parameters (CR and DR) of

the system and, in consequence, could prioritize their tasks better. This exploration was not needed for the value of a task as it was displayed next to each task.

- The majority of subjects took the parameters of the system under consideration in the order of weight (value), DR, CR, and proximity. That is, if the values of the tasks were significantly different, the higher value ones were attended to without much consideration of CR or DR (Scenario 4 and 5). When weights (values) were equal or nearly equal, tasks with lower DRs were attended to without much consideration of CR. If weight and DR were equal or nearly equal, subjects attended to high CR tasks. And when the difference between value, DR and CR across the tasks was not quite apparent, tasks that were in close proximity of each other were attended to. (Note: "proximity" is not considered as a parameter in the mathematical model in this research.)

- The order in which subjects could identify the parameters was also very similar to the above order in which the subjects took those parameters under consideration. The increasing order of difficulty for understanding the parameters was: weight (value), proximity, DR, and CR. That is, subjects had a perfect knowledge of the values as they were displayed on the screen next to each task, and they also had a perfect knowledge of the tasks' proximity; they had a good knowledge of the DRs as each task could be compared with its adjacent tasks when deteriorating simultaneously; and a fair knowledge of the CRs because it was not easy to compare two CRs simultaneously. For the same reason, subjects reported that Scenario 5 (different weights, DRs, and CRs) was the easiest to distinguish its parameters, and Scenario 1 (identical tasks) was the most difficult.

Ironically, they gained the farthest average score from the optimal in Scenario 5, and the second closest score to the optimal in Scenario 1.

8.5.3 Learning Effect

Any experiment involving human subjects is in jeopardy of being affected by the learning effect. Such effects are typically controlled, reduced, or eliminated by taking proper counteractions. One or more of the following learning effects might have affected this experiment:

1) Learning effect within the scenario: the more the subject practices a scenario, the better score he/she gains.
2) Learning effect between the scenarios: the more the subject practices the game in general, the better score he/she gains.
3) Learning effect of the experimenter: the more the experimenter conducts the experiments, the better he learns how to conduct them.

8.5.3.1 Learning Effect Within the Scenario

This effect was desirable because the experimenter intended to compare the subject's best possible performance with that of tabu search. This was handled by letting the subjects play the one-minute practice trials as many as ten times. Some of the subjects who had reached their peak of performance early in their practice trials showed little or no improvement in their consecutive scores. Having settled on their preferred strategy, subjects were allowed to quit the practice trials before the tenth round. A good majority of the subjects, however, used all ten trials since they wanted to practice and experience new strategies even after reaching their performance peak. Therefore, the practice trials were very effective in letting the subjects reach the peak of their learning curve. However, these trials could not be fully successful because the time span of the practice trials (1 minute) was different from the time span of the data collection (5 minutes). One might argue that the

subjects could still improve their score if they had practiced more with the 5-minute time span. However, this was not practical due to being too time consuming for the scope of this research.

8.5.3.2 Learning Effect Between the Scenarios

Unlike the previous learning effect, this effect was undesirable, so it was handled by randomizing the order of the scenarios that the subject was exposed to. Many of the subjects gradually became more familiar and skilled with the environment and goals of the experiment as they advanced through the scenarios. This learning was independent from the order of the scenarios in which they were exposed to. Randomization of the scenarios' order would successfully even out (but not eliminate) this effect across the scenarios. Hence, the *statistical* conclusions for each scenario, or all scenarios combined, would not be affected by this learning effect. However, randomization would not eliminate this effect if the subjects' results were to be analyzed *individually*. If a person has received the lowest score among the other subjects in one scenario, that scenario might have been the first scenario that he/she was exposed to.

Perhaps, one other method that could further reduce the learning effect was to randomize the order of tasks (parameters) within each scenario. For example, in table 8.3, it can be seen that Scenario 3 and 5 have equal correction rates for each task. If a subject were exposed to these two scenarios consecutively, his/her knowledge that the task on the right most of the screen (task 5) has the highest CR (13) would carry on to the next scenario. Although subjects were encouraged to reevaluate the tasks' parameters in each scenario independently, their understanding of these parameters might have been affected by the learning effect if exposed to certain scenarios successively. This problem could be overcome by randomizing the order of tasks in each scenario. Figure 8.9 shows the existence of the learning effect between the scenarios.

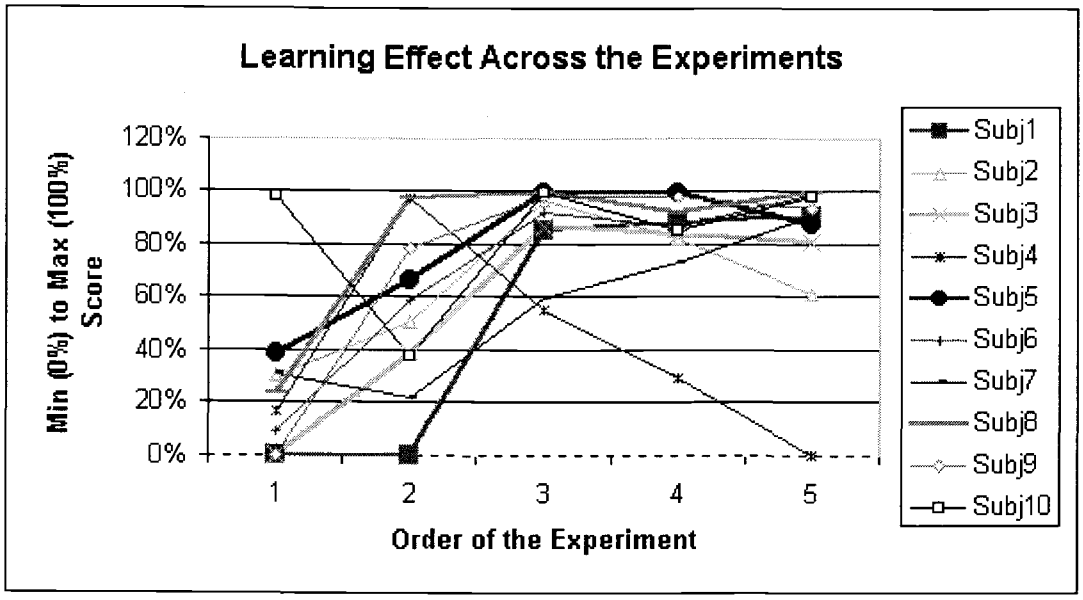


Figure 8.9 Learning effect across the order of experiments

Note that the X-axis in Figure 8.9 indicates the order, and not the type of scenario. For example, 4 in that axis does not mean Scenario 4, but it means the fourth scenario that the subjects were exposed to, which was different between the subjects. The Y-axis shows a relative percentage of the subject's score compared to the other subjects in the scenario that he/she was exposed to. For example, if a subject's second scenario were Scenario 4, and he gained the lowest score in that scenario (4), he would receive a 0% at x = 2. On the other hand, he would receive a 100% at x = 2 if he gained the highest score among the other subjects for Scenario 4. If a subject's x^{th} experiment were Scenario i, the y value for the point (x, y) would be calculated by:

$$y \text{ value} = \frac{\text{subject's score in scenario } i - \min(\text{scores in scenario } i)}{\max(\text{scores in scenario } i) - \min(\text{scores in scenario } i)} \times 100$$

As demonstrated in Figure 8.9, there is an obvious learning curve across the scenarios exposed to. In the third, fourth, and fifth experiment, eight of ten subjects scored on the top 20% of the score range for the scenario, which they were exposed to. On the other hand, in the first scenario, nine out of ten subjects scored in the lower 40% of the score range for the scenario, which they were exposed to. This clearly shows that the subjects have performed better the more they progressed into the experiments. Figure 8.9 also shows that no subject had very poor or very good performance. All subjects had at least one score in the top 20% and in the low 40%.

8.5.3.3 Learning Effect of the Experimenter (Serial Effect)

When analyzing the data collected from an experiment, one of the undesirable effects that the data analyst needs to investigate is called *serial effect*. This effect may be in the form of a gradual increase/decrease in the results of the experiment as one advances through data collection. For further information on serial effect see the textbook by Ramsey and Schafer (2002). In general, serial effect is attributed to gradual change in the settings of the experimental environment, including the experimenter, over time. For the purpose of this experiment, the influential parameters of the experimental environment such as the computer, monitor, mouse, chair, illumination, and so on are believed to have remained unchanged. However, one parameter was not completely immune to this effect, and that was the way the experimenter conducted the experiment.

Although an experimenter usually tries his/her best to be consistent throughout the experiments, he/she gradually learns how to conduct his/her experiments faster, easier, and smoother. This gradual improvement in conducting the experiment might affect the results of the experiment and cause inconsistency, which is undesirable. The experimenter of this research also might have been subjected to this learning effect. This effect was handled by asking the subject to read a one-

page instruction sheet (Appendix H) with sufficient time before the experiment to maintain consistency of the instructions that they received. Moreover, there were minor hints that were given verbally by the experimenter before the start of the experiment. These hints were written down as a memory aid for the experimenter's consistency, but explained verbally to the subject to save time. The experimenter also had determined in his mind that subjects would not be given any information about the CRs and DRs of a task, or any recommendation on what strategy to choose. The potential for inconsistency, however, still existed in answering the subjects' questions in the middle of the experiments. Each question and answer was unique and there was no one way to answer it, and this might have caused the experimenter to gradually answer questions in a certain, biased way. The following graph shows the learning curve across the subjects.

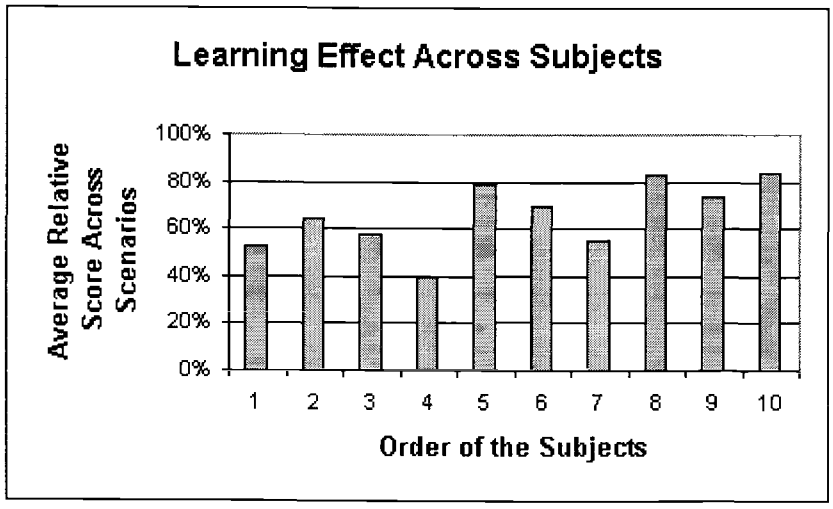


Figure 8.10 Learning effect across the order of subjects

The Y-axis of Figure 8.10 (above) is the average of y values in the formula explained for Figure 8.9 across all five scenarios for each subject. It can be seen that the last three subjects scored on the top 30% of the score ranges of the

scenarios that they were exposed to. On the other hand, the first four subjects scored only on the top 50%. Moreover, the two highest performances belong to the last three subjects. All these observations suggested the possibility of a learning effect caused by the experimenter across subjects. The report generated by STATGRAPHICS PLUS Version 5.0 in Appendix N shows that there is a statistically significant relationship between the 'average relative score across scenarios' and the 'order of subjects' at the 95% confidence level. This concludes that the learning effect caused by the experimenter across the subjects is significant.

8.6 CONCLUSIONS

This chapter started with developing an environment in which a subject was exposed to several tasks that had to be managed concurrently. The subject's goal was to gain the highest score in every scenario. This score was compared against the near-optimal score for each scenario. This environment also allowed recording the pattern of the subject's attendance and analyzing it by replaying it or using graphs.

A primary conclusion is that no subject could gain a score, even after the data repair, higher than the near-optimal score in any of the scenarios. This conclusion was verified statistically with 95% confidence that tabu's score is superior to that of the subject in each scenario and in all scenarios combined.

The best average results for subjects' repaired scores were found in Scenarios 1 and 4, which were within 15% of the near-optimal. In Scenario 1, all the tasks were identical and task management had its least effect. And in Scenario 4, tasks only differed in weight (value), which was an explicitly displayed parameter on the screen. The worst average score (within 30% of the near-optimal) was in Scenario 5 where all the tasks had different parameters (CR, DR, and value). This clearly indicates that the subjects did not perform optimally and the gap between their

scores and the near-optimal score enlarged as the complexity of the problem increased.

The maximum repaired scores in each scenario followed the same pattern as the average repaired scores mentioned in the above paragraph. Scenarios 1 and 4 had the highest maximum scores (within 1% and 3% of the near-optimal), and Scenario 5 had the lowest maximum score (within 15% of the near-optimal). These results imply that by further practice and/or proper training humans can achieve near-optimal scores especially in less complicated scenarios.

The subjects' consideration of parameters when attending to tasks was closely related to their understanding (identifying) of those parameters. The parameters that were easier to identify by the subjects, were also considered more in making decisions. The decreasing order in which the subjects considered the task's parameters in their decision-making was: weight (value), DR, and CR. Not surprisingly, the decreasing order of simplicity in which the subjects could identify and distinguish these parameters between the tasks was the same: weight (value), DR, and CR.

The more the subjects tried to keep tasks with higher weight (value), lower DR, and higher CR in a higher average SL, the better score they received. The majority of the subjects had the above strategy in mind for at least a portion of time.

It was observed that the subjects who performed poorly were overly concerned with being penalized (seemingly "failure") and attempted to handle too many tasks. Those who ignored one, two, or more tasks gained a higher score.

In general, the subject's strategy with regard to the number of tasks attended, and the selection of those tasks was the most important factor in gaining a good score.

Handling too many tasks or attending to the wrong tasks decreased the subject's score significantly. Moment to moment decisions on how to attend to tasks were insignificant in gaining a high score compared to the strategy chosen.

Generally, tabu search had a stepwise strategy in handling the tasks. It first brought three tasks to a high SL concurrently, and then attended to the fourth and sometimes fifth one while maintaining the previous tasks attended to at a high status. Most of the subjects performed poorly in this regard. They either attempted to handle too many tasks all at once, which resulted in a poor score; or they ignored half of the tasks even while the other half was in good standing, which still resulted in a good score if the choice of tasks were correct.

The pattern of attendance by the subjects was not too far from the one suggested by the near-optimal solution. Those subjects who scored average or higher had partial similarities with the near-optimal pattern for at least a portion of time.

The learning effects caused by the subjects and by the experimenter affected this experiment. Although the primary statistical and general behavioral conclusions are robust and reliable for the subjects together, one should consider these effects carefully when interpreting the behavior of the subjects individually.

The main contribution of the present work is the development of an abstract, but flexible environment that can measure and analyze a human's task management, and compare it with that of another human or a near-optimal performance. A good/bad or near-optimal strategy depends on the structure of the experiment and the parameters of the system. Therefore, one has to be cautious in extending his/her conclusions to other environments with different parameters. There are many further experiments that can be done in such a flexible environment, but probably

the most important of all is to assess how closely this environment represents real-life task management.

CHAPTER 9: CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

The motivation behind this research was to prevent humans' task management errors, which will also be the ultimate goal for future studies of this kind. While studying accidents and incidents in several complex systems such as cockpits and nuclear power plants, many researchers noticed a common pattern: human error occurred while managing multiple concurrent tasks. Although these researchers could easily recognize such drastic errors in task management, they have not had a generic prescription on how to prevent them. Is it possible to manage tasks *perfectly* in such environments? This question was the driving force behind this research.

In the real world, people are striving for perfection, but ironically, the moment one speaks of 'perfection' he distances himself from the real world. Fortunately the realm of mathematics offers opportunities for attaining such perfection. In the field of operations research, this perfection is called "optimality." For this reason, operations research as it is applied to *scheduling*, was used to find an optimal method of managing tasks. Typically in the domain of scheduling, jobs are sequenced in a certain optimal order to be processed by machines. This theory was thought to be applicable to sequence tasks in a certain optimal order, to be attended to by an operator such as a pilot. It was soon evident, however, that environments such as a cockpit are too complex to model mathematically. The research was then simplified to finding an optimal method of managing tasks in a much more abstract environment of concurrent task management.

In Chapter 2, it is shown that other researchers, who attempted to find a mathematically optimal way of task management, had primarily focused on maximizing the number of tasks accomplished. It is, however, explained in Chapter

3 that such a performance measure does not apply to many multi-tasking environments. In these environments, a task's status varies repeatedly between poor and desired status depending on how much attention it gets, and a task never gets accomplished to leave the system. Instead of the number of tasks accomplished, a more appropriate objective for these situations seemed to be maximizing the operator's *quality* of performance. Hence, the metaphor of 'Juggler and spinning plates' was introduced in Chapter 4 to resemble an operator (a juggler) who manages several tasks (plates on vertical sticks) concurrently. Moreover, this metaphor with a graphic bar representation allowed to quantify parameters of the task management environment. This quantification paved the way for the mathematical model to capture this environment. The main elements quantified are:

1) Satisfaction Level (SL): the status of a task between zero and one.
2) Correction Rate (CR): the constant rate that a task's SL improves while attended.
3) Deviation Rate (DR): the constant rate that a task's SL deteriorates while not attended.
4) Importance, weight, or value (w): the constant relative worth of a task compared to other tasks.

Several mathematical models were offered in Chapter 5 to capture this environment. Although similar in concept, these models mostly differed in their objective functions, and the tasks' behaviors at extreme SLs. Only one of the models, which was thought to be the most representative of real-life situations, was used. Nonetheless, every one of these models has the potential to represent a different environment, and to be further studied. All models were formulated as a mixed (binary) integer-linear programming model. In these models, time was considered discretely, and the operator was assumed to attend to, at most, one task at a time, which is equivalent to the single machine scheduling. The model selected for the purpose of this research had an objective function of maximizing the

average satisfaction level across tasks over time. In this model, tasks were deteriorating if not attended, while only one task improved if attended to; neither tasks could exceed a SL of one, nor fall below a SL of zero. Tasks were, however, penalized linearly for the length of time that they stayed at a zero SL. This penalty was to discourage the operator from letting tasks stay at a very low satisfaction level.

The computational complexity of the mathematical model was proven to be NP-hard in the strong sense (Appendix B). Hence, an implicit enumeration technique such as the branch-and-bound could take an unreasonably long computation time for solving large size problems. To overcome this inefficiency, it was decided to apply a meta-search heuristic, known as tabu search, to the problem at hand. Although such heuristics do not guarantee optimality, they usually provide good near-optimal solutions in a relatively short time. The structure and steps of this heuristic are thoroughly explained in Chapter 6.

There are many ad hoc studies on the performance of tabu search. For the purpose of this study too, the performance of tabu search based heuristics also had to be evaluated. In Chapter 7, the quality of solutions found by tabu search, their time efficiency, and factors affecting their quality were investigated. Problems were divided into three different sizes of small, medium, and large based on their difficulty in finding the optimal solution by the branch-and-bound technique.

9.1 FACTORS AFFECTING SOLUTION QUALITY OF TABU SEARCH

A split-plot experiment was designed to investigate the effects of different tabu search methods and different initial solution finding mechanisms on the quality of tabu search solutions. Initial experiments had ruled out the possibility of any effect by these factors on the time that it takes for tabu search to find the best solution.

Thus, the only response variable in the experimental design was the objective function value of the solution. Three different levels for tabu search methods were considered based on their use of long-term memory. Also, two different levels considered for the initial solution finding mechanism included one completely random solution, and one weighted random solution based on the weight (value) of the tasks. The experiment concluded no significant difference between the factors in any of the problem sizes with 95% confidence, except that a 'Fixed tabu size, LTM-Max' method performs significantly better than a 'Fixed tabu size, no long-term' in large size problems. Differences in the initial solution finding mechanisms were concluded to be insignificant by the experimental design. However, when very large size problems were later solved, the differences between the initial solution finding mechanisms became more apparent, which one might consider for future research and experimentation.

9.2 SOLUTION QUALITY AND TIME EFFICIENCY OF TABU SEARCH

Table 9.1 shows that in small to large size problems, tabu search proved to be very time efficient and found the best solution in a very short time. Branch-and-bound technique could find the optimal solutions in no time for the small size, in less than eight hours for easier problems in the medium size, and could not find the optimal solutions for large size problems, even when unreasonably long computation time was allocated. Tabu search, on the other hand, gave the best solution in no time for small problems, a few seconds for medium problems, and in less than a minute for large size problems.

In small size problems, the best solutions found by tabu search were all optimal. For a few easier problems in the medium size, branch-and-bound technique could find the optimal solution (unlike tabu search) with an average time of 302 minutes. Although tabu search did not find the optimal solution for these problems, it gave

solutions that were only within 2% of the optimal, but 9000 times faster (less than two seconds!) For the medium size, it was also shown how to estimate an upper bound by a technique known as Lagrangian relaxation. Such upper bounds are helpful in evaluating tabu search solutions in the absence of knowing the optimal solution. For the sample problems, it took an average of 45 minutes to calculate their upper bounds assuming that optimal solutions were not known, which is 15% of the time that it actually took for finding their optimal solutions. Although tabu search solutions were found in less than two seconds, which is 1300 times faster than finding the upper bound, their objective function values were only within 11% of the upper bound.

Finding an optimal solution for the large size problems by branch-and-bound technique could take days, so it was not pursued. Therefore, it is not possible to evaluate the quality of tabu search solutions for the large size problems. However, looking at the remarkable performance of tabu search in small and medium size problems, it is believed that the large size solutions also could not be too far from the optimal. Table 9.1 summarizes the time efficiency and solution quality of the tabu search based algorithm in comparison with that of the branch-and-bound technique.

The fourth category of problems was later introduced, and called 'very large' problems. This category of problems was solely created to address problems with exceedingly large planning horizons. The motivation was to solve problems with five-minute planning horizons, as that was the length of time the human subjects were intended to experiment with later. Since human subjects were allowed to switch between the tasks as fast as one tenth of a second, a five-minute planning horizon translated into a 3000-time-unit planning horizon. Even tabu search could not give a timely and reasonable solution for such a huge problem. A conjecturing method was introduced to solve such problems by scaling the problem down to a

reasonable size: solving it by the tabu search, and rescaling it back up to its original size. For the sample size problems with smaller planning horizons, this conjecturing technique gave solutions as good as if the problem was solved directly, but in only 0.6% of the time. The quality of the solutions found by conjecturing, for 3000 time unit problems, is not known, but it proved to be sufficiently good when later compared with the performance of human subjects.

Table 9.1 Time comparison between tabu search and branch-and-bound

Problem Size	Time Taken by Branch-and-Bound		Tabu Search
	Optimal Solution	Upper Bound (Largrangian Relax.)	
Small	Quality: optimal Time: fraction of a second	Not needed	Quality: optimal Time: fraction of a second
Medium	Quality: optimal Time: 302 Min on Average for easier problems solved	(Opt. - Upp.) / Opt. %: -10.57% Time: 45 Min on average	(Upp. - Tabu) / Upp. %: 10.63% (Opt. – Tabu) / Opt. %: 1.47% Time: few seconds
Large	Quality: no solution found Time: unreasonably long	Quality: no solution found Time: unreasonably long	Quality: unknown Time: less than one minute
Very Large	Quality: no solution found Time: unreasonably long	Quality: no solution found Time: unreasonably long	*Quality: no solution found *Time: unreasonably long

* Later solved by conjecturing technique using tabu search in a reasonable time

9.3 HUMAN PERFORMANCE

Being able to generate solutions by the tabu search, it was possible to compare the human performance with a near-optimal solution. To measure and analyze human performance, a software environment was introduced in Chapter 8 that allowed a subject to manage several tasks concurrently under different scenarios. Subjects

were instructed to manage the tasks in a manner to maximize the score assigned by the computer. This score was later compared with the near-optimal score. Besides the score, the task management environment also allowed recording the subject's pattern of attendance, which could then be compared with the near-optimal pattern of attendance.

Five different scenarios, based on the combination of CR, DR, and weight (value) of the tasks, were designed. With the use of the conjecturing technique, tabu search was run a few times for approximately 3-4 hours per run. The best of solutions given by tabu search under each scenario was regarded as the near-optimal solution for that scenario.

No subject could gain a score higher than the near-optimal score in any of the scenarios. Subjects gained the best average scores, within 13% and 15% of the near-optimal, in Scenario 4 (different DRs) and Scenario 1 (identical tasks) respectively. On the other hand, subjects had the worst average score, within 30% of the near-optimal, in the most complicated scenario where all the tasks had different parameters. These results indicated that the subjects did not perform even near optimally, and their performance weakened as the complexity of the problem increased. Subjects' maximum best scores, however, ranged within 1% to 15% of the near-optimal in all scenarios. Such good scores imply that subjects can achieve a near-optimal score especially in less complicated scenarios with further practice/training.

Several of the subjects at their early scenarios overreacted to the idea of penalizing the tasks (seemingly "failure") while at the zero SL. This caused poor performance, whereas those subjects who ignored at least one of the tasks had a much better performance. To gain a good score, the subject's selection of the tasks and the number of tasks to attend was much more important than his/her moment-to-

moment decisions on how to attend to tasks. Selection of tasks should be based on higher weight (value), lower DR, and higher CR.

Tabu search solutions seemed to have a stepwise strategy in handling the tasks. That is raising the status of three tasks to a high SL concurrently, and then attending to the fourth and sometimes fifth task, while maintaining the high status of the previous tasks. In contrast, most of the subjects either attempted to handle too many tasks in poor statuses all at once, which resulted in a poor score; or they ignored half of the tasks even while the other half was in good standing, which resulted in a good score if the choice of tasks were correct, but could be improved by attempting to handle the next unattended task.

Nine out of ten subjects performed better in their late scenarios as compared to the early ones. Although this indicates a learning effect, the statistical and general behavioral conclusions are robust and reliable since the order of scenarios was randomized. However, this learning effect has to be considered cautiously when an individual's performance in an individual scenario is looked at. Moreover, those subjects who took the experiment later performed better than those who took it earlier. This serial effect in the data shows the learning effect of the experimenter as he advanced through the experiments. This improvement in the score did not affect the statistical results, but has to be considered if one compares the subjects' performances with each other.

It was observed that the order by which the subjects considered the parameters of a task, had a close relationship with the order by which they could understand (identify) those parameters. The easier it was to identify a parameter, the more it was considered when making decisions on how to attend to tasks. Because of the setting of the software, the decreasing order of simplicity by which a subject could identify and distinguish a parameter between the tasks was: weight (value), DR,

and CR. Understandably, the decreasing order by which the subjects considered the task's parameters in their decision-making was the same: weight (value), DR, and CR.

9.4 FUTURE RESEARCH

There are not many studies reported in the past that relate to the type of research that was addressed here. As a result, there are numerous ways to extend this research to other research topics. In the mathematical end, one can enhance the model's parameters by making CR, DR, or weight (value) time dependent, adding switching costs between the tasks, adding deadlines, including arrival time for new tasks, and so on. One may use other deterministic models such as non-linear programming or even stochastic models such as fuzzy logic, queuing theory, or simulation. In the computational end, one can investigate the application of parallel computer processing in tabu search or use other heuristic methods such as genetic algorithms or simulated annealing.

In the task management environment for the human, one can create many different experiments, which may include investigating the effects of salience of stimuli, proximity of tasks, discrete and/or stochastic task behaviors instead of continuous increase/decrease in their SL, easy scenarios that have manageable number of tasks with reasonable CR/DR rates versus difficult scenarios, etc. However, the most critical experiment, in this end, will be validating how closely this environment resembles real life task management. In the subject's side of it, one can investigate the effects of gender, age, expertise in different multi-tasking fields, race, culture, sleepiness, etc. In the objective function and scoring mechanism, also, there are many ways to create new experiments some of which are listed by different models in Chapter 5. These models are mostly created by variations in how tasks are penalized, and how they behave at extreme satisfaction levels.

9.5 FINAL COMMENTS

This research was no different from many other studies in that it created more questions than it answered, and in answering questions, it opened more roads than it closed. What is task management? What is perfection in task management? Can humans manage tasks perfectly? If not, how well can they? Can mathematical methods outperform humans? Are mathematical methods as good in practice as in theory? Which methods do perform better? And finally, how similar or different is human performance to these methods?

While this study has answered these questions one way, other perspectives may answer them differently. It is with the accumulation of such questions and answers from different perspectives in different fields that the great wall of science is built. This research was just another brick in the wall.

BIBLIOGRAPHY

Alidaee B. and Womer N. K. (1999) Scheduling with time dependent processing times: Review and extensions. *Journal of the Operational Research Society*, **50**, 711-720.

Barnes, J. W., Laguna, M. and Glover, F. (1995) An overview of tabu search approaches to production scheduling problems. *Intelligent Scheduling Systems* (Brown, D. E., and Scherer, W. T., Eds.), pp. 101-127.

Boeing Company (1997) *Statistical Summary of Commercial Jet Aircraft Accidents, Worldwide Operations, 1959-1996*, Airplane Safety Engineering (B-210B), Boeing Commercial Airplane Group, Seattle, WA.

Carbonell, J. R., Ward, J. L. and Senders, J.W. (1968) A queuing model of visual sampling: experimental validation. *IEEE Transactions on Man-Machine Systems*, **9**, 82-87.

Chou, C. (1991) *Cockpit Task Management Errors: A Design Issue for Intelligent Pilot-Vehicle Interfaces*, Doctoral Dissertation, Oregon State University.

Chou, C., Madhavan, D. and Funk, K. (1996) Studies of Cockpit Task Management Errors. *The International Journal of Aviation Psychology*, **6**, 307-320.

Chu, Y.-Y. and Rouse, W. B. (1979) Adaptive allocation of decision making responsibility between human and computer in multitask situations. *IEEE Transactions on Systems, Man, and Cybernetics*, **9**, 769-777.

Diamond, W. J. (2001) *Practical Experiment Designs for Engineers and Scientists*, 3rd edn., John Wiley & Sons, Inc., New York, NY.

Fisher, M. L. (1981) The Lagrangian relaxation method for solving integer programming problems. *Management Science*, **27**, 1-18.

Fisher, M. L. (1985) An Applications Oriented Guide to Lagrangian Relaxation. *Interfaces*, **15**, 20-21.

Garey, M. R. and Johnson, D. S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completenes*, W.H. Freeman and Company, New York.

Glover, F. (1986) Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, **13**, 533-549.

Glover, F. (1989) Tabu Search—Part I. *ORSA Journal on Computing*, **1**, 190-206.

Glover, F. (1990a) Tabu Search: A Tutorial. *Interfaces*, **20**, 74-94.

Glover, F. (1990b) Tabu Search—Part II. *ORSA Journal on Computing*, **2**, 4-32.

Greenberg, H. J. (1996-2002) *Mathematical Programming Glossary*, World Wide Web, <http://www.cudenver.edu/~hgreenbe/glossary/glossary.html>.

Greenstein, J. S. and Rouse, W. B. (1982) A model of human decision making in multiple process monitoring situations. *IEEE Transactions on Systems, Man, and Cybernetic*, **12**, 182-193.

Hicks, C. R. and Turner, K. V. (1999) *Fundamental Concepts in the Design of Experiments*, 5th edn., Oxford University Press, New York, NY.

Hillier, F.S. and Lieberman, G. J. (1995) *Introduction to Operations Research*, 6th edn., McGraw-Hill, Inc.

Hyper LINGO 4.0 (1998) Chicago, LINDO Systems Inc., <http://www.lindo.com>.

Kunnathur, A. S. and Gupta, S. K. (1990) Minimizing the makespan with late start penalties added to processing times in a single facility scheduling problem. *European Journal of Operational Research*, **47**, 56-64.

Logendran, R. and Sonthinen, A. (1997) A Tabu search-based approach for scheduling job-shop type flexible manufacturing systems. *Journal of the Operational Research Society*, **48**, 264-277.

Microsoft Excel 2000. (1985-1999) Microsoft Corporation.

Microsoft Visual Basic 6.0. (1987-1998) Microsoft Corporation.

Montgomery, D. C. (2001) *Design and Analysis of Experiments*, 5th edn., John Wiley & Sons, Inc., New York, NY.

Moray, N, Dessouky, M. I., Kijowski, B. A. and Adapathya, R. (1991) Strategic Behavior, Workload, and Performance in Task Scheduling. *Human Factors*, **33**, 607-629.

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

Pattipati, K. R. and Kleinman, D. L. (1991) A review of the engineering models of information-processing and decision-making in multi-task supervisory control. *Multiple Task Performance* (Damos, D.L., ed.), Taylor and Francis Inc, Bristol, PA, pp. 35-68.

Pattipati, K. R., Kleinman, D. L., and Ephrath, A. R. (1983) A Dynamic Decision Model of Human Task Selection Performance. *IEEE Transaction on Systems*, Man and Cybernetics, **13**, 145-166.

Pew, R. W. and Mavor, A. S., Eds. (1998) *Modeling Human and Organizational Behavior, Applications to Military Simulations*. National Research Council, National Academy Press, Washington D. C., pp. 120-122.

Ramsey, F. L. and Schafer, D. W. (2002) *The Statistical Sleuth: A course in Methods of Data Analysis*, Duxbury Press, Belmont, CA.

Reason, J. (1990) *Human Error*, Cambridge University Press, New York, NY.

Rouse, W. B. (1980) *Systems Engineering Models of Human-Machine Interaction*, Elsevier North Holland, Inc., New York, NY.

SAS: The SAS System for Windows, Release 8.02 TS Level 02M0, SAS Institute Inc., Cary, NC.

STATGRAPHICS PLUS for Window Version 5.0, Statistical Graphics Corp.

Subur, F. (2000) *A Methodology for Real-Time Scheduling of Jobs with Splitting on Unrelated Parallel Machines*, Master's thesis. Oregon State University.

Tulga, M. K. and Sheridan, T.B. (1980) Dynamic decisions and workload in multitask supervisory control. *IEEE Transaction on Systems, Man and Cybernetics*, **10**, 217-232.

Walden, R. S. and Rouse, W. B. (1978) A queuing model of pilot decisionmaking in a multitask flight management situation. *IEEE Transactions on Systems, Man, and Cybernetics*, **8**, 867-874.

Weber, D. C. and Skillings, J. H. (2000) *A First Course in the Design of Experiments: A Linear Approach*. CRC Press, Boca Raton, FL.

APPENDICES

APPENDIX A: OTHER CONCEPTUAL APPLICATIONS OF THE JUGGLER METAPHOR

A.1 DEADLINE

Deadlines can be defined in two different ways: attention deadline and completion deadline.

- Deadline to attend to a task (*attention deadline*): This means that a task has to be attended to before a certain point in time.

  $attention\_deadline$ = the time before which a task has to be attended to

  In the plate metaphor, if the objective is to avoid a plate crash, attention deadline equals to how long it takes for a plate to fall if *not* attended to. If the current satisfaction level of a task is SL, the attention deadline with the above assumptions is:

  $$attention\_deadline = \frac{SL}{DR}$$

- Deadline to complete a task (*completion deadline*): This means that a task has to be completed before a certain point in time. However, *completion* of a task does not have a clear meaning in the plate metaphor or multi-tasking environments in general. Completion can be defined as the resumption of a certain level of satisfaction, so processing time means the time that it takes for a task to be attended to until completed. Obviously, if a task is attended to continuously, it will be completed faster versus switching back and forth between the tasks. Completion and attention deadlines are not independent.

Recall that attention deadline is about as late as one can wait before attending to a task. Therefore, completion deadline could not be earlier than attending to a task just before its attention deadline and processing it continuously until completion, or resumption of the desired SL. In the following formula, 'processing time' in fact means 'the time that it takes a task to be completed since it is attended to continuously just before its attention deadline.'

$$earliest\_completion\_deadline = \text{processing time} + \text{attention deadline}$$

- Suppose that completion means resumption of the perfect status. The time that it takes to bring a task from zero to %100 SL if continuously attended to, plus the deadline to attend to a task (attention deadline), would be the *earliest completion deadline*. If the current satisfaction level of a task is SL, the earliest completion deadline with the above assumptions is:

$$completion\_deadline = \frac{1-SL}{CR} + \frac{SL}{DR}$$

In either attention or completion deadline, there has to be some kind of a penalty for not meeting them. The penalty might be as low as zero (i.e., the deadline can be passed without incurring any cost) or a very high penalty that causes the whole system of tasks to crash just by not meeting that one deadline. For example, when driving the deadline for turning before a sharp turn next to the ocean has a very high and deadly penalty that causes the driver-car system crash into the ocean. On the other hand, the penalty for not meeting the deadline for not running out of gas is the discomfort of staying next to the road, but its not deadly. Finally, the deadline to roll the window up when the weather suddenly changes to rainy is the time that the driver has before he passes the rainy portion of the road. If the driver does not meet the deadline, rolling the window up is of no use anymore, because the weather

is no longer rainy. The penalty for not meeting this deadline is a very minor discomfort compared to the previous two scenarios.

A.2 URGENCY

"Urgency" is implicitly included in the mathematical model for the primary objective function (Chapter 5) by penalizing a task the moment it reaches the zero SL. The closer tasks are to the zero SL, the more urgent they are. Therefore, not reaching the zero SL, in that environment, has a deadline that needs to be met. The following is a generic definition of the urgency and its relationship to deadline.

In general, the term *urgency* seems to be inversely related to the deadline. The closer a task is to its deadline or the smaller the *time opportunity window* is, the more *urgent* that task is.

$$urgency \propto \frac{1}{\text{closeness to deadline}}$$

With the above definition for the attention deadline, the following is true:

$$urgency \propto \frac{1}{SL/DR} = \frac{DR}{SL}$$

As it is seen in the above formula, the factors that contribute to the urgency of a task are its current satisfaction level (SL) and its deviation rate (DR) from the perfect status.

A.3 PRIORITY

The priority of a task to be attended specifies its position in the sequence of attending to tasks. This priority is determined in terms of the contribution of a task

to the overall goal, and not its urgency, importance, or status per se. For two identical tasks that have equal CR, DR, importance, and penalty for not meeting the deadline, the one that is more urgent has a higher priority also. However, if the previous parameters are different, priority cannot be found easily because it has to assure the best sequence that maximizes the performance over time. The optimal solution to the mathematical model is the sequence of highest priority tasks at each point in time to attend to, in order to maximize the overall performance. Finding such a perfect sequence that points out the highest priority task at each point in time throughout the planning horizon is very challenging, if not impossible, for a human operator.

A.4 SALIENCE OF TASK RELATED STIMULI

Ideally, this factor (*salience of stimuli*) is the same as *importance* defined earlier. However, this is not always true. A perfect human operator has a perfect knowledge of the task's importance. However, an imperfect human operator has to perceive the importance of a task, which might be influenced by the task's salience.

$$\text{perceived importance} = f(\text{importance, salience})$$

Perceived importance is influenced by the actual importance and its salience, whereas importance is the actual contribution that a task makes to the overall goal. An example would be to coat a metal plate with gold. The perceived importance of this plate would be equally important to a real golden plate.

A.5 AUTOMATION

A task may be performed manually or by an automated system. Automation can be divided into the following:

- Automation of attention: This means that the attention is allocated to the most urgent, or to the highest priority task (plate) automatically.
- Automation of performance: This means that as soon as a task is attended to once (automatically or manually), its satisfaction level increases automatically. The alternative would be the case when continuous attention is required to increase the satisfaction level. For example, modern gas nozzles are pushed once and they fill the vehicle's tank and stop automatically when it is filled, whereas the old ones required continuous push on the nozzle to fill the tank.
- Automation of hold: This type of automation keeps a task at a certain level of satisfaction without being attended to. e.g., cruise control of a car.

A.6 ARRIVAL OF NEW TASKS

A task that is and stays at its desired status with no attention is of no concern, and can be considered nonexistent. However, when a task's SL suddenly decreases to a level below 100%, it needs attention to resume the 100% level. Therefore, it can be assumed that a new task has been added to the system. For example, a pilot may be keeping an altitude of 20,000 feet according to the previous orders in order to maintain a SL of 100% for the task 'altitude.' As soon as the ATC asks for a change in altitude to 30,000 feet, the satisfaction level of 'altitude' task drops to a level below 100% until the pilot attends to this task and reaches the 30,000 feet.

A.7 MONITORING

Monitoring means assessing parameters of the system (such as SL, CR, DR, and importance) and continuously detecting any changes in them. Tasks normally stay steady at a certain level of satisfaction if they are fixed. However, in some cases they might deviate from where they are, so they need to be monitored. e.g., a car's speed is fixed by cruise control (SL = 100%), but for unknown reasons the device loses its control and SL starts decreasing.

A.8 SOME EXAMPLES

A.8.1 Example 1: Turbulence

"A pilot is trying to maintain certain heading, altitude, and speed when sudden turbulence causes an altitude deviation."

This will be equivalent to a sudden increase in the deviation of the task altitude from its perfect status. Therefore, the SL in this task suddenly drops and it requires attention to resume its perfect status. If properly monitored, the new SL of this task is detected and it is attended to according to its priority.

A.8.2 Example 2: Equipment Failure

"A pilot is maintaining heading, altitude, and speed when a critical equipment failure occurs. While attending to the failure, environmental conditions tend to move the aircraft away from the desired heading, altitude, and speed."

This is equal to a sudden increase in the deviation of a very important task, which is normally steady at its perfect status even when it is not being attended to. The distinction of this task (malfunction) from the other routine tasks is that the correction rate is unknown. i.e., the pilot does not know how fast he/she can fix the problem. The parameters of all the tasks should be reassessed frequently while fixing the problem. Detection of failure is not addressed in this research and operators are assumed to be informed of the failure by obvious signals.

A.8.3 Example 3: ATC New Altitude

"A pilot is satisfactorily controlling the aircraft's altitude when the air traffic controller advises him/her to descend to a new altitude."

This is treated exactly the same way as the case when there is turbulence. The new desired altitude, required by the ATC, suddenly increases the deviation of the task altitude. Consequently, the urgency of the task increases. The distinct difference between this scenario and the turbulence is that the sudden increase in deviation that is caused by the ATC advice is a routine practice while flying. Turbulence and malfunction, however, might have some extra psychological impacts on the pilot due to their rare occurrence. The psychological impact of the problem is not addressed in this research.

APPENDIX B: PROOF FOR NP-HARDNESS OF THE MATHEMATICAL MODEL

B.1 GENERALIZED TRAVELING SALESMAN PROBLEM (GTSP)

In the Generalized Traveling Salesman Problem, the salesman has to find the shortest distance to travel among *m* countries and return to where he started. Each country has *n* cities and only one city within each country can be visited. This problem is a known NP-hard problem in the strong sense. The following graph shows two feasible paths for four countries.

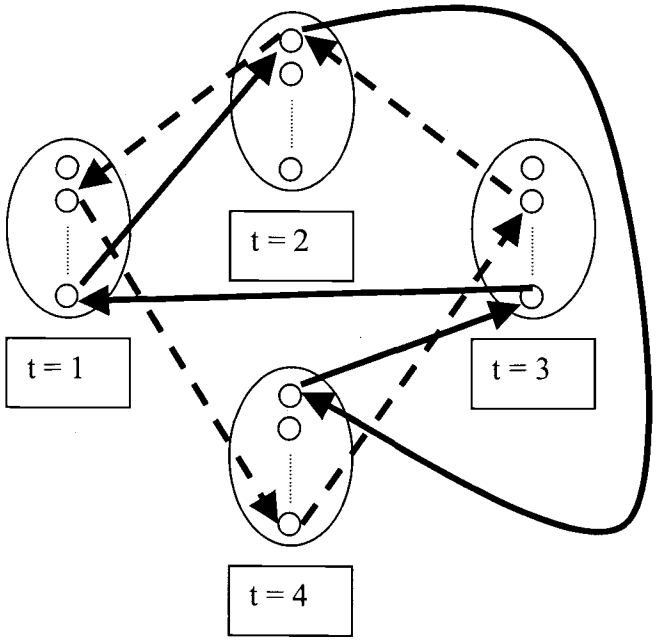


Figure B.1 Two feasible paths for traveling between cities of four countries

In the above problem, suppose the order of visiting the countries is predetermined. Also, suppose that the cost of moving to city *j* from any city of the previous country is equal and it does not change across the countries. There will be a polynomial time algorithm solution if the previous conditions hold true (Appendix C). Assume the salesman currently resides in city *j* of country *i*, and is deciding to which city in country *i*+1 (because the order is predetermined) to move. With the conditions explained above, his best choice would naturally be to pick the shortest path from his current location to any city in the next determined country (*i* + 1).

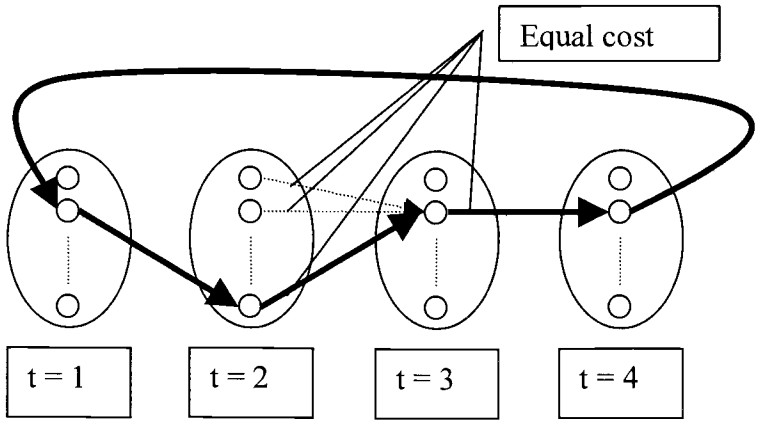


Figure B.2 Traveling between the cities of four countries with predetermined order of countries and equal cost of traveling to a city

B.2 REDUCTION OF TASK MANAGEMENT PROBLEM (TMP) TO GTSP

The task management problem (TMP) in this research can be reduced to GTSP formulation by assuming that the countries are points in time. Cities within each country can be considered as tasks, only one of which can be attended to at any point in time. The distance vector between cities of different countries would be the gain in the task management scoring system by attending to a task. Unlike the

GTSP, in the TMP the goal is to maximize the cumulative magnitude of vectors in the course of the salesman's trip.

If there were no limitations on the maximum (1) or minimum (0) SL that a task can take, the TMP problem had a polynomial time algorithm solution. This exactly matches with the special case explained previously in the GTSP with predetermined order of traveling between the countries, and equal cost for traveling from cities a country to a specific city in the next country. Since in the reduction of TMP, countries are points in time, the order of attendance between them must be in strict natural sequential order. One can only move from time zero to time 4 by passing through times 1, 2, and 3 sequentially. On the other hand, the gain of attending to a task is independent from the task that was previously being attended, and it does not change over time. Thus under these conditions, both the TMP and the GTSP can be solved in polynomial time.

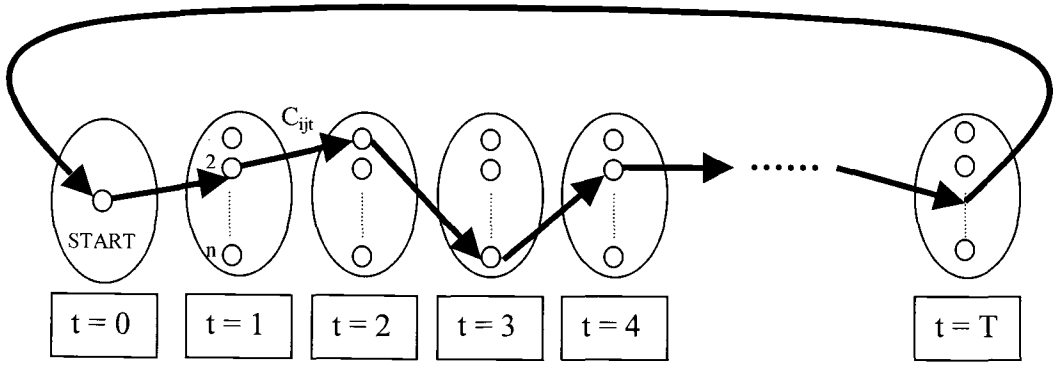


Figure B.3 Reduction of TMP to GTSP with predetermined order of countries if there were no boundary limitations for maximum/minimum SL of tasks

Introducing the boundary limitations for maximum/minimum SL of tasks, would increase the complexity of the TMP immensely. The cost of traveling between

cities of different countries (or equivalently the gain of going from task to task across time) would no longer have a unique magnitude. In fact, the number of possible values for the cost/gain vector will grow exponentially with the number of countries (or expansion of planning horizon). It is the decision among these exponentially growing cost/gain alternative vectors, instead of a unique value, that results in NP-hardness of the TMP with the SL boundary conditions.

In the following example, it is selected $n = 4$ (number of tasks) to describe how the complexity of the problem grows as the planning horizon gets larger. Assume that the bold path is the optimal path and note that a null task is added for not attending to any task that results in $n+1 = 5$ cities in each country.

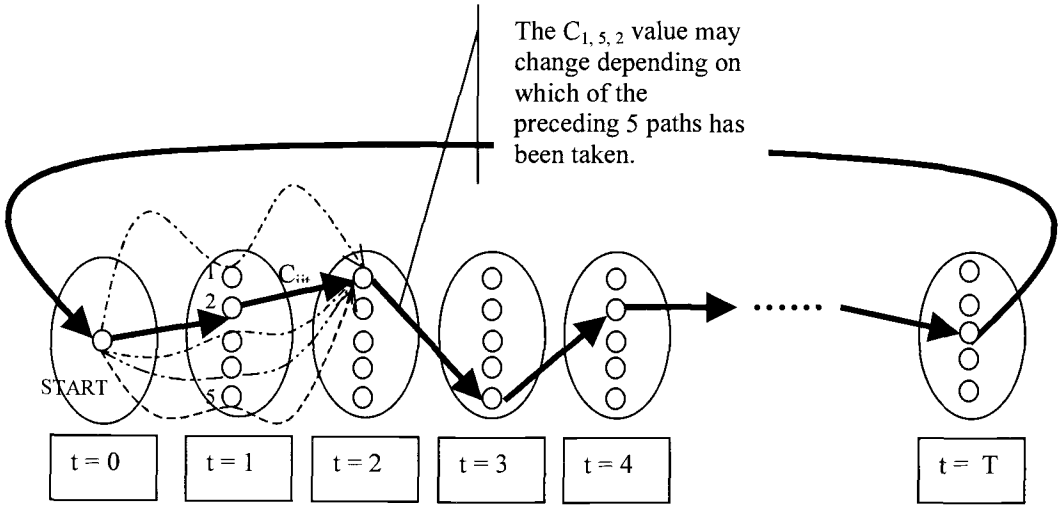


Figure B.4 TMP if boundary limitations for maximum/minimum SL of tasks were introduced

- T : Planning horizon

- n: Number of tasks. An additional task is accounted for a null task when no task is attended to. So the total number of cities would be $n + 1$.
- C_{ijt} is the gain (loss) of the scoring system by switching attention from task i to task j at time t-1 for one time period.

The C_{ijt} value varies depending on the whole path taken that has ended in task i at time t -1. The following table shows that by adding one more time unit, the number of paths to consider grows by a multiplication of $n = 5$. This clearly indicates the exponential growth of the problem in hand. In general, the number of paths to consider for calculating C_{ijt} at any time t-1 can be expressed as $(n+1)^{(t-2)}$, $t > 1$.

Table B.1 Exponential growth of the complexity of TMP

t	Number of Paths to Consider for Calculating $C_{i,j,t}$
1	0
2	1
3	5
4	25
5	125
T	$5^{(T-2)}$

If there were only the maximum (100%) SL constraints (i.e., no non-negative SL limit), C_{ijt} would be only influenced by the frequency of attendance to task j in the prior path taken. With both maximum and minimum SL constraints, however, all tasks in the previous time units (i.e., 0, 1, …, t-1) could affect a C_{ijt} value whether or not they have been attended to. To better match TMP with GTSP and graph

theory requirements, there has to be a cost/gain of C_{ijt} for any travel between the countries. To remove the unreasonable paths, the cost can be chosen very high, so it will never be selected. This applies to trips in the opposite direction of time (e.g., time t to time t-1) or skipping a number in the sequence of time (e.g., time t to time $t + 2$). Also, a null state is defined for $t = 0$ with a zero traveling cost from $t = T$ to $t = 0$, so the graph would be a complete cycle.

Since the computational complexity of the problem in hand is proven to be NP-hard in the strong sense, it should be solved using efficient heuristic algorithms (Chapter 6).

APPENDIX C: OPTIMAL RULE FOR MAXIMIZING THE TOTAL WEIGHTED AVERAGE SL ACROSS TASKS— SPECIAL CASE

C.1 INTRODUCTION

Recall the environment of the simple objective function in which tasks can repeatedly reach 100% SL or zero SL. The objective was to maximize the average SL across tasks. If the boundaries for SL are removed, there is an optimal rule for achieving the above objective function. That is, if tasks can have higher than 100% SL or lower than zero SL, the rule presented below is optimal.

C.2 THEOREM

Assuming that tasks can hold higher than 100% SL or lower than zero SL, the optimal order of attendance for maximizing the total weighted average SL across tasks over time is to attend to the task with the highest weighted summation of correction rate and deviation rate (i.e., $w(CR + DR)$) at all times.

C.3 PROOF

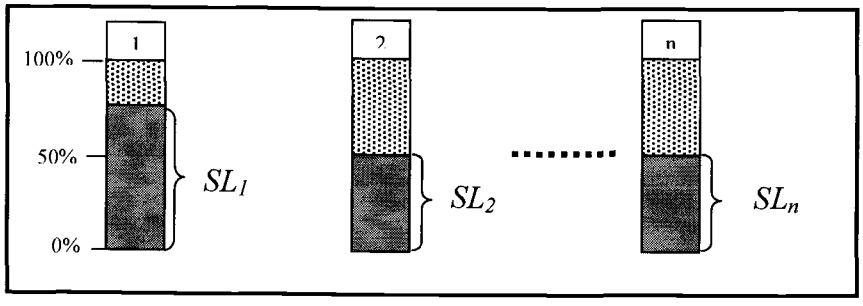


Figure C.1 Current status of tasks

Current value of the system: $\sum_{i=1}^{n}(w_i)SL_i$

Assume that task j is attended to for one time unit; now the value of the system is:

$$w_1(SL_1 - DR_1) + w_2(SL_2 - DR_2) + ... + w_j(SL_j + CR_j) + ... + w_n(SL_n - DR_n)$$

which is equal as:

$$w_1(SL_1 - DR_1) + w_2(SL_2 - DR_2) + ... + w_j(SL_j - DR_j + DR_j + CR_j) + ... + w_n(SL_n - DR_n)$$

which can be simplified to:

$$\underbrace{w_j(CR_j + DR_j)}_{\text{The larger this term, the higher is the increase in value}} + \underbrace{\sum_{i=1}^{n} w_i(SL_i - DR_i)}_{\text{Constant regardless of the order of attendance}}$$

As it can be seen the second term is constant regardless of the task that is attended to. Therefore, the first term is the only one that increases the value of the system. The larger this term, the higher is the increase in value of the system. Because the parameters of that term (w, CR, and DR) are independent of time and the order of attendance, the task with the highest w(CR+DR) has to be attended to at all times.

APPENDIX D: OPTIMAL RULE FOR MINIMIZING THE TOTAL COMPLETION TIME—SPECIAL CASE

D.1 INTRODUCTION

Recall the scenario where tasks will be accomplished or completed the moment they reach their maximum (100%) satisfaction level (SL). Within this scenario further assume that tasks can have negative SLs and there is no minimum for their SL (previously zero). For the last assumption, if the objective function is to minimize the total completion time of all tasks, there is a one to one correspondence between the optimal order presented below and the optimal order found for the deteriorating jobs presented by Kunnathur and Gupta (1990). In application, one can disregard the negative SL assumption if the tasks have too high statuses and their deviation rates are too low, since it is very unlikely that any of the tasks reach a zero (or negative) SL regardless of their order of attendance. However, in order to present the proof for the optimal rule, we must first prove that preemption is not optimal.

D.2 PARAMETERS AND VARIABLES

TC_i = total completion time for strategy i

C_i = completion time of task i

VP_i = variable processing time of task i when attended to after other tasks are attended

P_i = processing time of task i if attended to at time $t = 0$

CR_i = correction rate of task i

DR_i = deviation rate of task i

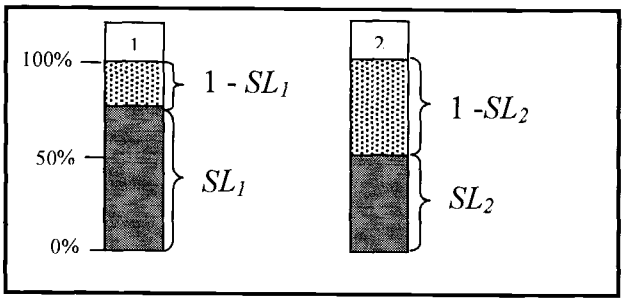


Figure D.1 Status of the tasks at time t = 0

D.3 PREEMPTION IS NOT OPTIMAL

In the following, two strategies are considered with and without preemption. It is proven that the one with preemption could always be improved with a no-preemption strategy for a better total completion time.

Strategy A: Task 1 gets attended to at time zero, and then attendance to task 1 gets interrupted at time *t* in order to attend to task 2 until it is completed, and finally task 1 is attended to again until its completion.

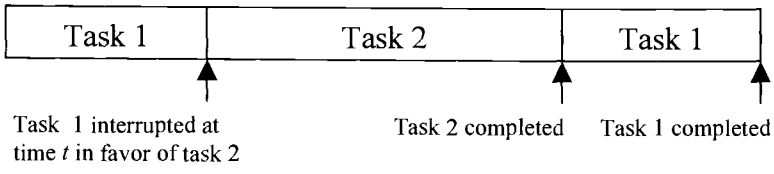


Figure D.2 Order of attendance with preemption.

$$C_2 = t + VP_2 = t + P_2 + t(\frac{DR_2}{CR_2})$$

$$VP_1 = P_1 - t + VP_2(\frac{DR_1}{CR_1}), \text{ or}$$

$$VP_1 = P_1 - t + [P_2 + t(\frac{DR_2}{CR_2})](\frac{DR_1}{CR_1}), \text{ or}$$

$$VP_1 = P_1 - t + P_2(\frac{DR_1}{CR_1}) + t(\frac{DR_2}{CR_2})(\frac{DR_1}{CR_1})$$

$$TC_A = C_1 = C_2 + VP_1, \text{ or}$$

$$TC_A = [t + P_2 + t(\frac{DR_2}{CR_2})] + [P_1 - t + P_2(\frac{DR_1}{CR_1}) + t(\frac{DR_2}{CR_2})(\frac{DR_1}{CR_1})], \text{ or}$$

$$TC_A = P_1 + P_2 + P_2(\frac{DR_1}{CR_1}) + t(\frac{DR_2}{CR_2}) + t(\frac{DR_2}{CR_2})(\frac{DR_1}{CR_1})$$

Strategy B: Task 2 gets attended to at time zero until its completion, and then task 1 gets attended to until its completion.

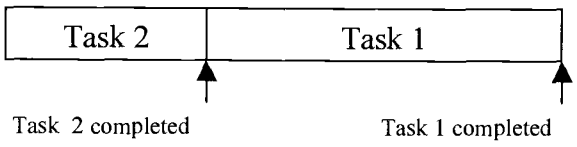


Figure D.3 Order of attendance without preemption.

$$C_2 = P_2$$

$$VP_1 = P_1 + C_2(\frac{DR_1}{CR_1}) = P_1 + P_2(\frac{DR_1}{CR_1})$$

$$TC_B = C_1 = C_2 + VP_1 = P_2 + P_1 + P_2(\frac{DR_1}{CR_1})$$

It can be seen that Strategy B has a better completion time than Strategy A.

$$TC_A - TC_B = t(\frac{DR_2}{CR_2}) + t(\frac{DR_2}{CR_2})(\frac{DR_1}{CR_1})$$

So it can be concluded that preemption is not optimal. Therefore, in finding the optimal sequence of tasks in the following theorem, tasks are attended to with no preemption.

D.4 THEOREM

Assuming that tasks can hold negative satisfaction levels, the optimal sequence of tasks for minimizing the total completion time of all tasks, must satisfy the following requirement:

$$(1-SL_1)/DR_1 \leq (1-SL_2)/DR_2 \leq \ldots \leq (1-SL_n)/DR_n$$

D.5 PROOF

This theorem is proven by induction. First, it is required to prove that the theorem holds true for k = 2 tasks. Second, it is supposed that the theorem holds true for k tasks, and then its validity is proven for k + 1 tasks. Thus, the theorem will be true for any number of tasks.

D.5.1 Proof for k = 2 Tasks

Theorem: Assuming that tasks can hold negative satisfaction levels, the optimal sequence of tasks for minimizing the total completion time of **k = 2 tasks**, must satisfy the following requirement:

$$(1-SL_1)/DR_1 \leq (1-SL_2)/DR_2 \quad (0)$$

The problem is solved with both possible orders of attendance, and then it is shown that the order that satisfies inequality (0) has the shorter total completion time.

Strategy a: Attend to task 1 first and task 2 second.

$$\begin{aligned}
&\mathrm{TC}_{\mathrm{a}} = C_2 \\
&C_1 = P_1 = \frac{(1-SL_1)}{CR_1} \\
&C_2 = C_1 + VP_2 \\
&VP_2 = P_2 + \frac{(C_1)(DR_2)}{CR_2} \\
&P_2 = \frac{(1-SL_2)}{CR_2}
\end{aligned}$$

The total completion time (TC_a) for Strategy a:

$$TC_a = C_2 = P_1 + P_2 + \frac{(C_1)(DR_2)}{CR_2}$$

Strategy b: Attend to task 2 first and task 1 second.

$$\begin{aligned}
&\mathrm{TC}_{\mathrm{b}} = C_1 \\
&C_2 = P_2 = \frac{(1-SL_2)}{CR_1} \\
&C_1 = C_2 + VP_1 \\
&VP_1 = P_1 + \frac{(C_2)(DR_1)}{CR_1} \\
&P_1 = \frac{(1-SL_1)}{CR_1}
\end{aligned}$$

The total completion time (TC_b) for Strategy b:

$$TC_b = C_1 = P_2 + P_1 + \frac{(C_2)(DR_1)}{CR_1}$$

Assuming Strategy a yields a better (shorter) time:

$$TC_a < TC_b$$
$$P_1 + P_2 + \frac{(C_1)(DR_2)}{CR_2} < P_2 + P_1 + \frac{(C_2)(DR_1)}{CR_1}$$
$$\frac{(C_1)(DR_2)}{CR_2} < \frac{(C_2)(DR_1)}{CR_1}$$
$$\frac{(1-SL_1)}{CR_1}\frac{(DR_2)}{CR_2} < \frac{(1-SL_2)}{CR_2}\frac{(DR_1)}{CR_1}$$
$$\frac{(1-SL_1)}{DR_1} < \frac{(1-SL_2)}{DR_2}$$

Thus, the optimal order of attendance to minimize the total completion time for k = 2 tasks has to follow this order:

$$(1-SL_1)/DR_1 \leq (1-SL_2)/DR_2 \quad (0)$$

D.5.2 Proof for k + 1 Tasks if the Theorem Holds True for k Tasks

Induction Assumption: Assuming that tasks can hold negative satisfaction levels, the optimal sequence of tasks for minimizing the total completion time of **k tasks**, must satisfy the following requirement:

$$(1-SL_1)/DR_1 \leq (1-SL_2)/DR_2 \leq \ldots \leq (1-SL_k)/DR_k \quad (1)$$

Theorem: Assuming that tasks can hold negative satisfaction levels, the optimal sequence of tasks for minimizing the total completion time of **k + 1 tasks**, must satisfy the following requirement:

$$(1-SL_1)/DR_1 \le (1-SL_2)/DR_2 \le \ldots \le (1-SL_k)/DR_k \le (1-SL_{k+1})/DR_{k+1} \quad (2)$$

The inequality (2) suggests that the optimal order is {1, 2, …, k-1, k, k+1}. It is evident that swapping the order for any two of the first k tasks $i = 1, 2, \ldots, k$ would not improve the total completion time according to inequality (1). Therefore, the only alternative order to the one suggested by the theorem is {1, 2, …, k-1, k+1, k}. If the new order is proven to be inferior to the one suggested by inequality (2), the theorem is proven.

Assume that completion time for the first k-1 tasks takes C_{k-1} time units. Therefore after finishing the first k-1 tasks, if it is proven that the inequality (3) holds true then according to inequality (0), the proof for k = 2 tasks, the alternative order would be inferior to the one suggested by inequality (2) and the theorem is proven.

$$\frac{(1-SL_{k,C_{k-1}})}{DR_k} \le \frac{(1-SL_{k+1,C_{k-1}})}{DR_{k+1}} \quad (3)$$

Where $SL_{i,t}$ is satisfaction level of task i at time t.

Note that the concept of each side of inequality (3) is the time that it has taken a task to deviate to its current SL from the perfect (100%) SL if unattended. Thus, this time period at $t = C_{k-1}$ has only increased by C_{k-1} time units compared to the time $t = 0$. Therefore, inequality (3) can be simplified as:

$$C_{k-1} + \frac{(1 - SL_{k,0})}{DR_k} \leq C_{k-1} + \frac{(1 - SL_{k+1,0})}{DR_{k+1}}$$

$$\frac{(1 - SL_{k,0})}{DR_k} \leq \frac{(1 - SL_{k+1,0})}{DR_{k+1}}$$

The simplified inequality is part of the order initially assumed by inequality (2) so inequality (3) is proven, and therefore the theorem is proven for k + 1 tasks.

D.5.3 Proof for All Tasks

The theorem is proven to hold true for k = 2 tasks. It is also proven that the theorem holds true for k + 1 tasks if it is assumed to be true for k tasks. Therefore, by induction the theorem is proven for any number of tasks.

If the denominator and numerator of the proven inequality is divided by the task's correction rate (CR), the following inequality will result:

$$\frac{(1 - SL_{k,C_{k-2}})/CR_k}{DR_k / CR_k} \leq \frac{(1 - SL_{k+1,C_{k-2}})/CR_{k+1}}{DR_{k+1}/CR_{k+1}} \leq \ldots \leq \frac{(1 - SL_{n,C_{k-2}})/CR_n}{DR_n / CR_n}$$

The numerator of this inequality is equivalent to the *processing time* and the denominator is equivalent to the *rate* that the processing time of a task increases if not attended to. This inequality has a one-to-one match with the proof presented by Kunnathur and Gupta (1990) for deteriorating jobs on a single machine.

APPENDIX E: DATA GENERATED FOR SMALL, MEDIUM, AND LARGE SIZE PROBLEMS

Table E.1 shows how problems with different sizes are created; and Table E.2 shows how parameters of each task for every problem size is generated.

Table E.1 Specifications of small, medium, and large problem sizes

Problem Size	Number of Tasks	Planning Horizon	Initial SL
Small	[2, 3]	[2, 4]	[0, 1]
Medium	[5, 8]	[5, 10]	[0, 1]
Large	[12, 16]	[11, 20]	[0, 1]

Table E.2 The rule underlying generating parameters of each task for every problem size

Parameter	A number generated from a random uniform distribution in the interval of
weight	[1, 10], integer value
deviation rate	[0.01, 0.10] up to 2 decimal points
correction rate	[1 to 10] multiplied by the corresponding deviation rate
initial satisfaction level for all tasks	[0, 1], up to 2 decimal points

In the following tables (Table E.3-Table E.5), the data generated for the small, medium, and large problem sizes of the experimental design is included.

E.1 SMALL SIZE

Table E.3 Data generated for the small size problems

		Blocks									
Block No:		1	2	3	4	5	6	7	8	9	10
Tasks: [2, 3]		2	2	3	2	2	3	2	3	2	3
Planning Horizon: [2, 4]		3	2	4	2	3	2	2	4	4	2
Initial SL: [0, 1]		0.270	0.663	0.983	0.849	0.008	0.161	0.575	0.147	0.212	0.37
	Block No:	1	2	3	4	5	6	7	8	9	10
Task											
1	CR (1-10)×DR	0.45	0.35	0.16	0.5	0.27	0.7	0.5	0.08	0.9	0.06
	DR (0.01-0.10)	0.05	0.05	0.08	0.05	0.03	0.1	0.1	0.02	0.09	0.01
	Weight (1-10)	8	9	1	3	2	9	8	3	6	4
2	CR (1-10)×DR	0.56	0.02	0.18	0.72	0.24	0.28	0.54	0.12	0.64	0.63
	DR (0.01-0.10)	0.07	0.02	0.02	0.08	0.04	0.07	0.09	0.03	0.08	0.09
	Weight (1-10)	7	2	3	7	5	2	9	9	9	1
3	CR (1-10)×DR			0.1			0.16		0.56		0.4
	DR (0.01-0.10)			0.02			0.08		0.08		0.1
	Weight (1-10)			10			4		8		7

E.2 MEDIUM SIZE

Table E.4 Data generated for the medium size problems

		Blocks									
Block No:		1	2	3	4	5	6	7	8	9	10
Tasks: [5, 8]		6	8	5	5	7	8	5	8	5	7
Planning Horizon: [5, 10]		5	8	6	8	5	6	10	10	9	10
Initial SL: [0,1]		0.126	0.531	0.677	0.163	0.484	0.964	0.837	0.54	0.376	0.94
	Block No:	1	2	3	4	5	6	7	8	9	10
Task											
1	CR (1-10)×DR	0.08	0.4	0.48	0.08	0.36	0.09	0.01	0.1	0.36	0.8
	DR (0.01-0.10)	0.01	0.04	0.08	0.08	0.04	0.01	0.01	0.02	0.06	0.1
	Weight (1-10)	8	4	1	8	4	8	7	5	3	10
2	CR (1-10)×DR	0.4	0.12	0.36	0.28	0.6	0.12	0.16	0.27	0.4	0.49
	DR (0.01-0.10)	0.04	0.03	0.06	0.07	0.06	0.02	0.04	0.03	0.1	0.07
	Weight (1-10)	5	1	10	9	8	1	4	7	9	3
3	CR (1-10)×DR	0.9	0.08	0.24	0.1	0.18	0.16	0.48	0.35	0.06	0.27
	DR (0.01-0.10)	0.1	0.01	0.03	0.1	0.06	0.04	0.08	0.07	0.03	0.09
	Weight (1-10)	10	7	6	8	8	9	5	10	7	9
4	CR (1-10)×DR	0.45	0.6	0.1	0.28	0.35	0.27	0.06	1	0.08	0.1
	DR (0.01-0.10)	0.05	0.06	0.01	0.04	0.05	0.03	0.02	0.1	0.04	0.01
	Weight (1-10)	7	4	1	2	10	8	2	8	5	5

Table E.4 (Continued) Data generated for the medium size problems

	Block No:	1	2	3	4	5	6	7	8	9	10
Task											
5	CR (1-10)×DR	0.16	0.16	0.36	0.3	0.42	0.54	0.7	0.8	0.08	0.6
	DR (0.01-0.10)	0.02	0.02	0.06	0.06	0.07	0.09	0.07	0.08	0.08	0.06
	Weight (1-10)	1	2	10	3	4	7	8	10	2	9
6	CR (1-10)×DR	0.16	0.12			0.12	0.08		0.36		0.4
	DR (0.01-0.10)	0.08	0.06			0.06	0.08		0.09		0.05
	Weight (1-10)	9	1			7	2		6		6
7	CR (1-10)×DR		0.45			0.28	0.18		0.01		0.24
	DR (0.01-0.10)		0.05			0.04	0.06		0.01		0.03
	Weight (1-10)		7			7	2		2		1
8	CR (1-10)×DR		0.3				0.72		0.21		
	DR (0.01-0.10)		0.06				0.09		0.07		
	Weight (1-10)		10				10		1		

E.3 LARGE SIZE

Table E.5 Data generated for the large size problems

		Blocks									
Block No:		1	2	3	4	5	6	7	8	9	10
Tasks: [12, 16]		14	13	16	13	13	14	15	13	16	13
Planning Horizon: [11, 20]		20	12	16	11	20	18	18	15	11	13
Initial SL: [1, 0]		0.348	0.669	0.047	0.365	0.432	0.868	0.796	0.874	0.116	0.253
	Block No:	1	2	3	4	5	6	7	8	9	10
Task											
1	CR (1-10)×DR	0.56	0.30	0.40	0.45	0.08	0.30	0.32	1.00	0.40	0.25
	DR (0.01-0.10)	0.08	0.05	0.10	0.05	0.01	0.05	0.08	0.10	0.05	0.05
	Weight (1-10)	7	8	10	6	9	1	8	1	10	4
2	CR (1-10)×DR	0.70	0.72	0.42	0.10	0.42	0.12	0.18	0.08	0.80	0.27
	DR (0.01-0.10)	0.07	0.08	0.06	0.01	0.07	0.04	0.02	0.04	0.10	0.03
	Weight (1-10)	2	10	7	2	6	3	9	7	2	4
3	CR (1-10)×DR	0.40	0.24	0.08	0.18	0.48	0.32	0.56	0.48	0.56	0.32
	DR (0.01-0.10)	0.08	0.08	0.01	0.02	0.06	0.08	0.07	0.08	0.08	0.04
	Weight (1-10)	5	7	3	9	8	5	1	10	9	4

Table E.5 (Continued) Data generated for the large size problems

Task	Block No:	1	2	3	4	5	6	7	8	9	10
4	CR (1-10)×DR	0.08	0.15	0.08	0.16	0.04	0.45	0.10	0.72	0.50	0.20
	DR (0.01-0.10)	0.01	0.05	0.02	0.04	0.04	0.05	0.05	0.09	0.10	0.05
	Weight (1-10)	4	1	10	7	6	5	3	6	6	6
5	CR (1-10)×DR	0.45	0.63	0.72	0.06	0.24	0.32	0.27	0.08	0.30	0.15
	DR (0.01-0.10)	0.05	0.07	0.08	0.01	0.06	0.08	0.09	0.02	0.05	0.05
	Weight (1-10)	7	1	1	10	5	6	10	5	3	7
6	CR (1-10)×DR	0.12	0.90	0.14	0.56	0.04	0.50	0.90	0.04	0.20	0.25
	DR (0.01-0.10)	0.02	0.09	0.07	0.07	0.01	0.05	0.10	0.04	0.04	0.05
	Weight (1-10)	8	4	1	7	8	1	9	5	3	6
7	CR (1-10)×DR	0.90	0.10	0.60	0.35	0.81	0.12	0.06	0.07	0.48	0.16
	DR (0.01-0.10)	0.10	0.10	0.06	0.07	0.09	0.04	0.06	0.01	0.08	0.08
	Weight (1-10)	2	10	6	1	9	5	2	9	7	10
8	CR (1-10)×DR	0.70	0.02	0.04	0.28	0.72	0.42	0.06	0.12	0.54	0.32
	DR (0.01-0.10)	0.07	0.01	0.01	0.07	0.08	0.06	0.01	0.02	0.09	0.08
	Weight (1-10)	3	3	2	5	1	1	10	10	10	7

Table E.5 (Continued) Data generated for the large size problems

Task	Block No:	1	2	3	4	5	6	7	8	9	10
9	CR (1-10)×DR	0.04	0.48	0.01	0.12	0.10	0.24	0.32	0.48	0.10	0.36
	DR (0.01-0.10)	0.01	0.06	0.01	0.02	0.10	0.08	0.04	0.08	0.05	0.06
	Weight (1-10)	4	10	2	8	10	6	4	7	7	4
10	CR (1-10)×DR	0.36	0.45	0.03	0.40	0.08	0.81	0.49	0.36	0.07	0.90
	DR (0.01-0.10)	0.04	0.09	0.03	0.08	0.04	0.09	0.07	0.06	0.07	0.09
	Weight (1-10)	10	9	4	7	3	9	6	7	10	8
11	CR (1-10)×DR	0.14	0.27	0.05	0.24	0.06	0.30	0.12	0.72	0.70	0.48
	DR (0.01-0.10)	0.07	0.03	0.05	0.08	0.03	0.10	0.06	0.09	0.07	0.08
	Weight (1-10)	4	8	9	7	3	2	5	4	1	7
12	CR (1-10)×DR	0.30	0.36	0.90	0.03	0.42	0.64	0.35	1.00	0.90	0.63
	DR (0.01-0.10)	0.10	0.09	0.09	0.03	0.06	0.08	0.07	0.10	0.09	0.09
	Weight (1-10)	9	7	6	3	6	6	10	10	2	4
13	CR (1-10)×DR	0.18	0.40	0.12	0.45	0.07	0.49	0.21	0.30	0.04	0.80
	DR (0.01-0.10)	0.09	0.05	0.02	0.09	0.01	0.07	0.07	0.06	0.02	0.08
	Weight (1-10)	1	7	4	10	5	8	9	3	3	4

Table E.5 (Continued) Data generated for the large size problems

	Block No:	1	2	3	4	5	6	7	8	9	10
Task											
14	CR (1-10)×DR	0.01		0.10			0.18	0.42		0.45	
	DR (0.01-0.10)	0.01		0.02			0.09	0.06		0.09	
	Weight (1-10)	7		1			7	10		8	
15	CR (1-10)×DR			0.06				0.64		0.72	
	DR (0.01-0.10)			0.01				0.08		0.09	
	Weight (1-10)			1				7		5	
16	CR (1-10)×DR			0.80						0.12	
	DR (0.01-0.10)			0.10						0.06	
	Weight (1-10)			10						5	

In the following table, the tabu result for the small, medium, and large size problems with three levels of tabu algorithms and two levels of random seeds is shown.

Table E.6 Tabu results for small, medium, and large size problems under all treatments; Tabu Algorithm (No_LTM: -1, LTM_Max: 0, LTM_Min: 1) and Random Seed (Uniform: -1, Weighted: 1)

BLOCK	Tabu Algorithm	Random Seed	SMALL	MEDIUM	LARGE
			Tabu Result	Tabu Result	Tabu Result
1	-1	-1	585.5	318.47	355.14
1	0	-1	585.5	318.47	359.65
1	1	-1	585.5	318.47	359.65
1	-1	1	585.5	318.47	375.16
1	0	1	585.5	318.47	375.16
1	1	1	585.5	318.47	375.16
2	-1	-1	843.18	661.21	543.21
2	0	-1	843.18	661.21	544.41
2	1	-1	843.18	664.39	543.21
2	-1	1	843.18	674.79	543.63
2	0	1	843.18	675.18	543.63
2	1	1	843.18	674.79	543.63
3	-1	-1	978.74	832.34	179.01
3	0	-1	978.74	832.34	184.35
3	1	-1	978.74	832.34	190.55
3	-1	1	978.74	832.34	185.69
3	0	1	978.74	832.34	188.59
3	1	1	978.74	832.34	188.13

Table E.6 (Continued) Tabu results for small, medium, and large size problems under all treatments; Tabu Algorithm (No_LTM: -1, LTM_Max: 0, LTM_Min: 1) and Random Seed (Uniform: -1, Weighted: 1)

BLOCK	Tabu Algorithm	Random Seed	SMALL	MEDIUM	LARGE
			Tabu Result	Tabu Result	Tabu Result
4	-1	-1	910.9	216.07	363.86
4	0	-1	910.9	216.07	363.86
4	1	-1	910.9	216.07	363.86
4	-1	1	910.9	216.07	364.38
4	0	1	910.9	216.07	364.38
4	1	1	910.9	216.07	364.38
5	-1	-1	234.85	541.75	380.27
5	0	-1	234.85	541.75	381.5
5	1	-1	234.85	541.75	380.27
5	-1	1	234.85	541.75	370.86
5	0	1	234.85	541.75	370.86
5	1	1	234.85	541.75	370.86
6	-1	-1	438.13	908.66	596.91
6	0	-1	438.13	908.66	604.81
6	1	-1	438.13	908.66	596.91
6	-1	1	438.13	907.84	600.12
6	0	1	438.13	907.84	604.99
6	1	1	438.13	907.84	600.12
7	-1	-1	756.17	862.87	583.43
7	0	-1	756.17	862.75	584.84
7	1	-1	756.17	862.87	583.43
7	-1	1	756.17	863.15	584.94
7	0	1	756.17	863.15	586.53
7	1	1	756.17	863.15	586.45

Table E.6 (Continued) Tabu results for small, medium, and large size problems under all treatments; Tabu Algorithm (No_LTM: -1, LTM_Max: 0, LTM_Min: 1) and Random Seed (Uniform: -1, Weighted: 1)

BLOCK	Tabu Algorithm	Random Seed	SMALL	MEDIUM	LARGE
			Tabu Result	Tabu Result	Tabu Result
8	-1	-1	384.82	634.56	730.47
8	0	-1	384.82	634.56	733.28
8	1	-1	384.82	634.56	731.08
8	-1	1	384.82	649.14	737.05
8	0	1	384.82	649.14	737.05
8	1	1	384.82	649.14	737.05
9	-1	-1	710.24	495.79	122.5
9	0	-1	710.24	495.79	122.5
9	1	-1	710.24	495.79	128.05
9	-1	1	710.24	490.04	134.12
9	0	1	710.24	490.04	134.12
9	1	1	710.24	490.04	134.12
10	-1	-1	559.44	865.09	189.74
10	0	-1	559.44	865.09	196.33
10	1	-1	559.44	868.09	189.74
10	-1	1	559.44	869.93	184.38
10	0	1	559.44	869.93	195.9
10	1	1	559.44	869.93	184.38

APPENDIX F: SUBJECTS' SPECIFICATIONS

Table F.1 Subjects' specifications

Subjects	Gender	Age	Computer Hours/Week	Driver Hours/Week
1	M	24	40	5
2	F	23	13	0
3	M	30	8	1
4	M	29	20	0
5	M	25	9	2
6	M	21	14	3
7	M	23	25	0
8	F	32	40	12
9	M	21	40	0
10	M	22	50	0
Average	0.8	25	25.9	2.3

APPENDIX G: INFORMED CONSENT DOCUMENT

Oregon State University
Department of Industrial and Manufacturing Engineering

I agree to participate in a Ph.D. dissertation research conducted under the supervision of Dr. Kenneth Funk of the Oregon State University, Department of Industrial & Manufacturing Engineering. After approximately half an hour of introduction, training, practice, and gaining an acceptable level of competence determined by the researcher, I will be asked to play a simple computer game and answer a brief questionnaire after finishing the game. The total length of the experiment should not be more than 2 hours.

I understand that my participation in this project is completely voluntary and that I will not be paid for this voluntary participation. I may withdraw from this study at anytime without any penalty and I may decline to answer any questions if I choose.

I understand that my participation could cause me only minimal risk, inconvenience, or discomfort. That is, I understand that the probability and magnitude of harm, inconvenience or discomfort anticipated in this study are not greater than those encountered in daily life. I also understand that Oregon State University does not provide compensation or medical treatment in the event that I am injured or harmed in any way as a result of participation in this study.

I understand that all records collected in this survey are available to the research investigators, support staff, and any duly authorized research review committee. I grant Oregon State University permission to reproduce and publish all records, notes, or data collected from my participation, provided that there will be **no association of my name** with the collected data and that confidentiality to the extent permitted by law is maintained unless specifically waived by me.

I understand that I will have the opportunity to ask questions and receive satisfactory answers from Shakib Shakeri. I understand any further questions concerning this survey should be directed to Dr. Kenneth Funk at (541) 737-2357.

If I have questions about my rights as a research subject, I should contact the IRB Coordinator, OSU Research Office, (541) 737-3437, IRB@orst.edu.

My signature below indicates that I have read and that I understand the process described above and give my informed and voluntary consent to participate in this study. I understand that I will receive a signed copy of this consent form.

Signature ..... Date .....

Name, Last name .....

Contact Info. .....

.....

APPENDIX H: INSTRUCTION SHEET READ BY THE SUBJECT

A Comparison of Optimal and Actual Task Management

A computer program named *Tardast* (Persian term for Juggler) will be utilized for the purpose of this experiment. A simplistic view of any multitasking environment can be compared to a juggler (user of this software) who has several tasks on hand and tries to perform satisfactorily in all those tasks. The use of this software may give a better understanding of how human operators allocate their attention among multiple concurrent tasks.

The software interface primarily consists of six tasks, which are shown as bars. If you do not attend to a task, its status starts deteriorating from the satisfactory level with a certain rate. On the other hand, its status improves toward the desired state by a different rate while you attend to it. You can attend to a task by simply depressing a button underneath each task using your mouse. Depending on how well you keep the average status of the tasks over time and whether or not you ignore the more important tasks, a numerical score is computed by the software. The computer also records what task you have attended to at any point in time. This data will be reviewed and investigated in the form of graphs. This will help to discover the pattern of attendance to tasks (if any) used by you. The pattern used by you and other subjects in this experiment will be compared against the near-optimal pattern found through Operations Research techniques.

There are five scenarios for which the data will be gathered. For each scenario, you will be given several short (one minute) practice trials to familiarize yourself before running the main experiment (five minutes). Throughout all the trials your objective will be to gain the maximum score possible. You can choose any strategy that you please if you feel it will get you the best score. This might mean trying to keep all tasks at a fairly satisfactory level, or keeping more important tasks at a very good level and letting less important tasks go, or attending to tasks randomly or sequentially, etc. At the very end of the experiments, you will be asked to answer a short questionnaire.

The satisfaction level (score) of each task can vary between 0 to 1000. For each task the '*Average score*' over time is calculated first. This average is simply multiplied by the task's '*weight*' or value (shown on the screen next to the task with the $ sign) and is called '*weighted Avg. score.*' And finally, the summation of these scores among all the tasks divided by the total weight of all tasks is called '*Total Weighted Average Score.*' This number again varies between 0 to 1000. There is a punishment if a task hits the zero level. For every time unit that a task stays at the zero level, its score is punished by (– 200). However, if a task has a very low value, it might be still beneficial to let it be penalized and attend to more valuable tasks. It is up to your discretion to choose your favorite strategy after practice trials.

Please let the experimenter know if you have any questions.

Good luck!

APPENDIX I: INSTRUCTIONS EXPLAINED TO THE SUBJECT VERBALLY

- Remind the subject that he/she can talk out loud.
- Remind the subject to use the *report* or *replay* features in the software at the end of each practice trial.
- Remind that while moving the mouse pointer between the tasks, the subject should not release the mouse button, until the point it reaches the other task, to keep improving the task status initially attended to. This would decrease the lost time for switching between the tasks.
- Remind that the subject can take a look at the *score* and the bar representing it on the top left of the screen. This would help the subject to see if he/she is actually improving the system status by acquiring a certain strategy.

APPENDIX J: POST EXPERIMENT QUESTIONNAIRE

Subject #:.....
Date:.....

1. Gender? M F

2. Age?

3. How experienced are you in working with computer in terms of hours per week?

4. If you are a driver, how many hours per week do you drive?

5. Are you a pilot? If Yes, how many hours per month do you fly?

6. Did you use a particular strategy for determining which task to attend to? If so, describe it.

7. Do you have any comments concerning this experiment?

APPENDIX K: ESTIMATE OF HOW MANY TASKS CAN BE KEPT AT A HIGH STATUS IN THE LONG RUN

The total correction rate in the system should be at least equal to the total deviation rate to be manageable :

1) $\sum_{t=0}^{T}\sum_{i=1}^{n}[(x_{it})(cr_i)-(1-x_{it})(dr_i)]\geq 0$

2) $\sum_{t=0}^{T}\sum_{i=1}^{n}(x_{it})(cr_i)\geq\sum_{t=0}^{T}\sum_{i=1}^{n}(1-x_{it})(dr_i)$

3) $\dfrac{\sum_{t=0}^{T}\sum_{i=1}^{n}(x_{it})(cr_i)}{\sum_{t=0}^{T}\sum_{i=1}^{n}(1-x_{it})(dr_i)}\geq 1$

If deviation rates are all equal to DR, and if also correction rates are all equal to CR, the following holds true :

4) $\dfrac{CR}{(n-1)DR}\geq 1$

5) $\dfrac{CR}{DR}\geq n-1$

Inequality (5) will be a good estimate for the number of tasks that can be managed in the long run if $\dfrac{\text{CR}}{\text{DR}}$ is replaced by $\dfrac{\text{Average (CR)}}{\text{Average(DR)}}$.

APPENDIX L: STATISTICAL ANALYSIS FOR THE HUMAN PERFORMANCE EXPERIMENT

L.1 SAMPLE SIZE

To find out the appropriate number of samples in an experiment, one has to first determine what statistical test with what precision he/she desires. Refer to the textbook by Diamond (2001) for further details on calculating the sample size. In general, if the mean of a population is going to be compared against a particular number, and if the variance of the population is known, the following formula is used to determine the sample size.

$$N = (U_\alpha + U_\beta)^2 \frac{\sigma^2}{\delta^2}$$

where N = sample size, σ^2 = population variance, δ = increment of importance, U_α = normal distribution number for alpha risk, and U_β = normal distribution number for beta risk.

The number of subjects was predetermined in this experiment to be ten subjects due to time limitations of the scope of this research. Since every subject had to take the experiment in five different scenarios, the total number of samples was fifty. Although the above formula was not used to determine the sample size, it can be used to find the precision of the experiment when sample size is known:

$$\delta = (U_\alpha + U_\beta)\frac{\sigma}{\sqrt{N}}$$

In this experiment, the population can be considered the paired difference in the subject's score and the near-optimal score gained by tabu search. The mean of this population can then be compared (double sided) against zero, which tests whether there is a difference between the scores gained by the subject and those gained by the tabu search. From the experiments, it was estimated that the standard deviation of the population, for all scenarios combined, was $\sigma = 120$. Therefore, if it is desired to have $\alpha = 0.05$ and $\beta = 0.10$ for a sample size $N = 50$, then:

$$\delta = (1.960 + 1.282)\frac{120}{\sqrt{50}} = 55.02 \cong 55$$

A precision of $\delta = 55$, in a score range of 0 to 1000, is very reasonable (5.5%). This precision means that the statistical test cannot distinguish the difference between the subject's score and tabu search's if it is less than 55. It is shown in Section L.2 that there is significant difference between the subject and tabu search, which also means that the absolute value of the sample mean was greater than 55.

The same approach can be applied to data for each scenario considered independently. It should be noted that there are only ten samples in each scenario. From the experiments, it was found out that the largest standard deviation among the scenarios belonged to scenario 5, $\sigma = 157$. Therefore, if it is desired to have $\alpha = 0.05$ and $\beta = 0.10$ for a sample size of $N = 10$, then:

$$\delta = (1.960 + 1.282)\frac{157}{\sqrt{10}} = 160.96 \cong 161$$

A precision of $\delta = 161$, in a score range of 0 to 1000, is fairly reasonable (16.1%) although it is the worst precision among all the scenarios. This precision means that

the statistical test, in scenario 5, cannot distinguish the difference between the subject's score and tabu search's if it is less than 161. It is shown in Section L.3 that there is significant difference between the subject and tabu search in scenario 5, which also means that the absolute value of the sample mean was even greater than 161. Section L.3 shows that there was significant difference between the means in all the other scenarios as well. Thus, knowing the precision of test for any of those scenarios is not much important as the null hypothesis is rejected.

L.2 ALL SCENARIOS COMBINED

The first test conducted, has considered all scenarios together and tested whether there is a significant difference between the mean of the subject's score and tabu-search's score. This test in a pair-wise comparison translates to finding the score difference between each subject and its corresponding tabu score for that scenario, and then testing whether this difference is zero or not.

H_0: The mean difference between a subject's score and its corresponding tabu search's score, for all scenarios combined, is zero or

$$\mu_{\text{difference}} = 0$$

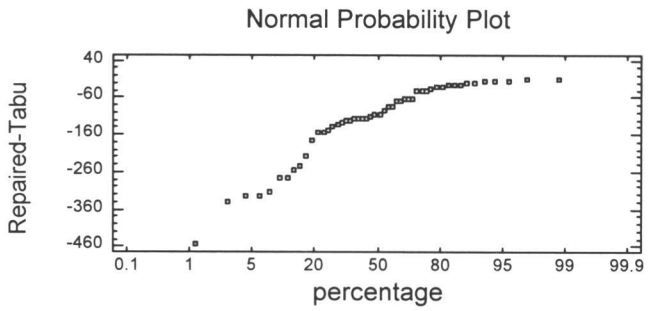
H_1: The mean of the score difference between subject and tabu is not zero or

$$\mu_{\text{difference}} \neq 0$$

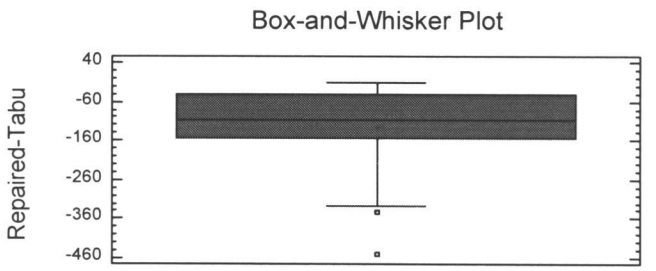
The mean and the median for this sample is:
Sample mean = - 121.83
Sample median = - 106.245

First, a pair-wised t test is conducted, and the result is:
Computed t statistic = - 8.1912
P-value = 9.6279E-11

Since the P-value for the test is much less than 0.05, the null hypothesis can be strongly rejected at the 95% confidence level. The t-test is based on normality assumption of the residuals. Figure L.1 shows the normal probably plot and box plot of the residuals.



(a)



(b)

Figure L.1 (a) Normal probability plot and (b) Box plot of the residuals for all scenarios combined

Since the normality assumption is not very strong and the data is skewed, two non-parametric tests are also performed, which are less sensitive to presence of outliers and assumption of normality. These tests are: Sign and Signed Rank test.

Sign test is based on counting the values above and below the hypothesized median. The hypotheses for this test are:

H_0: The difference between a subject's score and its corresponding tabu search's score, for all scenarios combined, has a zero median or

$$\text{Median}_{\text{difference}} = 0$$

H_1: The median of the score difference between subject and tabu is not zero or

$$\text{Median}_{\text{difference}} \neq 0$$

For a sign test the results are:
Number of values below hypothesized median: 50
Number of values above hypothesized median: 0
Large sample test statistic = 6.92965 (continuity correction applied)
P-value = 4.24549E-12

Since the P-value for this test is less than 0.05, the null hypothesis can be rejected at 95% confidence interval.

Signed Ranked test, on the other hand, is based on comparing the average ranks of values above and below the hypothesized median. The hypotheses for this test are:

H_0: The difference between a subject's score and its corresponding tabu search's score, for all scenarios combined, has a zero median or

$$\text{Median}_{\text{difference}} = 0$$

H_1: The median of the score difference between subject and tabu is not zero or

$$\text{Median}_{\text{difference}} \neq 0$$

For a signed-rank test, the results are:
Average rank of values below hypothesized median: 25.5
Average rank of values above hypothesized median: 0.0
Large sample test statistic = 6.14914 (continuity correction applied)
P-value = 7.82133E-10

Since the P-value for this test is also less than 0.05, the null hypothesis can be rejected at 95% confidence interval.

Therefore all three tests unanimously reject the hypothesis that subject's score is the same as tabu's score. Since the mean (- 121.83) and median (- 106.245) of the difference shows that subject's score is significantly inferior, and since none of subject's could get a score better than tabu in any scenario, it can be concluded that the score gained by tabu is superior to the score gained by the subject.

L.3 EACH SCENARIO CONSIDERED INDEPENDENTLY

In the previous section it was concluded that in all scenarios combined, subjects' performances were inferior to the tabu search solutions. However, chances are to observe a different result if each scenario is looked at independently. Imagine a scenario in which subjects had exceedingly poor performances. This result might have a high contribution in concluding that in all scenarios combined, subjects' performances were inferior although they might have performed well in some other scenario(s). Therefore, the next set of statistical tests is applied to each scenario independently. The tests and hypotheses are the same as the ones performed in all

scenarios combined. These tests are: t-test, Sign test, and Signed Rank test. The latter two are less sensitive to outliers and assumption of normality for residuals.

Table L.1 shows a summary of the results for each scenario independently. In all cases, the null hypothesis was rejected at 95% confidence because P-value was less than 0.05. Therefore, in none of the scenarios it could be proven that subjects performed as good as tabu search. Moreover, the mean and medians in all scenarios were negative indicating that the performance of the subject was inferior to that of tabu search's. The raw data is also included in Table L.2.

Table L.1 Summary of results for each scenario independently

Scen.	Mean Diffr.	Median Diffr.	t-Test		Sign Test		Signed Rank Test		Null Hypo.
			Ststic.	P-Val.	Ststic.	P-Val.	Ststic.	P-Val.	
1	-60.5	-40.3	-3.76	0.0045	2.846	0.0044	2.752	0.0059	Reject
2	-134.8	-103.9	-4.20	0.0023	2.846	0.0044	2.752	0.0059	Reject
3	-133.6	-117.4	-4.56	0.0013	2.846	0.0044	2.752	0.0059	Reject
4	-76.6	-33.6	-2.67	0.0256	2.846	0.0044	2.752	0.0059	Reject
5	-203.7	-117.9	-5.05	0.0007	2.846	0.0044	2.752	0.0059	Reject

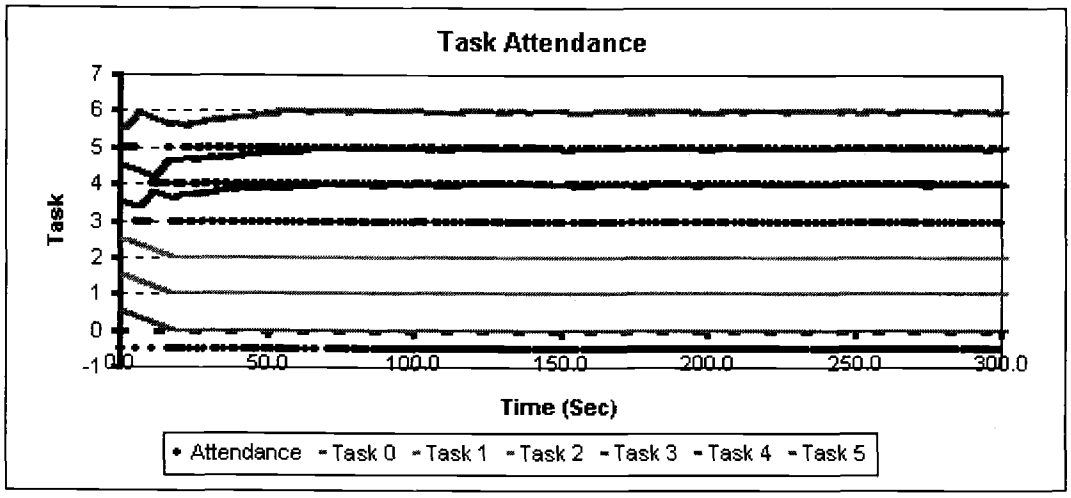
Table L.2 Raw data for all scenarios

Subject	Scenario	Subject's Score		Score	Delta
		Original	Repaired	Tabu	Repaired - Tabu
1	5	**570.01**	577.08	685.06	-107.98
2	5	404.35	440.16	685.06	-244.90
3	5	562.84	568.05	685.06	-117.01
4	5	554.52	**580.55**	685.06	-104.51
5	5	557.54	566.35	685.06	-118.71
6	5	562.95	572.61	685.06	-112.45
7	5	**156.36**	234.65	685.06	-450.41
8	5	564.06	570.53	685.06	-114.53
9	5	253.85	360.70	685.06	-324.36
10	5	315.01	343.26	685.06	-341.80
1	4	**559.85**	566.28	574.66	-8.38
2	4	360.55	427.20	574.66	-147.46
3	4	554.91	559.59	574.66	-15.06
4	4	508.22	534.53	574.66	-40.13
5	4	355.84	401.70	574.66	-172.96
6	4	516.71	534.49	574.66	-40.16
7	4	**269.94**	299.10	574.66	-275.55
8	4	536.92	552.47	574.66	-22.18
9	4	552.33	558.06	574.66	-16.60
10	4	503.40	547.66	574.66	-27.00
1	3	308.61	328.67	456.37	-127.71
2	3	328.41	341.72	456.37	-114.66
3	3	**116.11**	140.51	456.37	-315.87
4	3	204.14	315.98	456.37	-140.39
5	3	163.42	180.47	456.37	-275.90
6	3	**404.80**	430.24	456.37	-26.13
7	3	361.79	388.33	456.37	-68.04
8	3	286.42	336.28	456.37	-120.09
9	3	382.27	391.90	456.37	-64.47
10	3	358.94	373.17	456.37	-83.20

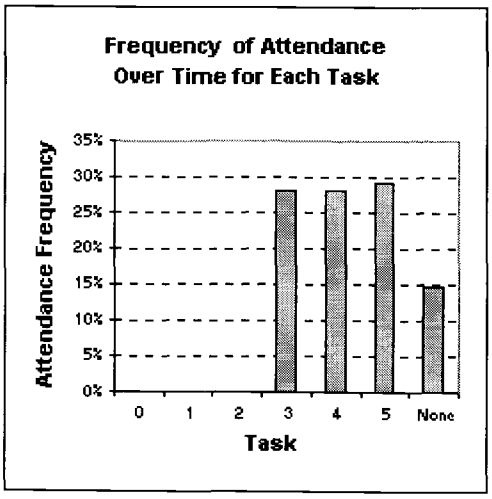
Table L.2 (Continued) Raw data for all scenarios

Subject	Scenario	Subject's Score		Score	Delta
		Original	Repaired	Tabu	Repaired – Tabu
1	2	499.72	520.08	581.70	-61.62
2	2	509.33	516.83	581.70	-64.86
3	2	473.02	496.96	581.70	-84.74
4	2	388.10	458.56	581.70	-123.14
5	2	400.33	427.90	581.70	-153.80
6	2	348.17	363.16	581.70	-218.54
7	2	509.73	552.48	581.70	-29.22
8	2	260.76	327.57	581.70	-254.13
9	2	**538.45**	549.50	581.70	-32.20
10	2	**232.57**	256.15	581.70	-325.55
1	1	260.68	267.61	398.45	-130.84
2	1	230.58	301.58	398.45	-96.87
3	1	359.72	374.60	398.45	-23.85
4	1	302.00	330.31	398.45	-68.14
5	1	**190.70**	243.28	398.45	-155.17
6	1	369.09	378.48	398.45	-19.97
7	1	351.36	**385.76**	398.45	-12.68
8	1	348.11	356.69	398.45	-41.76
9	1	**371.93**	381.50	398.45	-16.95
10	1	348.42	359.59	398.45	-38.86

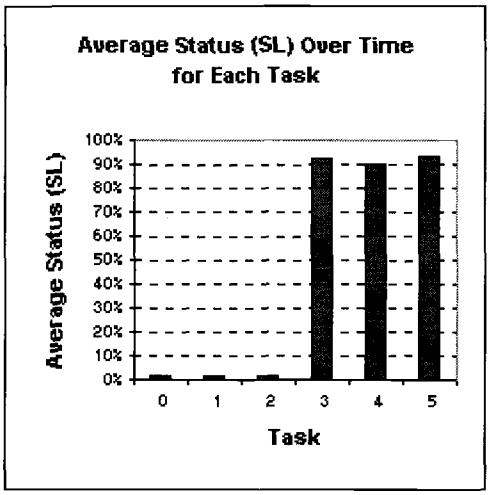
APPENDIX M: ANALYSIS OF THE TASK ATTENDANCE



(a) Task Attendance

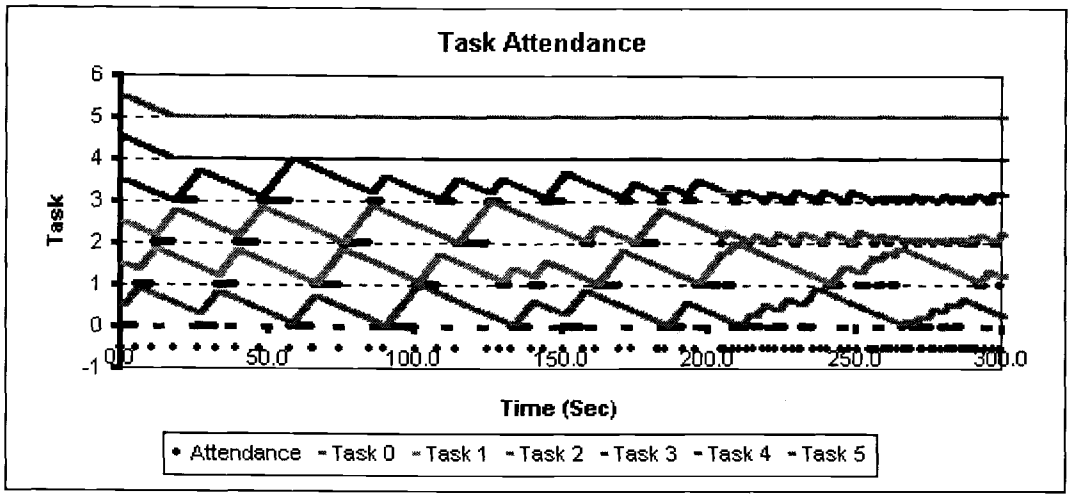


(b) Frequency of attendance over time

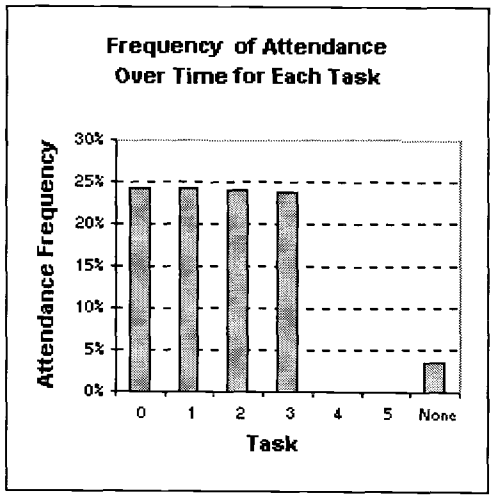


(c) Average status (SL) over time

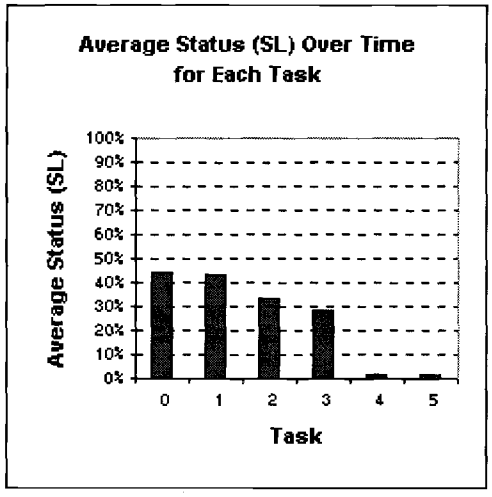
Figure M.1 Best human performance in scenario 1 (identical CR, DR, and weight, across tasks); score: 372



(a) Task Attendance

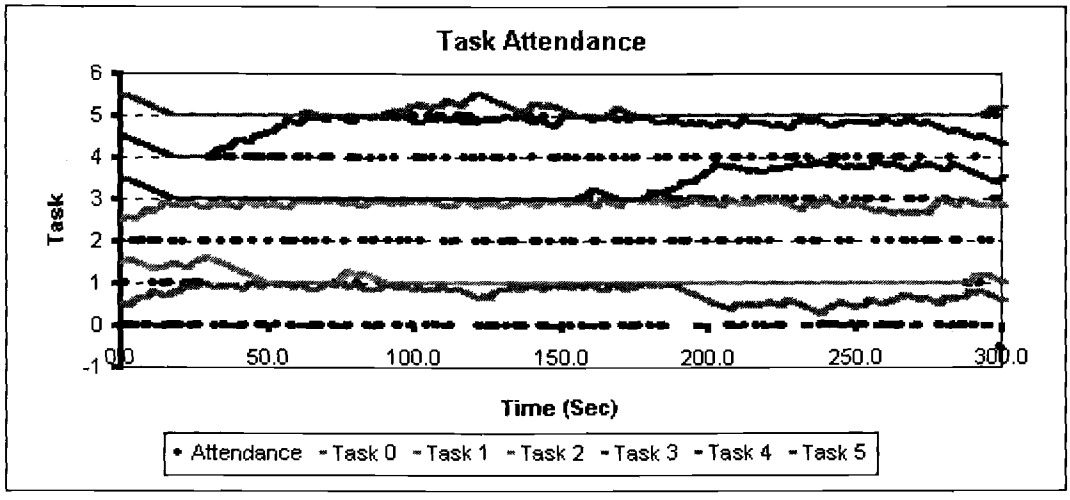


(b) Frequency of attendance over time

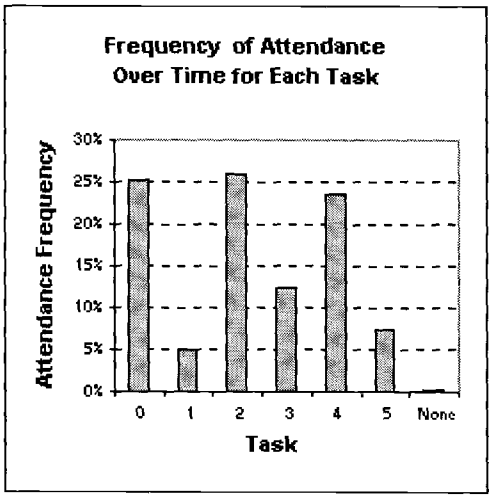


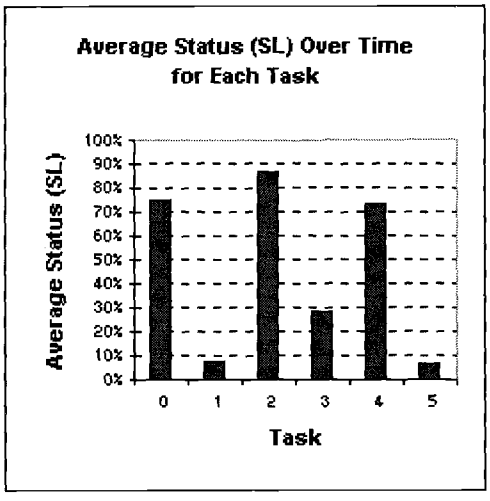
(c) Average status (SL) over time

Figure M.2 Worst human performance in scenario 1 (identical CR, DR, and weight, across tasks); score: 191



(a) Task Attendance

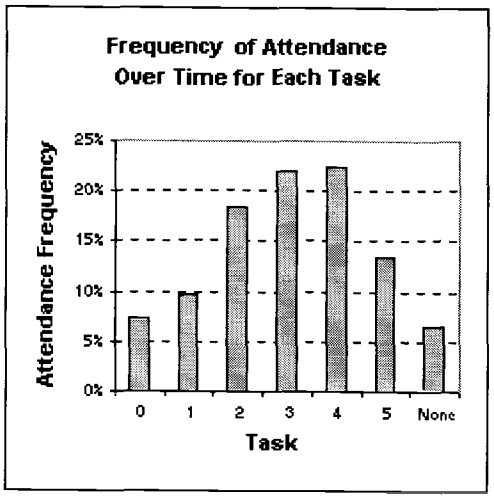


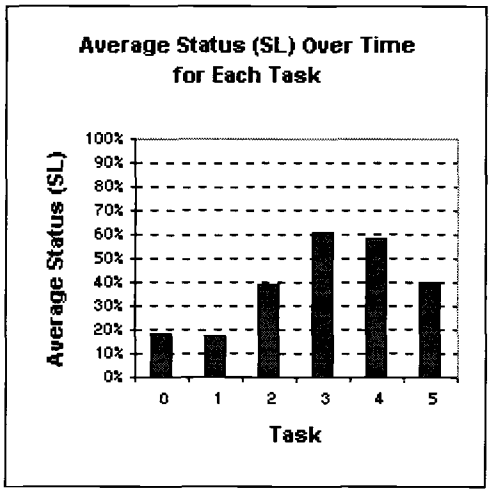


(b) Frequency of attendance over time

(c) Average status (SL) over time

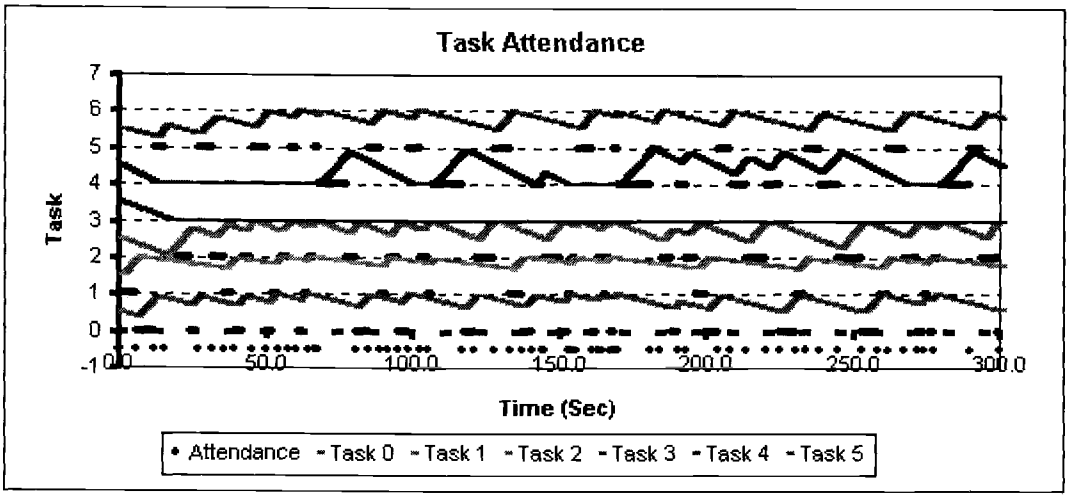
Figure M.3 Tabu performance in scenario 1 (identical CR, DR, and weight, across tasks); score: 398



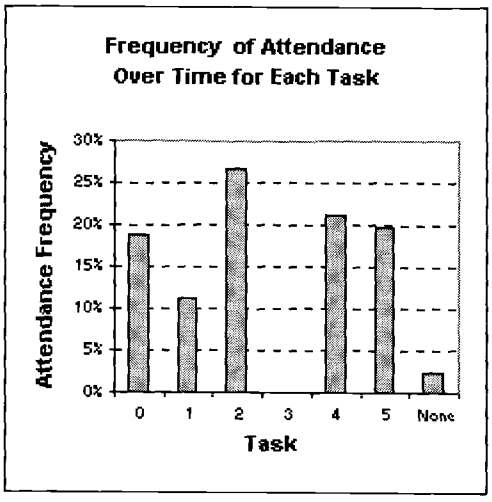


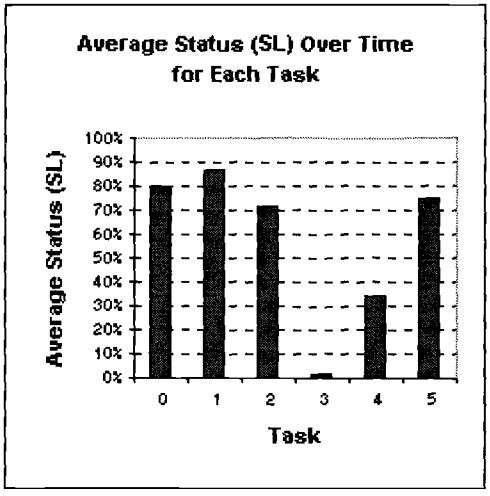
(b) Frequency of attendance over time (c) Average status (SL) over time

Figure M.4 Mean of the human performance in scenario 1 (identical CR, DR, and weight, across tasks); score: 313



(a) Task Attendance

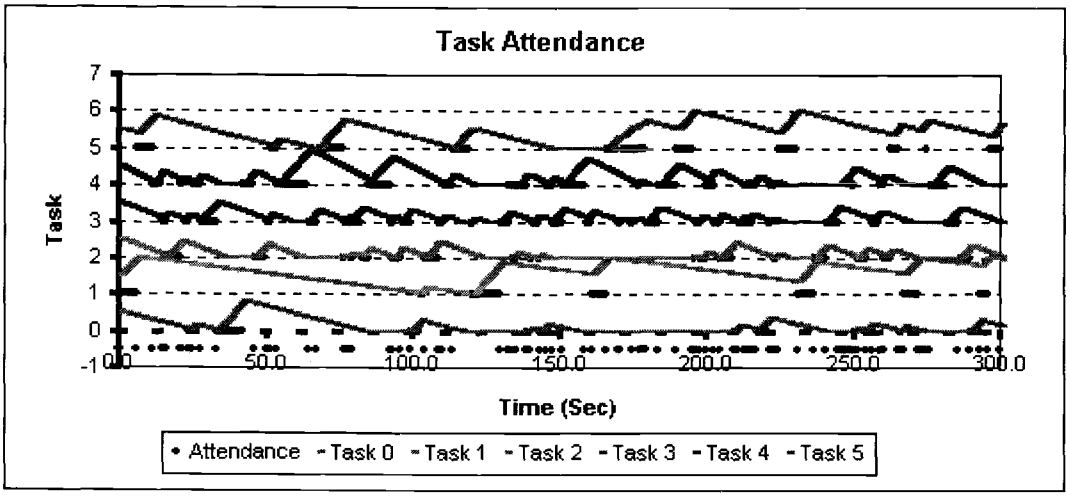




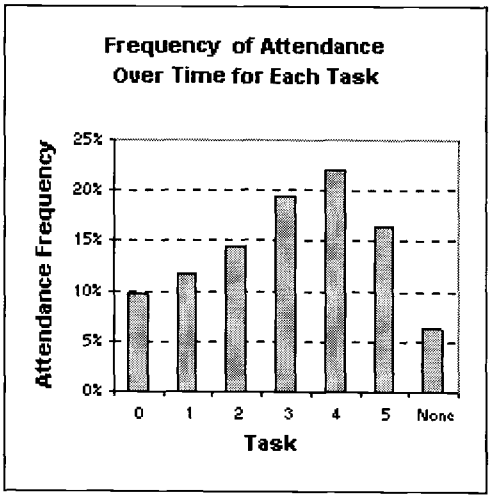
(b) Frequency of attendance over time

(c) Average status (SL) over time

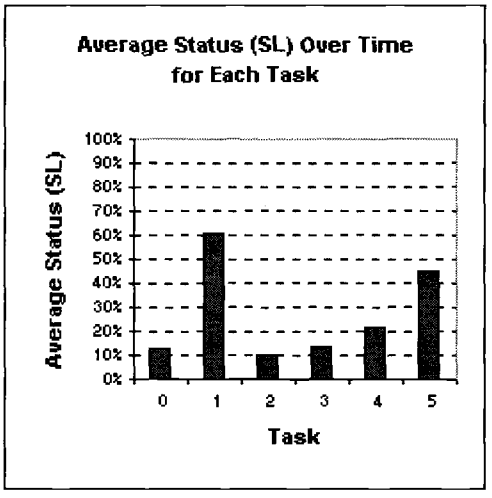
Figure M.5 Best human performance in scenario 2 (different DR across tasks); score: 538



(a) Task Attendance

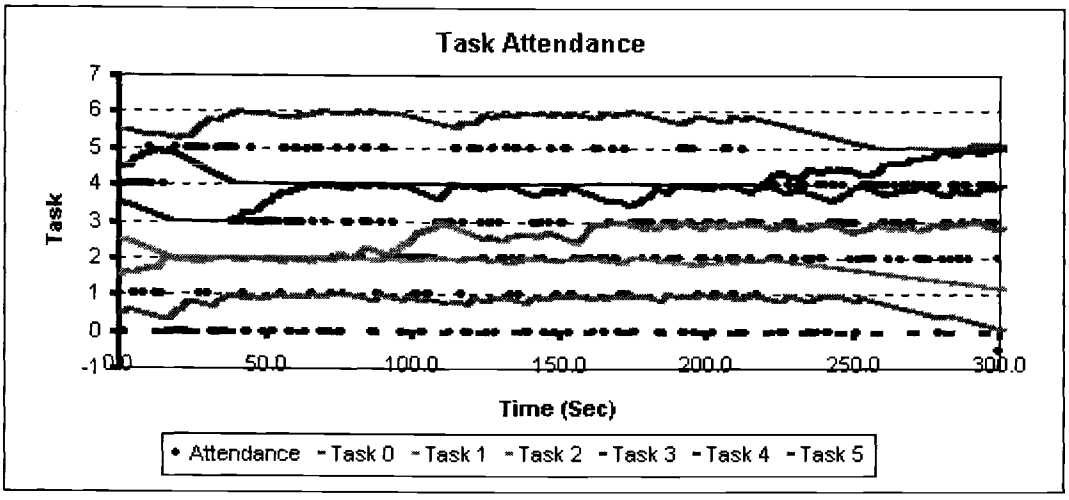


(b) Frequency of attendance over time

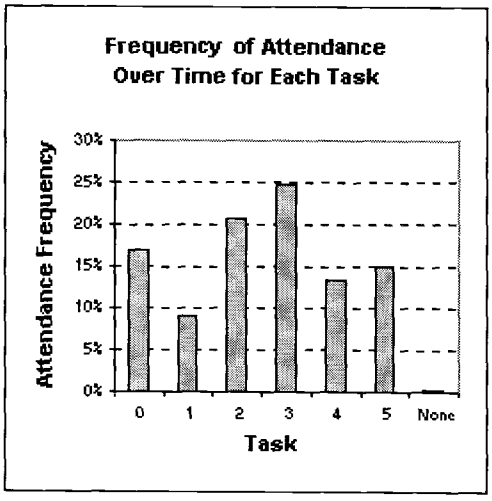


(c) Average status (SL) over time

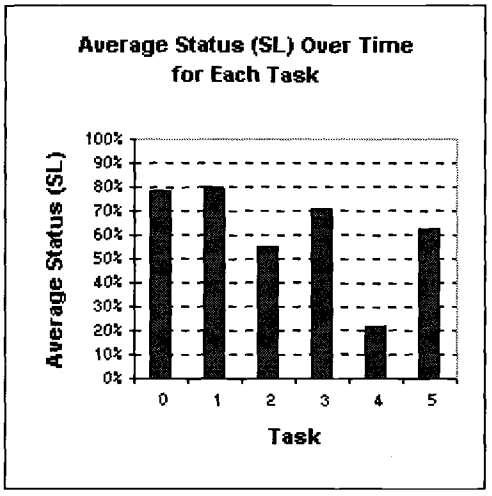
Figure M.6 Worst human performance in scenario 2 (different DR across tasks); score: 233



(a) Task Attendance

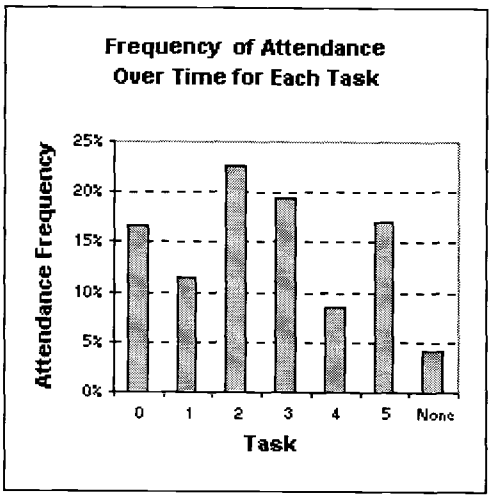


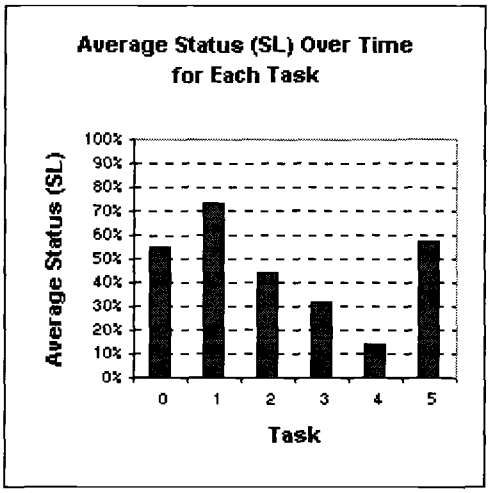
(b) Frequency of attendance over time



(c) Average status (SL) over time

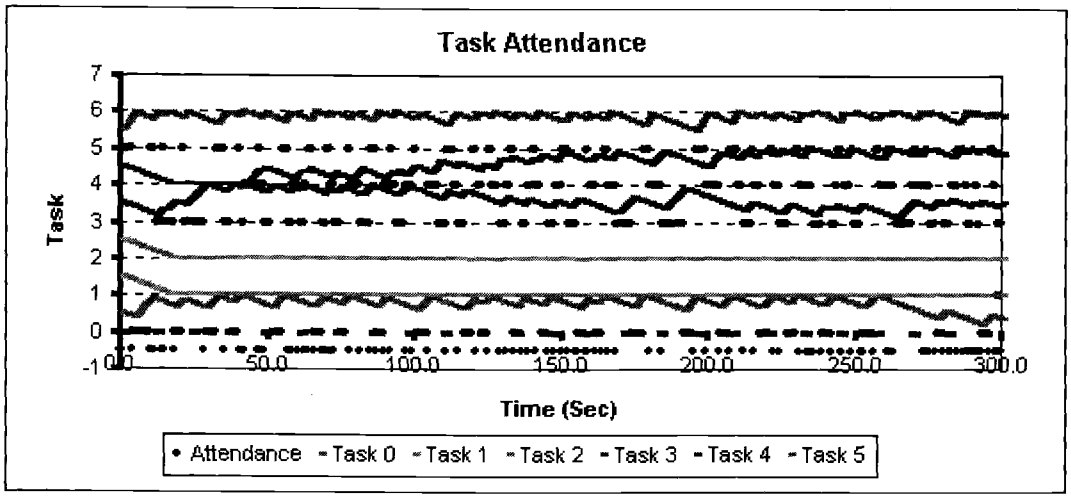
Figure M.7 Tabu performance in scenario 2 (different DR across tasks); score: 582



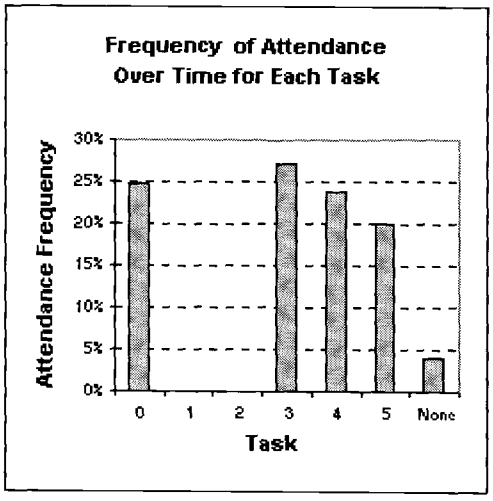


(b) Frequency of attendance over time (c) Average status (SL) over time

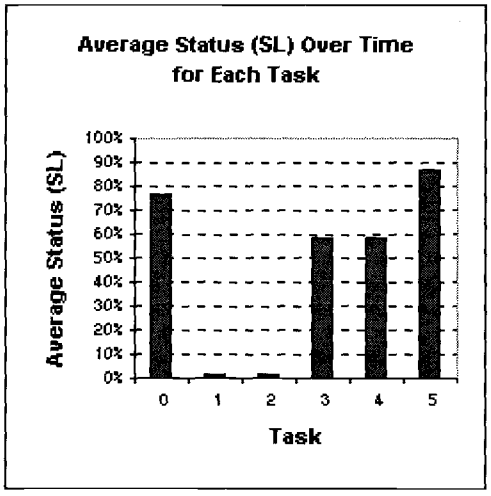
Figure M.8 Mean of the human performance in scenario 2 (different DR across tasks); score: 416



(a) Task Attendance

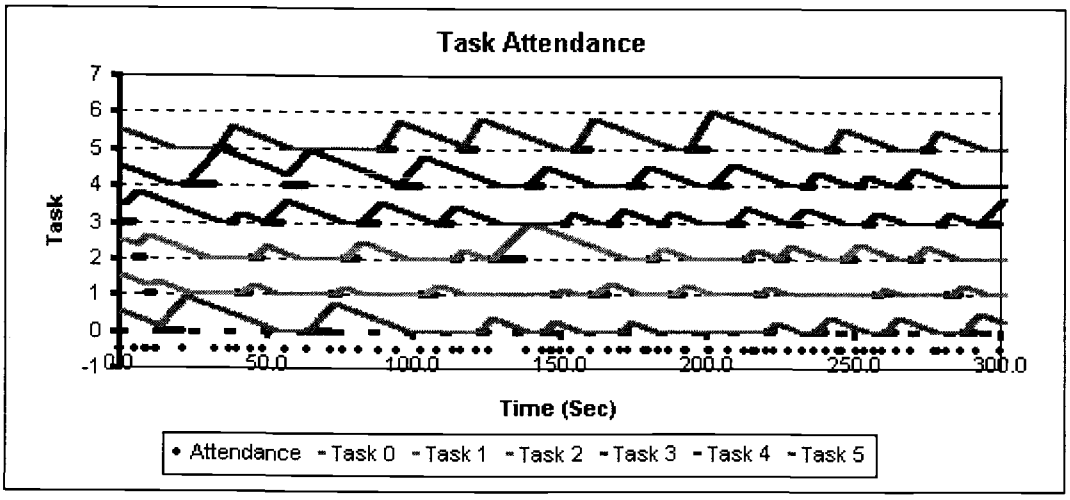


(b) Frequency of attendance over time

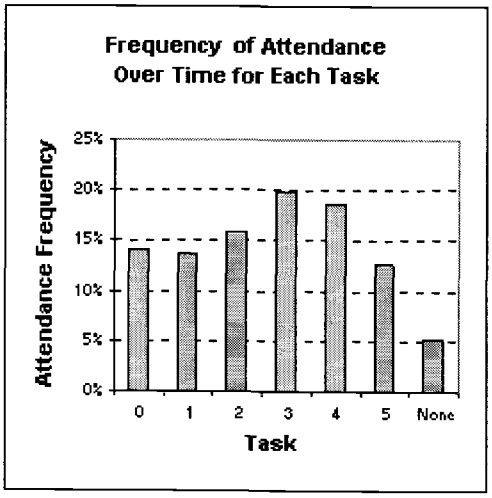


(c) Average status (SL) over time

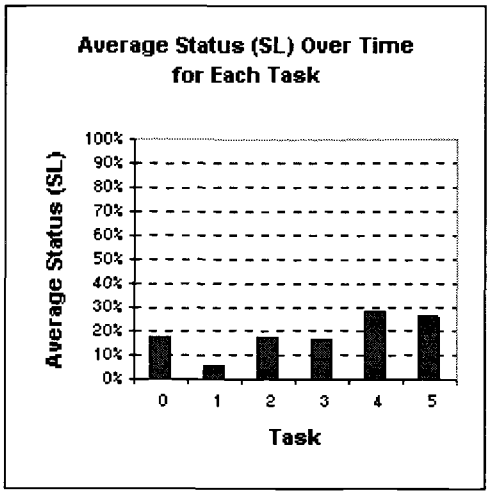
Figure M.9 Best human performance in scenario 3 (different CR across tasks); score: 405



(a) Task Attendance

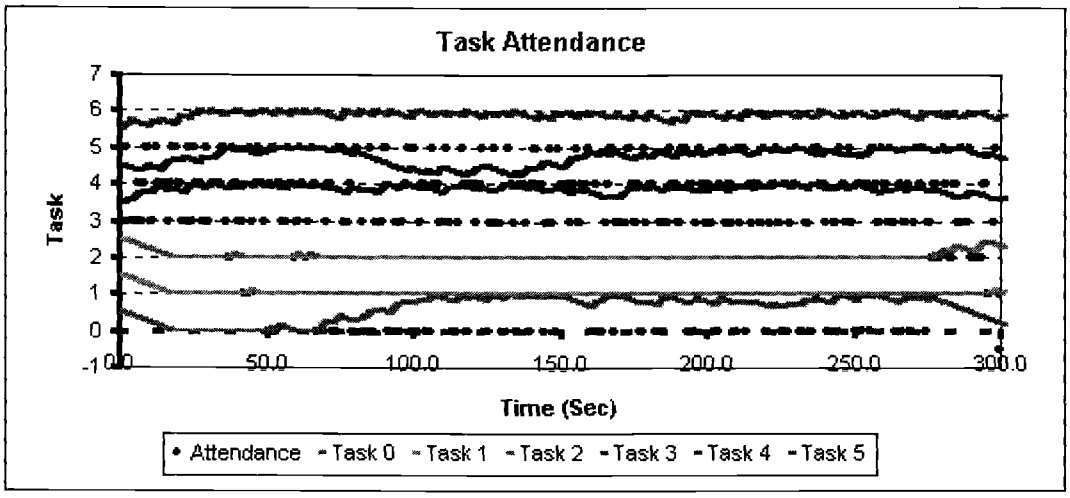


(b) Frequency of attendance over time

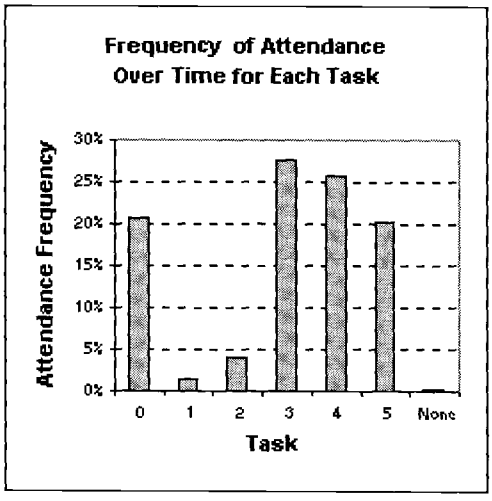


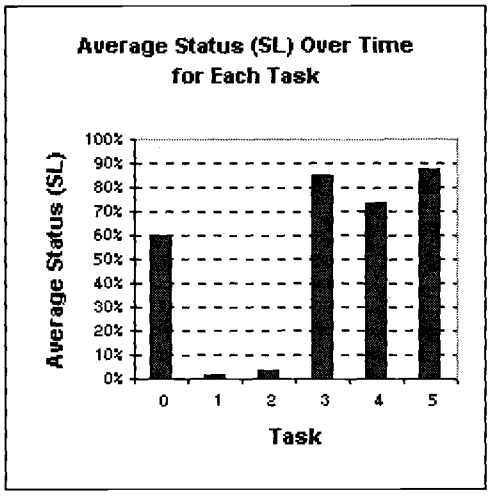
(c) Average status (SL) over time

Figure M.10 Worst human performance in scenario 3 (different CR across tasks); score: 116



(a) Task Attendance

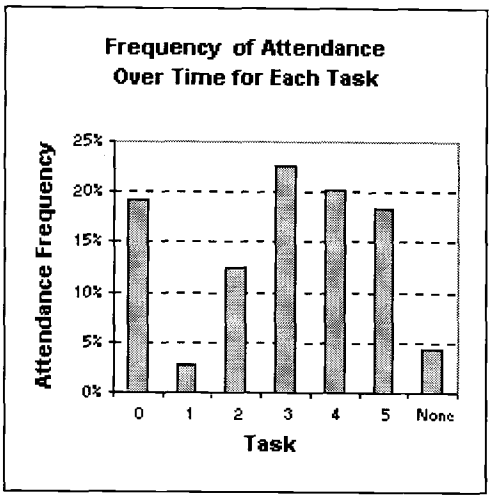


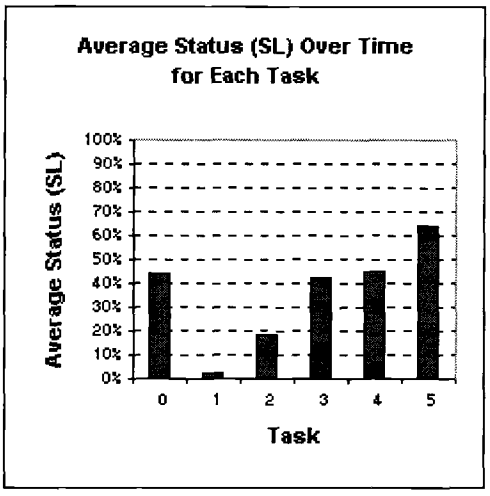


(b) Frequency of attendance over time

(c) Average status (SL) over time

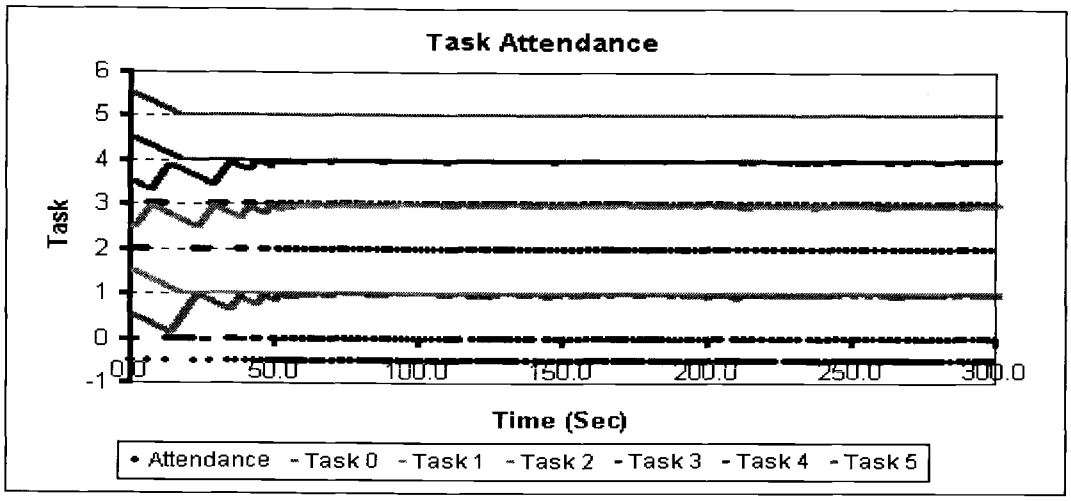
Figure M.11 Tabu performance in scenario 3 (different CR across tasks); score: 456



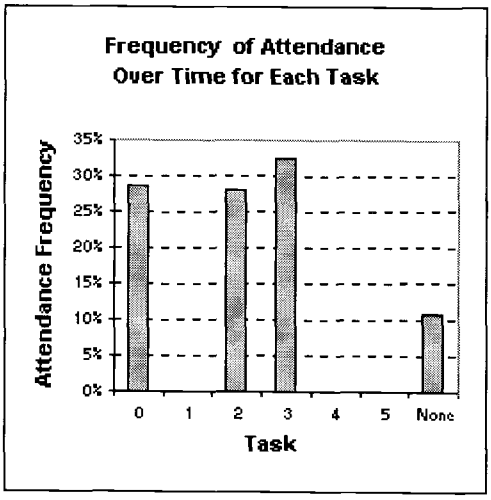


(b) Frequency of attendance over time (c) Average status (SL) over time

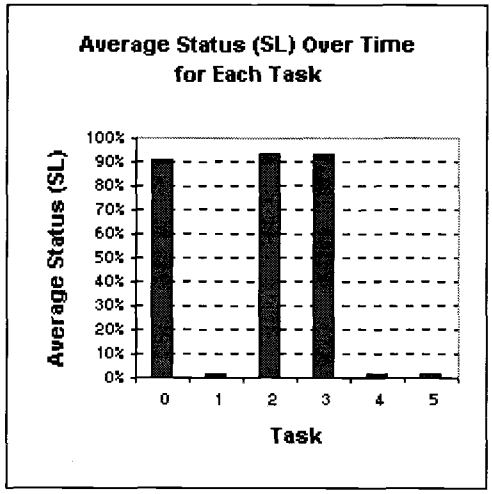
Figure M.12 Mean of the human performance in scenario 3 (different CR across tasks); score: 291



(a) Task Attendance

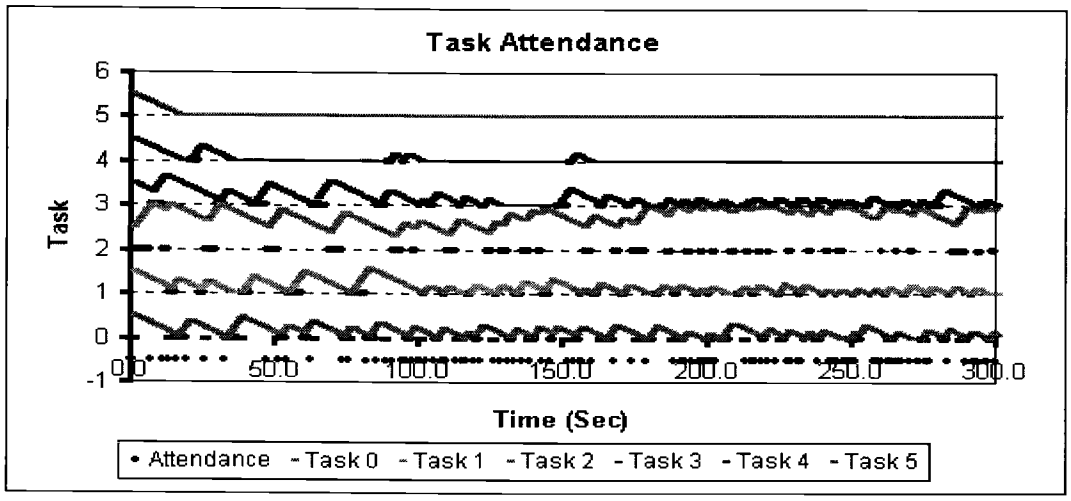


(b) Frequency of attendance over time

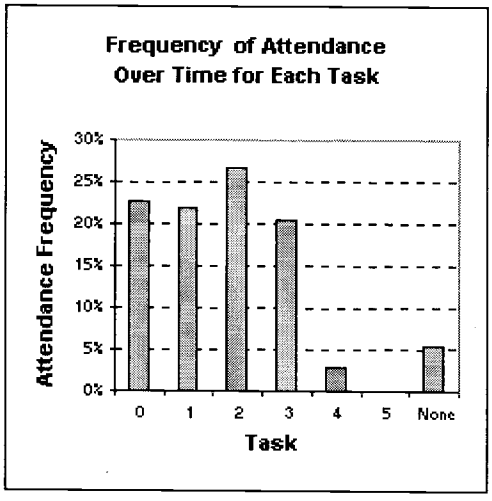


(c) Average status (SL) over time

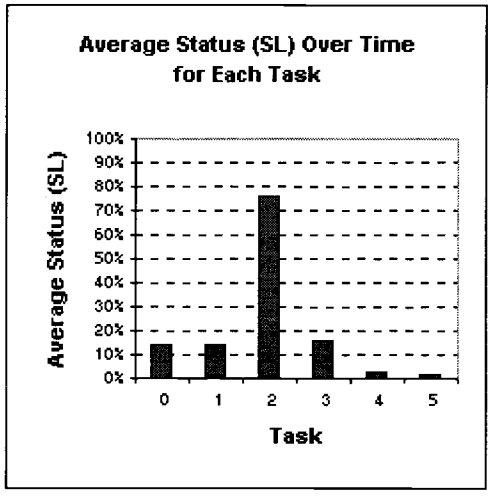
Figure M.13 Best human performance in scenario 4 (different weight across tasks); score: 560



(a) Task Attendance

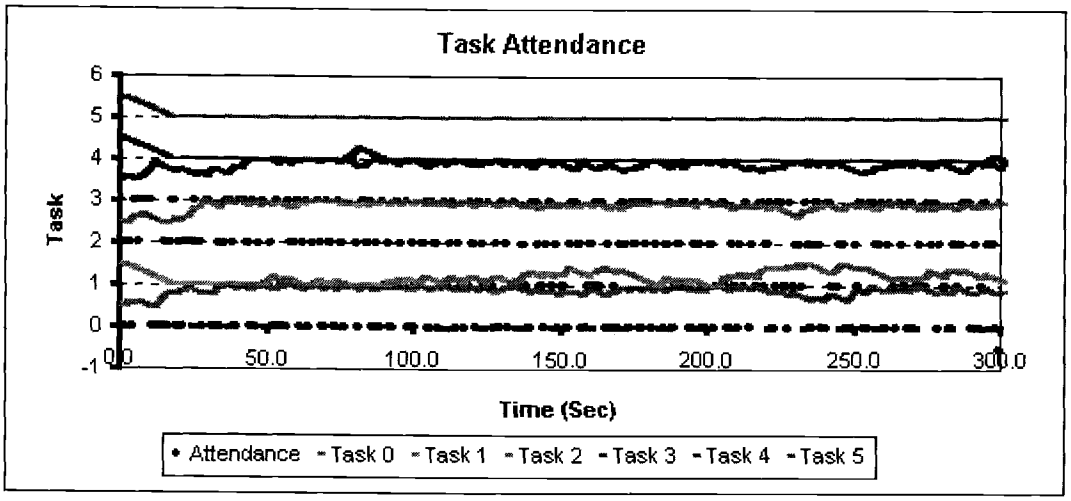


(b) Frequency of attendance over time

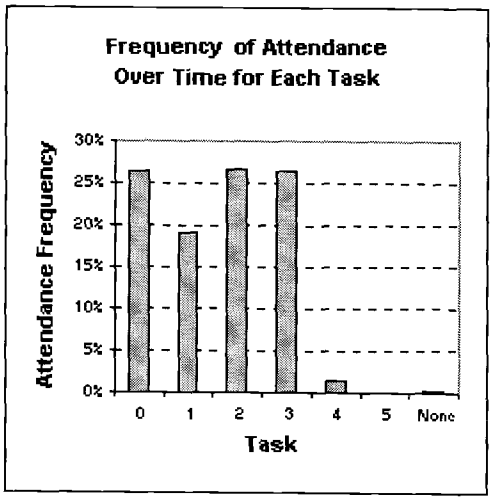


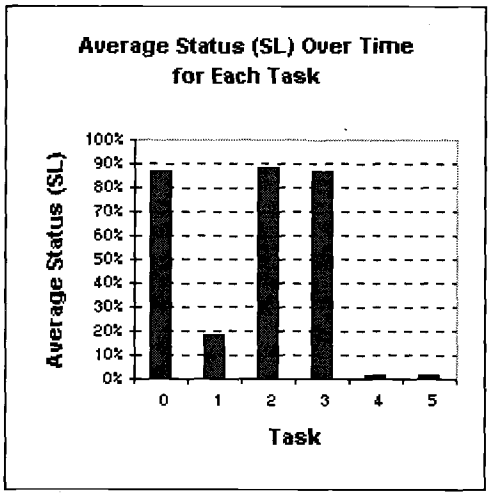
(c) Average status (SL) over time

Figure M.14 Worst human performance in scenario 4 (different weight across tasks); score: 270



(a) Task Attendance

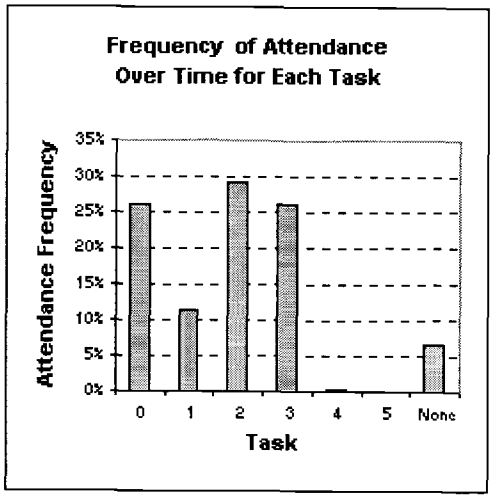


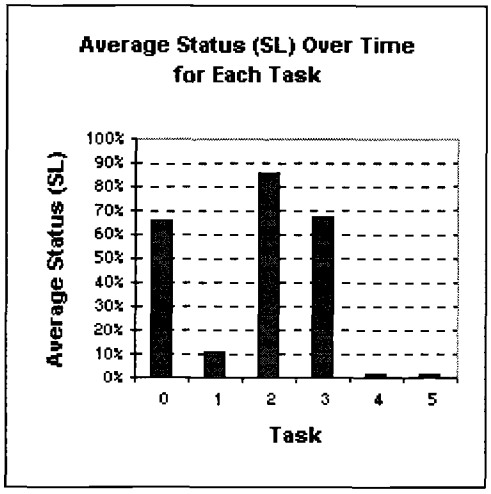


(b) Frequency of attendance over time

(c) Average status (SL) over time

Figure M.15 Tabu performance in scenario 4 (different weight across tasks); score: 575

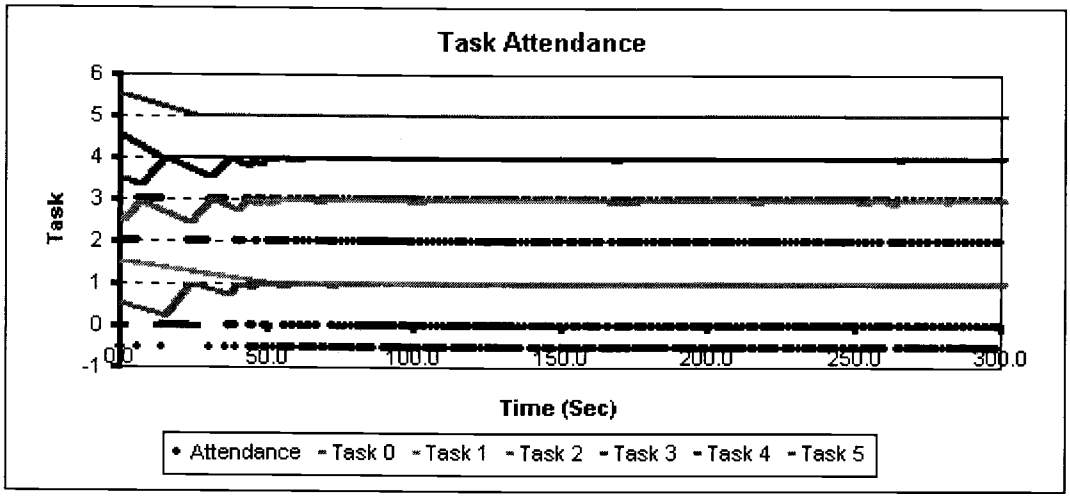




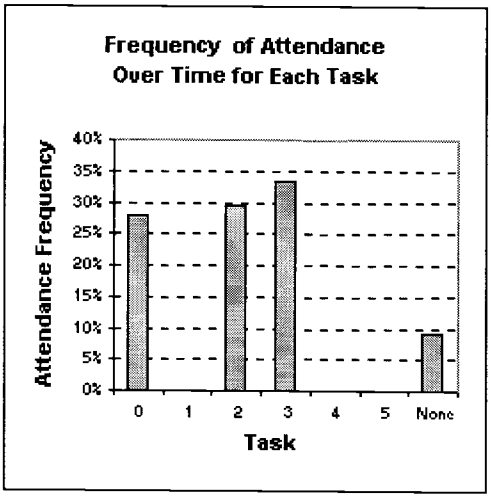
(b) Frequency of attendance over time

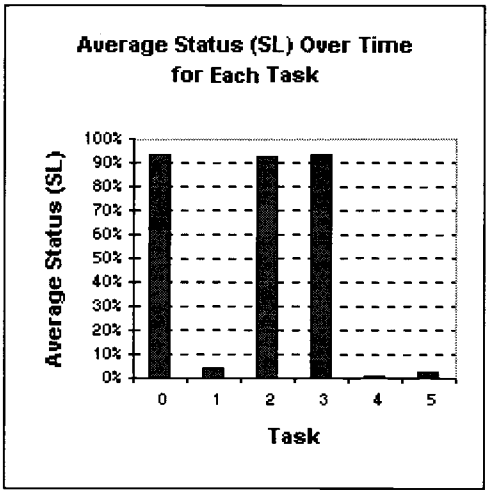
(c) Average status (SL) over time

Figure M.16 Mean of the human performance in scenario 4 (different weight across tasks); score: 472



(a) Task Attendance

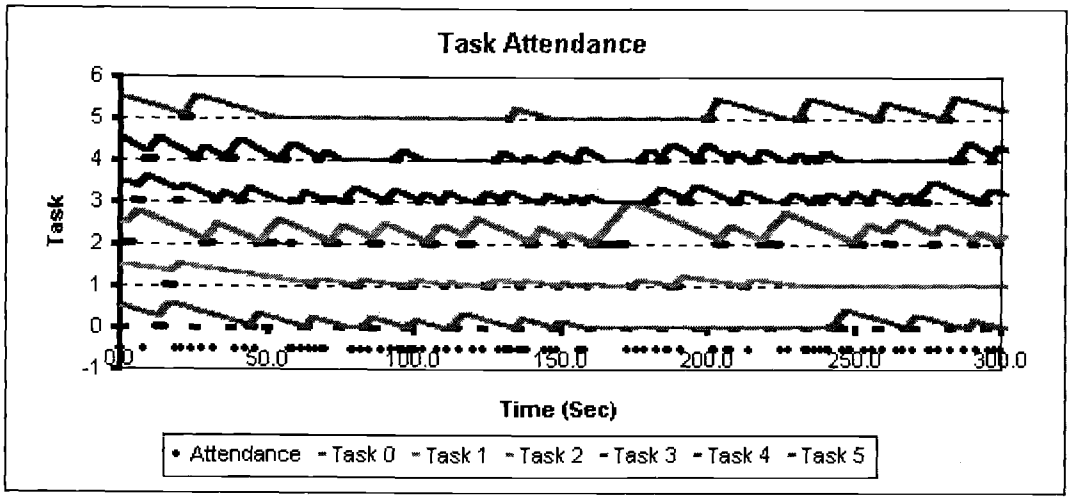




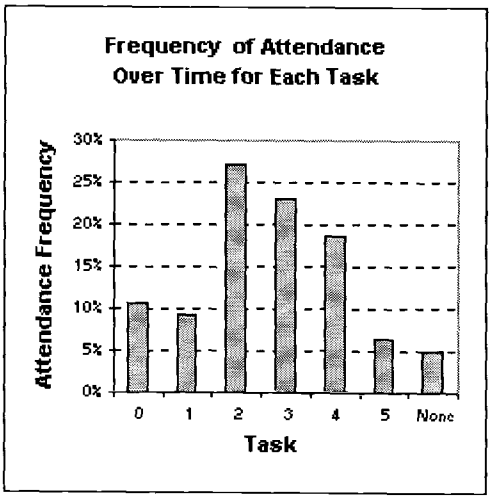
(b) Frequency of attendance over time

(c) Average status (SL) over time

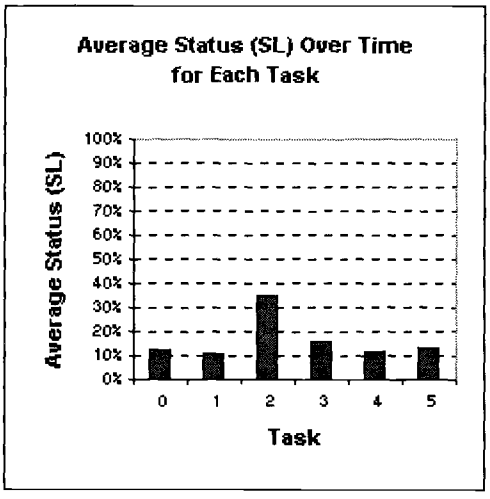
Figure M.17 Best human performance in scenario 5 (different CR, DR, and weight, across tasks); score: 570



(a) Task Attendance

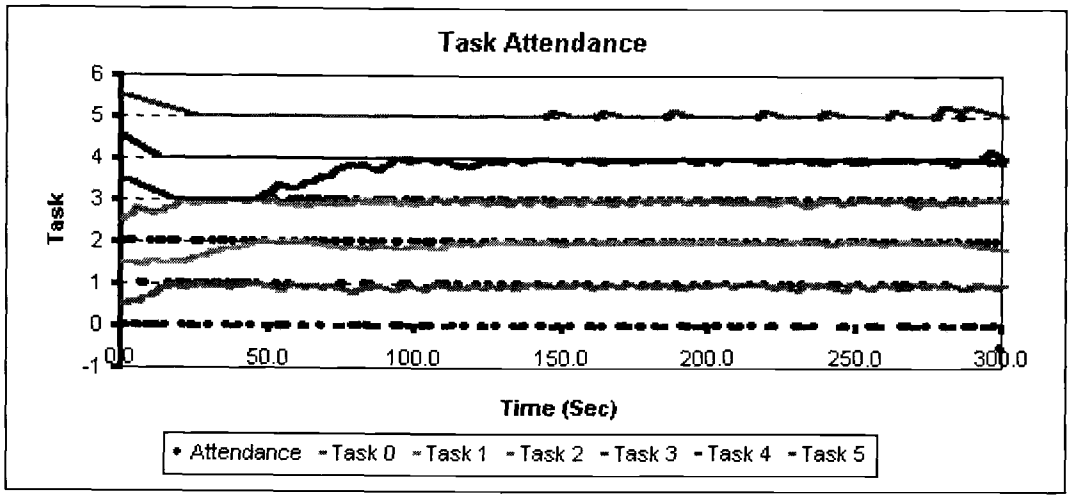


(b) Frequency of attendance over time

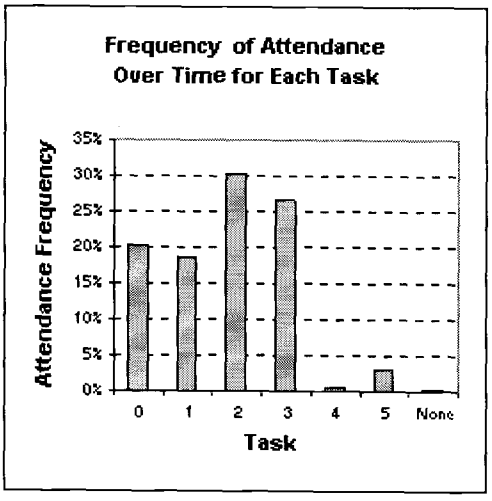


(c) Average status (SL) over time

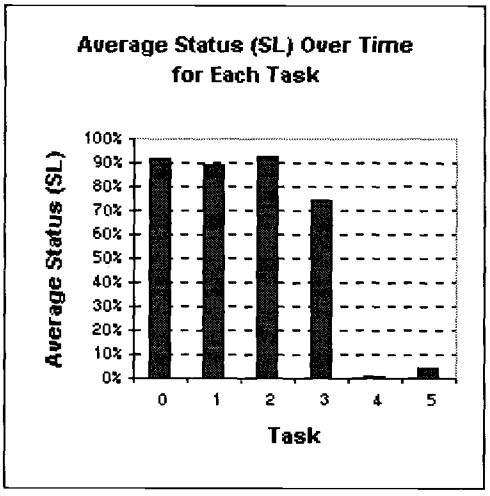
Figure M.18 Worst human performance in scenario 5 (different CR, DR, and weight, across tasks); score: 156



(a) Task Attendance

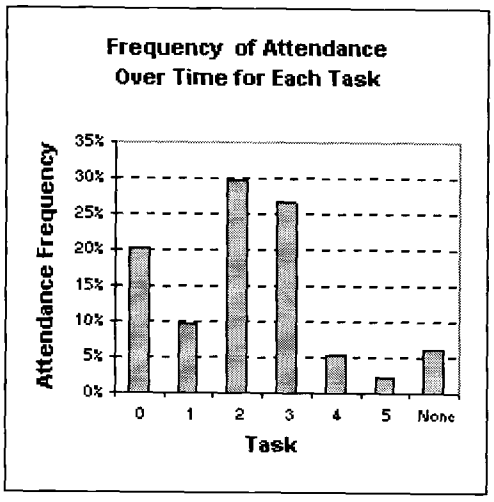


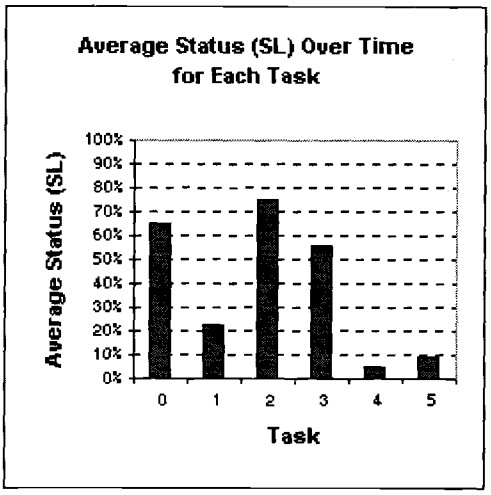
(b) Frequency of attendance over time



(c) Average status (SL) over time

Figure M.19 Tabu performance in scenario 5 (different CR, DR, and weight, across tasks); score: 685





(b) Frequency of attendance over time

(c) Average status (SL) over time

Figure M.20 Mean of the human performance in scenario 5 (different CR, DR, and weight, across tasks); score: 450

APPENDIX N: SIMPLE REGRESSION ANALYSIS FOR THE LEARNING EFFECT ACROSS THE SUBJECTS

The following graph, generated by STATGRAPHICS PLUS Version 5.0, shows the plot of the fitted regression model to the data underlying Figure 8.10.

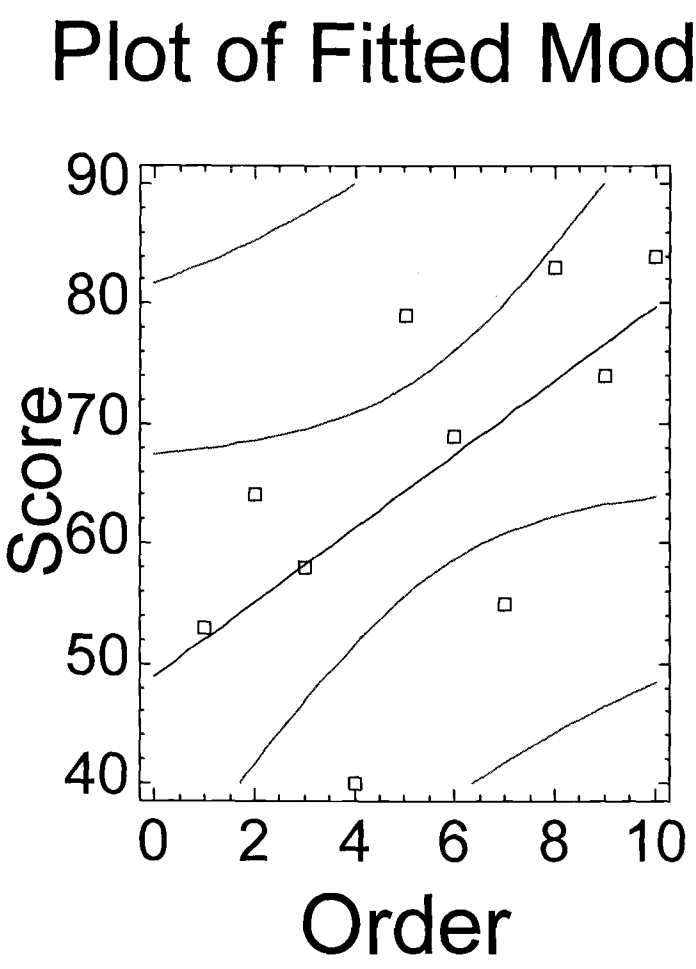


Figure N.1 Plot of the fitted regression model to the average relative score gained by the subjects across scenarios, and the order of subjects

The following is the regression analysis for the fitted regression model generated by STATGRAPHICS PLUS Version 5.0.

Table N.1 Regression analysis – linear model: $Y = a + b \times X$

Dependent variable: Score

Independent variable: Order

Parameter	Estimate	Standard Error	T-Statistic	P-Value
Intercept	48.9333	8.02426	6.09818	0.0003
Slope	3.08485	1.29323	2.38539	0.0442

Analysis of Variance

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Model	785.094	1	785.094	5.69	0.0442
Residual	1103.81	8	137.976		
Total	1888.9	9			

Correlation Coefficient = 0.644698

R-squared = 41.5636

R-squared (adjusted for d.f.) = 34.259 percent

Standard Error of Est. = 11.7463

Mean absolute error = 7.93697

Durbin-Watson statistic = 2.86045 (P=0.0203)

Lag 1 residual autocorrelation = -0.438723

The output shows the results of fitting a linear model to describe the relationship between Score and Order. The equation of the fitted model is

Score = 48.9333 + 3.08485*Order

Since the P-value in the ANOVA table is less than 0.05, there is a statistically significant relationship between Score and Order at the 95% confidence level.

The R-Squared statistic indicates that the model as fitted explains 41.5636% of the variability in Score. The correlation coefficient equals 0.644698, indicating a moderately strong relationship between the variables. The standard error of the estimate shows the standard deviation of the residuals to be 11.7463. This value can be used to construct prediction limits for new observations.