SCHEDULING PROJECT MEETINGS

BY

DARRYL W. DORMUTH

A Thesis
Submitted to the Faculty of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF SCIENCE

Department of Actuarial Mathematics
and Management Sciences
University of Manitoba
Winnipeg, Manitoba, Canada

ISBN 0-315-76846-0

SCHEDULING PROJECT MEETINGS

BY

DARRYL W. DORMUTH

A thesis submitted to the Faculty of Graduate Studies of
the University of Manitoba in partial fulfillment of the requirements
of the degree of

MASTER OF SCIENCE

Table of Contents

Table of Contents i
Abstract iii
Acknowledgements iv
Nomenclature v
List of Tables vii
List of Figures vii

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: BACKGROUND 9

CHAPTER 3: METHODS USED FOR SYNCHRONIZED DATA TRANSFER 24

3.1 Introduction 24
3.2 Detailed Description of the Integrated Simulation Package 25
3.3 General Assumptions Used In the Models 26
3.3.1 Objective of A Strategy 27
3.3.2 Definition of Performance 27
3.3.3 Solving By Iterated Techniques 28
3.3.4 Complexity of A Strategy 30
3.4 Heuristic Strategies 30
3.4.1 MINSTEP Strategy 31
3.4.2 MAXSTEP Strategy 32
3.4.3 MEANSTEP Strategy 34
3.5 Non-linear Programming (NLP) Strategy 37
3.5.1 Objective Function 38
3.5.2 Deriving DC_i 39
3.5.3 Deriving LQ_T 41
3.5.4 The Non-Linear Program 53
3.5.5 Solving The Non-Linear Program 54
3.6 Summary 56

CHAPTER 4: COMPARISON OF STRATEGIES FOR SCHEDULING SYNCHRONIZED DATA TRANSFERS 57

4.1 Method of Comparison 57
4.2 First Comparison: Stable Conditions 60
4.3 Sensitivity Analysis On The K_i's 62
4.4 Second Comparison: Unstable Conditions 63
4.5 Summary of Results 65

CHAPTER 5: CONCLUSIONS AND SUGGESTED FUTURE RESEARCH . 70

5.1 Conclusions 70
5.2 Suggested Future Research 74

APPENDIX A: CALCULATION OF THE COEFFICIENTS FOR THE DIRECT COST FUNCTIONS 76

REFERENCES 79

Abstract

Many experts in project management embrace the idea of maintaining good communication among project team members. They use phrases like "on a timely basis" and "frequent reporting" when describing the importance of communication to the overall project performance, yet there is little literature that gives any quantitative explanations to these qualitative expressions. This study introduces some quantitative concepts to optimizing the scheduling of project meetings by using a system analogous to a project team executing a project. This system has the ability to repeat projects under identical conditions and as often as needed to obtain good statistics. Therefore, strategies for optimizing the scheduling of meetings can be tested and compared on this system. After evaluating these strategies, one can use the obtained results to better understand the problem of optimizing the scheduling of project meetings.

Key Words: Scheduling, meetings, distributive problem solving

Acknowledgements

The author wishes to express his
deep gratitude to

Bruce, Edwin, and my Dad for all
their support and input into
this study.

My Mom, for putting up with my
"neat and tidy" work area and for
letting me cook once in a while.

Marni, for her support, encouragement,
and her ability to make me feel good even
when I try hard not to.

Nomenclature

The following notations are used in this thesis:

a_i, α_i: coefficients for the direct cost function used by the Non-linear Programming Strategy.

β: cost factor for the loss of quality in the integrated simulation results.

c_i: coefficient used in the Taylor expansion of LQ_i which represents the rate at which the loss of quality in results is changing with respect to τ.

$DC_i(\tau)$: the direct cost, for program i, associated with scheduling a data transfer τ units away from the current one.

DTNT: the **D**esired length of **T**ime until the **N**ext synchronized data **T**ransfer requested by program i.

k_i: the percentage of the total loss of quality in an integrated simulation's results accounted for by program i.

$LQ_i(\tau)$: the cost incurred due to loss of quality in the results provided by program i for scheduling a data transfer τ time units away from the current one.

$LQ_T(\tau)$: the cost incurred due to loss of quality in the results provided by an integrated simulation for scheduling a data transfer τ units away from the current one.

m: the number of data transfers that occurred during an integrated simulation.

M: the number of performance indicators used to determine the Mean Average Percentage Error.

MAPE: Mean Average Percentage Error.

N: the number of programs in the integrated simulation package.

NLP: Non-linear program(ming)

O: data transfer schedule which provided a converged solution.

P: data transfer schedule evaluated using the MAPE method.

Φ: a function used to represent a portion of the MAPE representation of LQ_i.

τ: the length of time between the current synchronized data transfer and the next.

t_i: the desired length of time until the next synchronized data transfer requested by program i.

$TC(\tau)$: the total cost of scheduling a data transfer τ time units away from the current one.

$v_{rs}(H)$: the value of indicator s at data transfer r if the length between transfers is H.

$V_{mM}(H)$: the matrix containing the values of the M indicators recorded at m data transfers.

w_i: the weight assigned to program i to reflect its contribution to the results of the integrated simulation. It is used by the MEANSTEP Strategy.

W_i: total cost function for program i.

List of Tables

Table		Page
4.1	Values of Parameters Determined Prior To First Comparison.	60
4.2	Ranges of K_i's Used In The Sensitivity Analysis.	62
4.3	Test Points For Sensitivity Analysis.	63
4.4	Values of Parameters Determined Prior To Second Comparison.	64
A.1	CPU Costs Attained Using The Initial And Boundary Conditions of The First Comparison.	77
A.2	CPU Costs Attained Using The Initial And Boundary Conditions of The Second Comparison.	78

List of Figures

Figure		Page
4.1	Comparison of Strategies Using Stable Operating Conditions.	67
4.2	Sensitivity Analysis of The k_i's Using Unstable Operating Conditions.	68
4.3	Comparison of Strategies Using Unstable Operating Conditions.	69

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize anyone to reproduce this thesis by photocopying or any other means, in total or in part.

Darryl W. Dormuth

The University of Manitoba requires the signatures of all persons using or photocopying this thesis. Please sign below and give address and date.

CHAPTER 1

INTRODUCTION

Many project management texts emphasize the importance of good communications among team members. Phrases like "communicating on a timely basis" and "finding out what's going on" [Meredith 1985, p 285; Frame 1987] stress qualitatively the need for team members to communicate well with others, but the cited literature does not refer to any method for quantifying how often this communication should occur. In particular, the problem of scheduling meetings for project team members to exchange information is really only discussed in the context of how to organize and run meetings but not on when they should be held. With the corporate trend towards global competition and Multi-Organizational Enterprises [Kelly 1982], holding meetings may involve gathering project members from several parts of the world. As a result, the cost associated with these meetings can consume a large portion of a project's budget. Closer examination of scheduling project meetings may provide ways to reduce this cost.

In many cases meetings are an important part of project monitoring and control [Kerzner 1984], even though they can

incur large expenses. Many project managers will schedule team meetings to monitor the performance of the project. Using information obtained at these meetings, the project manager can make informed decisions that will keep the project performance as close as possible to the project plan [Meredith 1985, p 286]. Meetings also present opportunities for team members to gain access to needed information. If meetings are never scheduled or are scheduled too far apart, a project can lose direction and end up over-budget and behind schedule, which can be summarized as poor project performance [Meredith 1985]. However, bringing back into focus the possibility of high costs associated with project meetings, the project manager can face the problem of determining how often to schedule these meetings so that cost is minimized.

To illustrate the conflict between the cost associated with and the necessity of having a project meeting, consider the following example:

Example 1.1

> An engineering firm has been assigned the project of producing a large technical report about a complex industrial process. A project leader has been appointed to oversee three other project members, each of whom is responsible for one volume

of the report, and to make sure that a high quality and consistent document is produced.

Each member works on his or her volume independently of the others but needs information from some or all of them in order to maintain consistency with the other volumes of the report. Because each member works at a rate best suited to optimize his or her individual performance, it is unlikely that everyone will want to meet at the same time. Add to this, possible problems and setbacks that each might encounter and it becomes even more unlikely that they will all want to meet at the same time.

Meetings are deemed essential by the project manager because she believes it is the only realistic means of assuring consistency in the report. If meetings are held too infrequently, a substantial amount of time, ergo money, can be lost rewriting text to regain this consistency. However, two of the project members work in the Regina office while the other member and herself work in Halifax. Consequently, meetings should be scheduled as infrequently as possible to reduce travel costs. The project manager faces the

problem of scheduling project meetings frequently enough to maintain consistency in the report while trying to keep costs associated with meetings as low as possible.

What the project manager needs in Example 1.1 is a strategy that can be used to determine when to schedule meetings. It would be preferable if several strategies were available so that the manager could choose the one best suited for her situation. However, it would be very difficult to properly test and compare these strategies on project teams because this could require complex analysis possibly using statistical methods. One suggestion to avoid this potential nightmare in initial research would be to use an analogous system on which strategies could be designed, tested and compared. The results from this research could then be used to help tackle the original problem of scheduling project meetings.

An analogous system where developed strategies can be tested and compared without the need of complex analysis is considered the following example:

Example 1.2

A large research company is responsible for performing experiments for its customers. Part of this research involves using computer programs to model (simulate) these experiments. With experiments becoming more complex, there is a need to use many programs in an integrated manner to perform the necessary analysis. In prototype form, an integrated simulation package has been developed to model one such experiment. This package is composed of four programs, each one running on its own computer and linked to the others via a Local Area Network (LAN) [Tanenbaum 1988].

Three of the programs simulate different physical components of the experiment over a specified time horizon, producing data characterizing that component's behaviour. To accomplish this, each program steps through time, performing calculations and updating relevant results. A program will adjust the size of these steps in accordance to the changing conditions of the physical component it is modelling. A program will take large steps when conditions are fairly

stable and small steps when conditions are rapidly changing. A program stepping through a simulation is similar to a person reading a piece of literature: when reading material which is easy to understand, one tends to skim over it quite quickly; when the material is difficult, one tends to step through it word for word and in some cases re-reads passages in order to make sense of what is being said.

To model the experiment in an integrated fashion, the three modelling programs exchange intermediary results during the course of a simulation. In this prototype, intermediary results are exchanged only by synchronous data transfers. Because these programs model different parts of the experiment, it is quite unlikely that all the programs will want to exchange information at the same time. It is the job of the controller to determine when data will be transferred.

At every transfer, each program exchanges intermediary results with the others and informs the controller of the desired time it would like to next transfer data. Based on these three times, the controller must determine when to schedule the

data transfer.

The analogy of Example 1.2 relates each program to a project team member, the controller program to the project manager, the synchronous transfer of data to a meeting, and the simulation to a project. The problem of scheduling meetings, which is the concern of the thesis, can be addressed in the analogous example as the problem of scheduling synchronous data transfers.

If one focuses on analyzing the problem of synchronous data transfers, develops and tests strategies on this computer system to tackle the problem, one then may be able to use these results to help address the original problem of scheduling project meetings. The benefit of this approach is that this computer system provides an excellent test-bed for repeating simulations (projects) under identical conditions and as many times as necessary to produce good statistics. This creates the potential to do a direct comparison of strategies without the involvement of possibly complex statistical analysis. It is recognized at the outset that this approach will by no means solve the original problem dealt with in this thesis. It does, however, provide a solid base to start research. The results obtained from this research may provide key insights into the problem of scheduling meetings and give focus to future research efforts.

To better understand the problem of scheduling project meetings, background research associated with scheduling meetings for project teams and with synchronous data transfers for distributive computer systems will be discussed. Then a system similar to the one described in Example 1.2 will be used to conduct research on the thesis problem in the context of synchronous data transfers. Four strategies to tackle the problem will be presented and compared using this system. Conclusions will be made in the context of this system and then will be discussed in terms of the original problem of scheduling project meetings.

CHAPTER 2

BACKGROUND

The topic of scheduling meetings for project team members is not new to the field of project management. Most of the cited literature [Meredith 1985; Frame 1987; Kerzner 1984; Schroeder 1989] devotes some discussion to scheduling meetings. However, this discussion is limited to qualitative analysis or quantitative analysis using specific examples. In order to develop generalized strategies for scheduling project meetings using quantitative methods, it would be helpful to discuss certain aspects of project management which are applicable to the problem.

"A project defines a combination of interrelated activities that must be executed in a certain order before the entire task is completed" [Taha 1987, p 469]. The writing of the technical report in Example 1.1 and the integrated simulation in Example 1.2 can be considered projects using this definition. To provide systematic and effective ways of optimizing the efficiency of executing projects, many project management techniques have been developed. Efficiency, used in this context, refers to achieving the utmost reduction in the time (scheduling) required to complete the project while accounting for the economic feasibility of using available

resources (costs). These techniques help project managers to control costs that are directly controllable by the project organization and to control the project schedule to meet established milestones.

Often project performance competes with project scheduling and project costs. Project performance refers to the characteristics of the product or service being produced by the project. In Example 1.1, performance refers to supplying a high quality report to the customer. In Example 1.2, performance refers to producing high quality data for a given simulation. Consider the following situation where performance competes with cost and scheduling: if the components being manufactured by a company have a high percentage of defects, additional hours may be allocated to the supervision of the production process to get performance up to target levels, causing changes in cost and schedule. Seldom can performance, cost and schedule be accurately predicted before the project begins and, therefore, numerous trade-offs may be required while the project is under way. [Schroeder 1989, p 386]

Administering these trade-offs falls under the category of project monitoring and control.

> "Monitoring is collecting, recording and reporting information concerning any and all aspects of project performance that the project manager and

> others in the organization wish to know." [Meredith 1985, p 285]

By monitoring key performance indicators, one should obtain a steady stream of data that indicate variances yet to come, noting that these indicate the likelihood of future problems rather than predicting with certainty. Project control involves updating the project schedule based on performing periodic analysis of the project's progress. By monitoring performance, cost and schedule, a manager can take corrective measures to ensure control over the project.

The frequency of collecting information and modifying the project schedule is usually specific to the project. This is reflected in the following statement:

> "The frequency of reporting should be great enough to allow control to be exerted during or before the period in which the task is scheduled for completion. For example, efficacy tests of drugs do not produce rapid results in most cases. Thus, there is no need for weekly (or perhaps even monthly) reports on such tests. When test results begin to occur, more frequent reports and updates may be required." (Meredith 1985, pp 292-293)

These reports should be available in time to be used for project control, they should generally correspond to project milestones and they should be scheduled in the project plan. In essence, the nature of the monitoring reports should be consistent with the logic of the budgeting, planning and scheduling systems.

Meetings can be an important part of the reporting process. Often meetings provide an effective way of obtaining quality information which can then be used to guide the next phase of the project. If meetings are scheduled in advance and the importance of them is emphasized, most individuals will make a concerted effort to be sure the information they report is accurate and up-to-date. Once all the information is reported, often steps can be taken to correct problems at the same meeting, providing a very efficient means of monitoring and controlling the project.

The frequency of meetings can be heavily dependent upon the project, the team members, and the conditions under which the project is being executed. This may account for the lack of information on generalized strategies. However, even if strategies were developed, there exists the difficulty of testing and evaluating them on a project team for a particular project. This study will attempt to bridge this gap by using a system which exhibits many similar characteristics to a project team executing a project but which has the ability to repeat a project under identical conditions, making it possible to test and evaluate possible strategies.

The following quotation is of particular interest to the problem finding a system which is analogous to a project team executing a project, where projects can be repeated under

identical conditions:

> "One way to solve a difficult problem is to break it into separate pieces and work on all the pieces at the same time. This is the whole basis for parallel (distributive) computing. A parallel (distributive) program breaks a difficult problem into pieces and then distributes them to a crowd of waiting processors. All processors work together to find a solution." [Gelertner 1990, p 213]

It should be intuitive to the reader that the breaking down of a project into tasks and the assigning of these tasks to project team members can be thought of in the above quotation by Gelertner. Thus, it seems quite reasonable to use a distributive computer system as an analogy to the system of a project team executing a project.

One such distributive computer system is the prototype computer simulation package described in Example 1.2. This package was developed, in prototype form, by Atomic Energy of Canada Limited at Whiteshell Laboratories in Pinawa, Manitoba to assist in pre-experimental and post-experimental analysis of complex experiments [McDonald 1990]. The four programs in this package are: a thermal-hydraulics program, an aerosol program, a fission product release program, and a program to control the transfer of information among the other three. The first three programs were mentioned by name to emphasize that they each model different physical properties and not to emphasize what they model. The fourth program will be discussed in more detail throughout the course of this study.

Some historical information about the prototype is provided below.

The prototype was designed to respond to the realization in the nuclear research community that aerosol transport calculations should be integrated with thermal-hydraulics calculations to effectively model experiments involving aerosol movement. In the past, these programs have been integrated by merging them into a "super-program" or by doing the thermal-hydraulics calculations first and then feeding them to an aerosol program. However, these two methods have several drawbacks such as lack of feedback and amount of computer memory needed to run the programs. The prototype system mentioned above overcomes these drawbacks by maintaining the individual structure of each program on separate computers and then having these programs communicate with each other via a computer network under the supervision of a communications controller program.

This method of keeping the programs separate and having them communicate through a computer network was also used by Nicolas, et al., [Nicolas 1990] at Electricite de France, who refer to this process as "explicitly coupling the programs". They also arrived at the conclusion that merging two software modules into one (called implicit coupling) was unmanageable. Implicit coupling can slow down the upgrading process and

penalizes users who are not necessarily interested in a coupling capability, these drawbacks are eliminated using explicit coupling [Nicolas 1990, p 787]. In their system, two explicitly coupled programs run simultaneously and remain independent, exchanging information via "pipe" files. Information exchange is dictated by one of the programs in a master-slave relationship. This type of exchange can negatively affect the performance of the slave program if it is forced to step through the simulation at a rate different from what its numerics dictate [Press 1989, pp 615-622].

The prototype designed at Whiteshell Laboratories uses the controller program to overcome the possible negative effects of forcing a program to step through a simulation at a different rate than what its numerics dictate. The controller program decides when data are to be synchronously transferred among the programs. At each synchronous data transfer, data are transferred among the programs and each program informs the controller of the next time it would like to transfer information. Based on these times, the controller program must determine when the next transfer should occur. The analogy of this prototype to a project team executing a project was explained on page 7.

To tackle the problem of determining when to schedule synchronous data transfers among computer programs, literature

from the computer science fields of distributive systems and network theory was reviewed [Agha 1990; Durfee 1988; Gelertner 1990; Smith 1983; Tanenbaum 1988]. These authors discussed the topic of scheduling synchronized data transfers by using examples of specific distributive or parallel systems and not by using any generalized analysis. Despite the limited information on the subject of scheduling synchronized data transfers, some of the examples provided helpful insights into the problem of scheduling synchronous data transfers and reinforces the need for research in this area.

Distributive processing is an example of course-grained multi-processing [Tazelaar 1990] and encompasses the concept of distributing the components of a program to waiting processor nodes which contain memory and a processor [Gelertner 1990]. The prototype simulation package used in this paper employs distributive processing techniques to perform the necessary simulations.

The literature reviewed on distributive computer systems suggests three common patterns in parallelism where research is being focused [Agha 1990]. The first pattern, pipeline concurrency, involves the enumeration of potential solutions and the concurrent testing of these solutions as they are enumerated; the second, divide and conquer, involves the concurrent elaboration of difficult sub-problems and the

joining of some or all their solutions in order to obtain a solution to the overall problem; the third pattern, cooperative problem solving, involves each object (program) carrying out its own individual computational process, and communicating with other objects to share the intermediate results it has computed. An example of a cooperative problem solving system would be a simulation where physical objects are represented by logical (computational) objects as is the case with the prototype system discussed in this paper. Therefore, literature on this pattern was researched in more detail.

Two books by Durfee [Durfee 1988] and Smith [Smith 1983] provided strong insights into the notion of cooperative problem solvers by addressing algorithms for building powerful problem solvers that effectively utilize the characteristics of distributive processor architectures. These authors focused their research on distributed architectures where control is decentralized and where the nodes are loosely-coupled (i.e., they spend a far greater percentage of time doing computations and not communicating with others), communicate via messages, and cooperate to find a solution to an overall problem. This form of distributive architecture is categorized as a nearly decomposable system, i.e., a system in which interactions among the components are weak but not negligible. The short term behaviour of each component is

relatively independent of the behaviour of the others; the long term behaviour of each depends only in an aggregate way on the behaviour of others.

Smith suggests that a metaphor for a distributive problem solving system is a group of human experts experienced at working together, trying to complete a large task. In examining the operation of this group, a major interest is the way in which they interact to solve the overall problem, the manner in which the workload is distributed among them, and how results are integrated for communication outside the group. This suggestion by Smith re-enforces the idea of using a distributive problem solving system to represent a project team working on a project.

Smith categorizes the major concerns of designing a distributive problem solver into three following layers:

> (1) Architecture - the lowest layer where research is concerned with the design of individual nodes and the communication system that connects them.
>
> (2) Systems - the second level where research is addressing issues like the design of communication protocols and the design of local operating systems to manage the resources of the individual nodes.
>
> (3) Problem Solving - the top layer where the concerns are internode control and knowledge organization; in particular, how to achieve effective problem-solving behaviour from a collection of asynchronous nodes.

In discussing the three layers of distributive problem solvers, Smith states that:

> "The bulk of research has been restricted to the systems and architecture levels. It has generally been assumed that a well-defined and a priori partitioned problem exists and the major concerns lie in an optimal static distribution of subtasks, optimal methods for interconnecting processor nodes, resource allocation, and prevention of deadlock. Complete knowledge of the problem has also been generally assumed (i.e., explicit knowledge of timing and precedence relations between tasks) and the major reason for distribution has been assumed to be load-balancing." [Smith 1983, p 3]

In order to construct a distributive problem solver rather than a distributive processor, one must focus on the problem solving layer.

The work published by Durfee and Smith provides a framework for constructing problem solvers in a distributive processing environment. The framework specifies mechanisms for communication, control, and knowledge organization. The communication component provides the basis for node interaction, the control component specifies the possible modes of interaction between processor nodes, and the knowledge organization component specifies how knowledge is organized in individual nodes and distributed throughout the collection of nodes. The key issue resolved in this framework is how tasks are distributed among the processor nodes. For the most part, Durfee and Smith assume that communication among nodes is done asynchronously and not synchronously. Thus the literature on distributive problem solvers did not provide much insight into scheduling synchronized data

transfers, although it did provide valuable insights about the architecture of the integrated simulation package.

Literature on synchronized data transfer in distributive computer systems was reviewed in areas other that distributive problem solvers but most of the information was narrowly focused on database systems [Agha 1990; Milenkovik 1981; Chambus 1984]. After reviewing further literature on database systems there was the realization that the concept of synchronization was different from the concept used in the context of this study. This difference can be shown in a multi-access database which handles airline ticket bookings by various travel agents. In the context of this study, the problem of synchronization deals with deciding when each agent should access the database to receive and modify ticket information under the restriction that the agents all must access the database at the same time. From the point of view of Milenkovic [Milenkovic 1981], the problem of synchronization deals with preserving the consistency of the database, i.e., ensuring that all data are consistent with the actual transactions and that all data are accessible to every agents.

The concept of synchronizing data transfers presented by Milenkovic and other computer scientists [Chambus 1984] does not require that all agents communicate at the same time and,

in fact, they prefer that the system avoid such situations. Their main concern is that all information is current and that if two or more agents try to access the same piece of information (either by writing to or reading from it) at the same time, the system will make sure that everyone gets the proper information as efficiently as possible. This difference in concept of synchronization means that strategies for synchronous data transfers in multi-user database systems are not directly applicable to the problem encountered in the prototype simulation package.

Other Articles [Fugimoto 1990; Geihns 1990; Back 1990; Avrunin 1985] provided many interesting insights into distributive problem solving systems; however, the authors did not discuss generalized strategies for synchronous data transfers.

After completing this initial background research in the areas of project management and distributive computer systems, we decided to develop some intuitive and fairly basic strategies rather than devote more time to another literature search. Future research effort will, of course, include a more extensive literature review.

The background research for this study is summarized as follows:

(1) A literature review was conducted in the field of project management which yielded little information on any generalized quantitative strategies for scheduling project meetings

(2) It was decided that a system analogous to a project team executing a project be used on which projects could be repeated as often as necessary to obtain good statistics for evaluating strategies.

(3) It was determined that a prototype integrated simulation package possessed the characteristics discussed in (2). The problem of scheduling project meetings was then transformed into the problem of scheduling synchronous information transfers.

(4) A literature review was conducted in the fields of distributive computer systems and network theory but little information was available on generalized strategies for scheduling synchronous data transfers.

(5) It was decided to develop, test, and compare strategies for the integrated simulation package rather than devote more time to another literature review.

The next chapter, Chapter 3, discusses the development of four strategies that determine the optimal schedule for transferring data in the integrated simulation package.

CHAPTER 3

METHODS USED FOR SYNCHRONIZED DATA TRANSFER

3.1 Introduction

As discussed in the previous chapter, the reviewed literature suggests there is little information available on generalized quantitative strategies for scheduling project meetings [Meredith 1985; Frame 1987; Kelly 1982]. This study proposes to make a start on developing such strategies by using a small and controllable system which is representative of a project team executing a project. Using this system, generalized strategies will be developed, tested, and compared. This initial study will hopefully provide some direction for future research into the problem of scheduling project meetings.

This chapter will focus on developing four deterministic, intuitive and straightforward strategies for scheduling synchronized data transfers for a prototype integrated simulation package, discussed in Example 1.2. Before proceeding with the discussion on these strategies, the integrated simulation package first will be described in more detail.

3.2 Detailed Description of The Integrated Simulation Package

A system representative of a project team executing a project is the prototype integrated simulation package developed by Atomic Energy of Canada Limited Research Company (AECL Research) at Whiteshell Laboratories in Pinawa, Manitoba, to assist in pre-experimental and post-experimental simulations of a complex experiment (see Example 1.2) [McDonald 1990]. Scientists at AECL Research designed the prototype by consolidating existing documented, verified programs on to a distributive computer system to produce an integrated simulation package. This package is composed of four individual programs, three of the programs model different physical properties of the experiment while the fourth (the controller) controls the synchronous transfer of data among the other three. Each program runs on its own computer and communicates with the others via a Local Area Network (LAN) [Tanenbaum 1988]. The methods the programs use to model specified physical phenomena are discussed in Example 1.2. The analogy of this system to a project team executing a project is explained on page 7.

The three modelling programs exchange intermediary results during the course of the simulation to model the experiment in an integrated fashion. In this prototype, intermediary results are exchanged only by synchronous data

transfers.

At every transfer, each program exchanges intermediary results with the others and informs the controller of its desired time for the next data transfer (this will be denoted as DTNT or Desired Time of Next Transfer). It is quite unlikely that all the programs will want to exchange data at the same time, i.e., all the DTNT's will be equal, because these programs model different physical components of the experiment. It is the job of the controller to determine when to schedule these synchronous data transfers, based on these three DTNT.

This rest of this chapter will focus on developing strategies which the controller program can use to schedule the data transfers. Three of the strategies use heuristics or "rules of thumb" [Taha 1987, p 15] to provide a straightforward approach to scheduling the transfers and the fourth uses concepts of non-linear programming to attack the problem.

3.3 General Assumptions Used By The Strategies

Each of the four strategies discussed in this study makes different assumptions in order to determine the schedule for

synchronized data transfers. Assumptions that are specific to a strategy are addressed in the respective description of the strategy, found in the appropriate section of this chapter. Assumptions that are common to all strategies are dealt with in this section.

3.3.1 Objective

It is assumed that the objective of a strategy is to minimize the cost associated with the synchronized transfer of data. Because each of the strategies deals with the cost associated with synchronized data transfer in a different manner, this cost will be discussed in more detail in the following sections. For the sake of simplicity, the strategies discussed in this paper represent cost using only a single objective function. The possibility of developing strategies that represent cost using multiple objective functions is left for future research.

3.3.2 Performance Measurement

When implementing a project, one often refers to the term project performance when discussing how well a project is meeting its pre-defined goals [Meredith 1985, p 285]. There

are key factors or indicators that can be measured to rate a project's performance, once a project is complete. This concept can be used for the integrated simulation package. In this case, a simulation's performance will be determined by the quality of the data produced in a "reasonable" time frame [McDonald 1990]. To measure a simulation's performance, a cost will be assigned to the quality of the data and to the time taken to complete the simulation. The assignments of these costs are specific to the strategy and will be discussed in the respective sections.

3.3.3 Solving By Iterative Techniques

As stated on page 26 of this chapter, at each transfer, the programs exchange intermediary results and inform the controller of their DTNT. Using the three DTNT, the controller must determine when to schedule the next data transfer. Each strategy is iterative, that is, a strategy uses only the DTNT supplied at a given data transfer to determine the next data transfer. Once the time for the next data transfer arrives, the strategy uses the DTNTs supplied at that transfer to determine the next transfer time. This process repeats until the completion of the simulation.

Another approach that could be used in developing a

strategy is to determine a complete schedule for data transfer prior to commencing the simulation. In a project setting, this would involve making a calendar of all project meetings to be held during the execution of a project, prior to the project starting. Intuition reveals that this method could work if the project is well-defined and the risk of major complications is low. However, if major complications do arise this calender may quickly become obsolete. The iterative process suggested in this study allows more flexibility to deal with unforeseen complications.

During the course of a simulation, each strategy determines the time for the next data transfer using only the DTNTs supplied by the programs, i.e., the next data transfer time is a function of only the DTNTs. To represent this concept, the following definitions are introduced.

Definition 3.1

Let t_i represent the desired length of time until the next synchronized data transfer (DTNT) requested by program i. (i=1,2...,N $t_i \geq 0$)

Definition 3.2

Let $\tau(t_1,t_2,\ldots,t_N)$ represent the length of time between the current synchronized data transfer and the next. ($\tau \geq 0$)

Some of the strategies may, however, use additional information obtained prior to the start of the simulation to determine τ.

For the integrated simulation package discussed in this paper, N = 3 (there are three programs in the system) and the unit of measurement for t_i and τ is simulation seconds. Note that a simulation second is one second relevant to the transient which a program is simulating and is not necessarily equivalent to one second of real time.

3.3.4 Complexity of A Strategy

It would defeat the purpose of optimizing the scheduling of data transfers if a strategy is not efficient in determining the next data transfer. Therefore, the four deterministic strategies introduced in this paper are basic and straightforward, providing the controller with an efficient means of scheduling data transfers.

3.4 Heuristics

Heuristics are methods which seek a good solution to a problem, but not necessarily an optimal solution, based on

rules of thumb.

> "The advantage of heuristics is that they normally involve less computations when compared to exact solutions. Also, because they are rules of thumb, they normally easier to explain to users who are not mathematically oriented." [Taha 1987, p 15]

To capitalize on these advantages, this section introduces three heuristics that can be used to schedule synchronous data transfers in a deterministic and iterative manner.

3.4.1 MINSTEP Strategy

When a measure of a simulation's performance weights the quality data heavily over the completion time, i.e., failure to secure high quality data could be costly, a strategy which is risk-adverse to possible inaccuracies in the data is desirable. In this case, the controller should monitor the simulation as closely as possible. The MINSTEP algorithm provides a good rule of thumb to schedule meetings under these conditions. For project managers, this strategy can be used when quality of the project performance is a major concern.

In this strategy:

$$\tau = \min_{i} \{t_i\} \quad i=1,2,..,N$$

This strategy simply chooses the minimum of the DTNTs as the time to schedule the next transfer.

3.4.2 MAXSTEP Strategy

When one needs a simulation to provide only a rough approximation of the behaviour for an experiment and there is a desire to complete the simulation quickly, i.e., the measure of the project performance weights the completion time over the quality of data, the controller will want to transfer data as infrequently as possible. The MAXSTEP algorithm provides a suitable scheduling strategy for these conditions. In a project management setting, this strategy can be used when meeting costs are a major concern.

For this strategy:

$$\tau = \max_{i} \{t_i\} \quad i=1,2,\ldots,N$$

This strategy simply chooses the maximum of the DTNTs as the time to schedule the next transfer.

The two heuristics, MINSTEP and MAXSTEP, represent the extremes of the four strategies developed during the course of this study. For MINSTEP, data transfers are scheduled using the earliest of desired times for next transfer given by the

three programs. In this case, programs will have the most recent data to work with, therefore, the overall quality of the results produced by the simulation should be high. However, transferring data frequently can use a lot of computer time preparing the data for transfer and end up being quite costly. If the MAXSTEP algorithm is used, the costs associated with computer time may be less but so may be the quality of the simulation results. When used in cases where the cost associated with the loss of quality in the results is high, the MAXSTEP algorithm could be expensive. In order to use these strategies, one should be confident that one of the costs, cost of computer time or cost of quality of data, greatly outweighs the other through the course of the simulation. To summarize, there appears to be limitations on the number of applications for which the MINSTEP or the MAXSTEP strategies can be used.

To bring the above paragraph back into the focus of scheduling project meetings, let us expand on the situation in Example 1.1. As was stated in that example, two of the project members (M1 and M2) reside in Regina and two reside in Halifax (M3 and PM). Suppose that the meetings are held in Halifax and that over the course of two months M1 wants to meet every 4 weeks; M2, every 3 weeks; and M3, every week. Using the MINSTEP strategy the PM would schedule meetings every week as requested by M3. However, the costs of flying

M1 and M2 to Halifax eight times for a meeting will be very high. On the other hand, if the MAXSTEP strategy is used, the PM would schedule meetings every four weeks as requested by M1. This would result in only two trips to Halifax, substantially reducing the cost of airfare. However, if M3 does not get the information he needs every week his volume could contain substantial erroneous information by the time of the next meeting. This may result in time consuming, and therefore costly, rewrites. If the PM chooses the wrong strategy to schedule project meetings, she could be penalized with unnecessary costs.

The MEANSTEP strategy offers a little more flexibility than the other two heuristics by assigning a weighted mean of the t_i's to τ. This strategy will produce a τ that lies in between the τ's produced by MINSTEP and MAXSTEP for the same set of t_i's.

3.4.3 MEANSTEP Strategy

In most cases, high quality data is wanted in as short a time as possible. This requires a strategy that exhibits behaviour of both the MINSTEP and MAXSTEP algorithms. The MEANSTEP algorithm schedules data transfers based on a weighted average of the times submitted by the programs. The

weights are assigned to each program by the controller (actually, the programmer of the controller) prior to commencing the simulation and reflect how each program contributes to the overall simulation performance.

For this strategy:

Let $w_i \equiv$ weight assigned to program i, $0 \leq w_i \leq 1$.

Then $\tau = w_1 \cdot t_1 + w_2 \cdot t_2 + \ldots + w_N \cdot t_N$

where $w_1 + w_2 + \ldots + w_N = 1$

Note that if all the weights are equal, τ will be the arithmetic mean of the t_i's.

For the MEANSTEP strategy, τ is represented only by the magnitudes of the t_i's and the w_i's. The effectiveness of this strategy rests on how well the w_i's are chosen and the assumption that they do not change during the course of a simulation. However, in many simulations the weight assigned to a program may fluctuate depending on the behaviour of the physical component it is modelling. For example, if the flow of water through a pipe is stable during the initial part of an experiment and then varies rapidly during the later part, the program modelling the thermal-hydraulic component should

have a small weight assigned to it during the initial part of the simulation and a larger weight assigned when the flows start varying to reflect its contribution to the simulation results. Though the MEANSTEP strategy is more flexible than the other two heuristics, the restrictions of accurately assigning weights and assuming that these weights are held constant throughout the simulation limits the number of applications for which this strategy can be used.

Expanding on the situation described on page 33 for Example 1.1, the PM could assign weights to each team member to reflect his or her contribution to the overall report and then use the MEANSTEP strategy to schedule project meetings. This would result in meetings being scheduled somewhere between one and four weeks apart for those two months. The effectiveness of this strategy relies heavily on these weights initially assigned by the PM. If the contributions of the team members to the report change over the two months, the assigned weights may become obsolete, making the MEANSTEP strategy less effective in minimizing the cost of scheduling meetings.

The heuristics described in this section are static strategies used for scheduling synchronized data transfers, i.e., the rules for scheduling data transfers are set prior to commencing and do not change during the course of the

simulation. The next evolutionary step is to develop a dynamic strategy by relaxing the assumption of constant w_i's. The fourth strategy, the Non-Linear Programming (NLP) strategy, does relax this assumption by using a more rigorous approach to minimizing the cost of data transfer than the three heuristics. The following section gives a detailed description of this strategy.

3.5 Non-Linear Programming (NLP) Strategy

As discussed in the previous section, the three heuristics can cause extra costs to be incurred for a simulation, if they are used in an inappropriate situation. A strategy that is robust to the ways which a simulation's performance can be determined is desired. The NLP Strategy is a step towards this desired strategy. The NLP Strategy reflects the changing conditions of the simulation by dynamically adjusting its objective function at every data transfer. Once the objective function is determined, a non-linear program is formulated and solved. The solution provides the controller with an optimal time to schedule the next data transfer. This section describes, in detail, how the objective function is determined and how the non-linear program is formulated and solved.

3.5.1 The Objective Function

The NLP Strategy assigns a cost to having a synchronous data transfer and a cost to the loss of quality in a simulation's results. These two costs, in essence, represent the two factors which determine a simulations performance. The objective of the NLP strategy is to minimize the sum to these two costs. The sum of the two costs (the total cost) is represented by the following function:

$$TC(\tau) = [\sum_{i=1}^{N} DC_i(\tau)] + LQ_T(\tau) \qquad (3.1)$$

where

$TC(\tau)$ represents the total cost of scheduling a data transfer τ units away from the current one.

$DC_i(\tau)$ represents the direct costs, for program i, associated with scheduling a data transfer τ units away from the current one. The coefficients for the function DC_i are determined prior to commencing the simulation. This function is discussed in more detail in Sub-section 3.5.2.

$LQ_T(\tau)$ represents the costs due to loss of quality in the data associated with scheduling a data transfer τ units away from the current one. The function LQ_T is evaluated at each data transfer. This function is discussed in more detail in Sub-section 3.3.5.

Recall that for the integrated simulation package discussed in this study, N = 3.

3.5.2 Deriving DC_i

For the case of the integrated simulation package which is used in this study, the direct cost was defined to be sum of all costs which were directly accountable to a synchronized data transfer. The direct costs for each program were measured by the cost of computer time taken to complete the program's portion of the simulation. The cost of computer time was calculated by taking the number of CPU seconds used to complete a simulation multiplied by a cost factor. Note that a CPU second is one second of real time that a process has used the Central Processing Unit of the computer. The CPU second was used as the unit of measure because, unlike real time, it was independent of how many processes the CPU is working on.

In reference to Example 1.1, direct costs could include travel costs, the portion of a member's salary that is consumed at meetings, costs attributed to the time each member spends preparing for a meeting, and other overhead costs, like hotel accommodations, that could be incurred at a meeting. In essence, direct costs are all those costs that can be directly

attributed to having a meeting.

It can be inferred that as the frequency of data transfers increases, so too do the direct costs. In fact, one can state that as the length of the interval of time between data transfers approaches zero, the direct costs associated with them approaches infinity. In the objective function of this non-linear program, a function must be developed to represent the direct cost component. Below is the definition of the direct cost function, which will be used throughout the remainder of this study.

Definition 3.3

Let $DC_i(\tau)$ represent the direct costs associated with scheduling synchronous data transfers τ time units apart for program i. This function has the following properties:

(i) $DC \geq 0$ for $\tau > 0$
(ii) $DC \rightarrow \infty$ as $\tau \rightarrow 0$

Note that DC_i is not defined at $\tau = 0$ as this would imply that data transfers are held continuously.

For the case of the integrated simulation package which was used in this study, it was assumed that DC_i could be represented by the hyperbolic function in Equation 3.2. The coefficients a_i and α_i define the concavity of the function.

$$DC_i(\tau) = a_i\tau^{\alpha_i}$$
$$\text{where } a_i, \tau \geq 0 \ ; \ \alpha_i \leq 0 \qquad (3.2)$$

The coefficients a_i and α_i, in Equation 3.2, were determined for each program prior to the commencement of and remained constant throughout a simulation.

3.5.3 Deriving LQ_T

When a program depends on data from an external source, i.e., another program, there can be a relationship between the quality of the data obtained and the quality of the result produced by that program. If the data is outdated and possibly incorrect the quality of the program's result may suffer. To what extent this occurs depends on the relationship between the data and the results produced by program.

In order to assess the overall loss of quality in the results produced by an integrated simulation, it was required that the controller know how data from one program affected the results of another program. It should be emphasized at this point the reviewed literature in the area of distributive

processing systems (and in the area of project management) did not address the concept of loss of quality in a program's results (or a project member's performance) in any significant detail. With this in mind, following simplifying assumption was made: let the loss of quality in the results produced by an integrated simulation be a function dependent on the loss of quality in the results produced by each program.

The concepts of loss of quality in the results produced by an integrated simulation and the loss of quality in the results produced by a program are expressed in the following definitions:

Definition 3.3

Let $LQ_i(\tau)$ represent the cost due to loss of quality in program i's results incurred by scheduling a data transfer τ time units away from the current one. It is assumed that $LQ_i(\tau)$ has the following property:

(i) $LQ_i(\tau) \geq 0$ for all $\tau \geq 0$

Definition 3.4

Let $LQ_i(\tau)$ be defined by Definition 3.3, then $LQ_T(LQ_1, LQ_2, \ldots, LQ_N)$ is defined as a function of all the $LQ_i(\tau)$'s and represents the cost due to loss of quality in the results for the entire simulation incurred by scheduling a data transfer τ time units away from the current one. It is also assumed that property (i) in Definition 3.3 holds for LQ_T.

The function, LQ_i, is specific to the program and the problem. The development of these functions was by no means straightforward. Future research along much the same lines as utility theory [Lee 1985] may provide more insight into representing loss of quality with a mathematical function. In Definition 3.4, LQ_T was expressed as a function of all the programs' LQ_i's, however, it is possible to give a stronger definition. When investigating how the quality of one programs results affects the quality of the overall results, it seems quite logical to concentrate on examining the rates of changes of the various parameters.

With the assumption that the functions defined in **Definition 3.4** are at least once differentiable with respect to their independent variables, the chain rule can be used for multi-variable functions to derive the following mathematical expression:

$$\frac{dLQ_T}{d\tau} = \sum_{i=1}^{N} \frac{\partial LQ_T}{\partial LQ_i} \cdot \frac{dLQ_i}{d\tau} \qquad \textbf{(3.3)}$$

Equation **3.3** represents the relationship between the quality of the overall results provided by an integrated simulation to the quality of results provided by each program.

This is done by assuming that the rate of change of the quality in the results of an integrated simulation is a function of the contribution each program provides to the overall results (represented by the partial derivatives) and the rate of change of the quality of results for each program (represented by the straight derivatives).

To understand these concepts in terms of project management, let us refer back to Example 1.1. Three of the project team members are responsible for writing a volume of the report. In order to maintain consistency in the report, each team member requires information from the others. The quality of each team member's respective volume will depend, at least in some part, on the quality of the information received from the other members. If a team member obtains information from another which is later found to be incorrect, that member will incur costs in rewriting the erroneous section. Using Definition 3.3, one can represent the costs associated with loss of quality of information in a member's respective volume as a function of the time between meetings; using Definition 3.4, one can represent the quality of the entire report as a function of the quality of the three volumes; using Equation 3.3, one can define even a stronger relationship between the quality of each volume and the quality of the overall report. It should be noted that the partial derivatives in Equation 3.3 should be determined by

the project manager because each member could be biased towards her or his contribution to the report.

To proceed with the formulation of a function to represent LQ_T, the following was assumed:

$$\frac{\partial LQ_T}{\partial LQ_i} = k_i \ (constant) \qquad (3.4)$$

For further simplification the k_i's were normalized:

$$\sum_{i=1}^{N} k_i = 1 \qquad (3.5)$$

With these assumptions, each k_i represents the percentage of the total loss of quality in the simulation performance accounted for by program i. For instance, k_3 equalling 0.85 means that program 3 (fission product release program) has an 85% influence over the quality of the results produced by the integrated simulation (the other two programs have a combined influence of 15%). Note that the k_i's in the NLP Strategy have a similar role to the w_i's in the MEANSTEP Strategy.

To determine LQ_i, it is assumed that M indicators are selected prior to commencing a simulation for program i to represent M data values. For instance, one indicator for program 2 (aerosol program) could be the variable which represents the mass of aerosol located in one portion of the piping network of the experiment. Then, during the course of a simulation, the value of every indicator for each program is reported at every data transfer. Upon completion of a simulation, a mXM matrix, V_{mXM}, is produced by each program where m represents the number of data transfers that occurred during that simulation.

The loss of quality in the results produced by a program (LQ_i) will be represented as the Mean Absolute Percentage Error (MAPE) [Oberstore 1990] in the value of the recorded indicators as compared to a converged solution. This is expressed as follows:

$$LQ_i(P) = \frac{\beta}{m \cdot M} \sum_{r=1}^{m} \sum_{s=1}^{M} \left| \frac{V_{rs}(P) - V_{rs}(O)}{V_{rs}(O)} \right| \qquad \textbf{(3.6)}$$

where

ß represents Cost factor for loss of quality in the results for program i ($/MAPE).

$v_{rs}(H)$ represents the value of indicator s at data transfer r if data transfers are H time units apart, i.e., it represents the rs element of the matrix V(H).

P represents the data transfer schedule used in the simulation.

O represents the data transfer schedule which produced converged solution to the problem. Note that $v_{rs}(O)$ does not equal 0 for any value of r or s.

This is an appropriate metric to use for two reasons: it directly compares a calculated solution to the converged solution and it produces a comparable result regardless of the magnitudes of the values composing the different matrices. It should be noted that in order to use this metric, the matrices, P and O, being compared have to be the same size. To accomplish this, the values of matrix V(P) are interpolated to correspond to the data transfer time times of V(O).

For the sake of clarity the subscript i is dropped from the following calculations and it is assumed that these calculations are performed for each program.

To express LQ as a function dependent only on τ, v_{rs} is assumed to be a function which is at least twice differentiable. The function Φ_{rs} is then defined as follows:

$$Let\ \phi_{rs}(p) = \left| \frac{v_{rs}(p) - v_{rs}(o)}{v_{rs}(o)} \right| \qquad (3.7)$$

If $v_{rs}(p)$ and $v_{rs}(o)$ are assumed to have the same sign, the absolute value signs can be removed and Φ_{rs} can be expressed as:

$$\phi_{rs}(p) = \begin{cases} \dfrac{v_{rs}(p) - v_{rs}(o)}{v_{rs}(o)}, & v_{rs}(p) \geq v_{rs}(o) \\ \dfrac{v_{rs}(p) - v_{rs}(p)}{v_{rs}(o)}, & v_{rs}(p) < v_{rs}(o) \end{cases} \qquad (3.7a)$$

Due to symmetry, the following calculations will use the case where $v_{rs}(p) \geq v_{rs}(o)$, i.e.,

$$Let\ \phi_{rs}(p) = \frac{v_{rs}(p) - v_{rs}(o)}{v_{rs}(o)} \qquad (3.8)$$

by Taylor's expansion we can write

$$v_{rs}(p) = v_{rs}(o+\tau) = v_{rs}(o) + \tau v'_{rs}(o) + \tfrac{1}{2}\tau^2 v''_{rs}(o) + \cdots$$

or

$$v_{rs}(o+\tau) - v_{rs}(o) = \tau v'_{rs}(o) + \tfrac{1}{2}\tau^2 v''_{rs}(o) + \cdots$$

which implies

$$\phi_{rs}(p) = \frac{1}{v_{rs}(o)}\left[\tau v'_{rs}(o) + \tfrac{1}{2}\tau^2 v''_{rs}(o) + \cdots\right]$$

and therefore

$$\frac{d\phi_{rs}(p)}{d\tau} = \frac{1}{v_{rs}(o)}\left[v'_{rs}(o) + \tau v''_{rs}(o) + \cdots\right]$$

If only the first two terms of the Taylor expansion are used we have the following expression:

$$\frac{d\phi_{rs}(p)}{d\tau} = \frac{v'_{rs}(o)}{v_{rs}(o)} + \tau\frac{v''_{rs}(o)}{v_{rs}(o)}$$

$$Let \quad c_{1_{rs}} = \frac{v'_{rs}(o)}{v_{rs}(o)} \quad , \quad c_{2_{rs}} = \frac{v''_{rs}(o)}{v_{rs}(o)} \qquad (3.9)$$

$$then \ \frac{d\phi_{rs}(p)}{d\tau} = c_{1_{rs}} + c_{2_{rs}}\tau$$

(Note: One may question the need to go through all of these calculations if Φ_{rs} is assumed quadratic. This was done to show that Φ_{rs} is expressible in the form of a polynomial and that the assumption of it being quadratic is a reasonable one.)

From Equations 3.6 and 3.7, we have the following relationship:

$$\frac{dLQ(p)}{d\tau} = \frac{\beta}{m \cdot M}\sum_{r=1}^{m}\sum_{s=1}^{M}\frac{d\phi_{rs}(p)}{d\tau} \qquad (3.10)$$

Equation 3.10 states that LQ'(p) is expressible as a linear combination of $\Phi_{rs}'(p)$ and can be thus written as

$$\frac{dLQ(p)}{d\tau} = c_1 + c_2\tau \qquad \textbf{(3.11)}$$

For each program in this simulation package, the following initial conditions are assumed for each LQ function:

(1) LQ(0)=0 - this implies that there is no loss of quality with scheduling meetings 0 simulation seconds apart.

(2) LQ'(0)=0 - this implies that if meetings are scheduled 0 simulation seconds apart, a minimal loss of quality is achieved.

Using these two assumptions we can state the LQ function for each individual in the form

$$\frac{dLQ_i(\tau)}{d\tau} = c_{2_i}\tau = c_i\tau \qquad \textbf{(3.12)}$$

and LQ_T in the form

$$\frac{dLQ_T(\tau)}{d\tau} = \sum_{i=1}^{N} k_i c_i \tau \qquad (3.13)$$

where

k_i represents the contribution made by program i to the overall simulation results.

c_i represents the rate at which the loss of quality in the results provided by a program is changing with respect to τ.

3.5.4 The Non-linear Program

Using the functions for DC_i and LQ_T derived in Sub-sections 3.5.2 and 3.5.3,the non-linear program can be expressed as follows:

$$\begin{aligned} &Min \quad TC(\tau) = \sum_{i=1}^{N} [a_i \tau^{\alpha_i} + \tfrac{1}{2} k_i c_i \tau^2] \\ &s.t. \\ &\qquad \tau \geq 0 \end{aligned} \tag{3.14}$$

where

N represents the number of programs in the simulation package (in this study N = 3)

a_i and α_i represent the coefficients of the direct cost function for program i which are determined prior to commencing a simulation.

k_i represents the contribution program i makes to the overall results of an integrated simulation. It is determined prior to commencing a simulation.

c_i represents the rate at which the loss of quality in the results provided by program i is changing with respect to τ.

Note that the only constraint used in this non-linear program to model the data transfers in the integrated simulation package was to make sure τ was greater than 0.

3.5.5 Solving the Non-linear Program

The non-linear program expressed in Equation 3.14 is solved at every data transfer to provide the next time for the data transfer. In order to solve this non-linear program, all the parameters in the objective function must be determined.

The coefficients a_i and α_i are all determined prior to commencing the simulation. Appendix A describes how these coefficients are determined. The k_i's are also determined prior to commencing the simulation and, for this study, they were evaluated by estimating how important each program's results were to the results of the overall simulation.

The c_i coefficient for program i is evaluated at each data transfer. This coefficient reflects how fast the physical model, which program i simulates, is changing. The faster things are changing, the more often the program will want to transfer data. To determine c_i, a total cost function is needed for each program. This can be expressed as

$$W_i(\tau) = DC_i(\tau) + LQ_i(\tau) = a_i\tau^{\alpha_i} + c_i\tau^2 \qquad (3.15)$$

where the coefficients are defined as in Sub-section 3.5.4.

Recall from Sub-section 3.3.3 , that at every data transfer, each program supplies its desired time for the next data transfer. In this study, this time, t_i, was considered to represent the minimum of W_i for program i. Therefore, the following can be stated:

$$\frac{dW_i(t_i)}{d\tau} = a_i\alpha_i t_i^{\alpha_i-2} + 2c_i t_i = 0$$
$$\Rightarrow c_i = \frac{-a_i\alpha_i t_i^{\alpha_i-2}}{2} \qquad \textbf{(3.16)}$$

Equation 3.16 shows that for program i, c_i is a function of t_i. If it is assumed that program i determines its desired time for the next transfer based on how conditions are changing in the system it's representing, then c_i will reflect these changing conditions. In essence, LQ_i function dynamically represents the loss of quality in results provided by program i as conditions change during the course of a simulation.

The formulation for the NLP Strategy is now complete.

3.6 Summary

We have now developed four algorithms to schedule data transfers among programs in the integrated simulation package. Section 3.4 discussed three heuristics which can be implemented to schedule these transfers and Section 3.5 introduced a dynamic method which uses non-linear programming techniques to attack the problem. Intuitively, the non-linear program should provide a solution which is close to the best for most conditions. However, with the assumptions made to derive it, this may not always be so.

The next chapter discusses a method for comparing these strategies to determine under what conditions each should be used.

CHAPTER 4

COMPARISON OF STRATEGIES FOR SCHEDULING SYNCHRONIZED DATA TRANSFERS

4.1 Method of Comparison

Each of the strategies developed in Chapter 3 provides a means of determining when to schedule data transfers among the programs in the integrated simulation package. The next step in this study was to compare these strategies using a method which produced results that can be analyzed in a quantitative manner.

The method of comparison used in this study evaluated each of the strategies based on total cost of completing an assigned simulation: the strategy that provided the cheapest total cost was termed the best strategy.

The total cost was evaluated as the sum of the direct cost and the cost incurred due to loss of quality in the results provided by the integrated simulation, where:

Direct costs were calculated for each program by determining the CPU cost consumed to complete its portion of

the simulation. The CPU costs were calculated by multiplying the CPU time a program took to complete its portion of the simulation by a cost factor obtained from the Whiteshell Laboratories Computer Centre [McDonald 1990]. The direct cost for the overall integrated simulation was the sum of the all the programs' direct costs.

Cost incurred due to loss of quality in the results was determined for each program by comparing the results obtained from its portion of the simulation to a pre-determined converged solution using Mean Average Percentage Error (MAPE) method (see p 46). This method produced a value indicating the percentage which the results deviated from the converged solution. A cost was determined by multiplying this value by a cost factor ß (which has the unit of measure $/MAPE). Because assigning a value to ß is dependent on factors such as programmer's value of data quality and on the particular problem being simulated, the cost incurred due to loss of quality in results provided by program i was expressed as a linear function of ß. The cost incurred in loss of quality in results provided by the integrated simulation was the average of the programs' costs.

The costs for the comparison used dollars ($) as the unit of measure.

To illustrate this concept of total cost, consider the following situation: after completing a simulation using the MAXSTEP Strategy for data transfer, it was determined that Program 1 incurred $8.05 in direct costs and $0.645ß in costs due loss of quality in its results, Program 2 incurred $4.90 in direct costs and $0.400ß in costs due to loss of quality, and Program 3 incurred $5.50 in direct costs and $0.330ß in costs due to loss of quality in the results. The total cost which this strategy incurred for this simulation was 8.05 + 4.90 + 5.50 + (0.645ß + 0.400ß + 0.330ß)/3 or $18.45 + $0.485ß.

Comparisons were made using two sets of initial and boundary conditions for the experiment. Two sets were the minimum number required to determine any dependence a strategy has on experimental conditions. The first set of conditions represented the experiment running under stable operating conditions; the second condition represented unstable operating conditions.

In each of the two comparisons, a simulation was repeated four times, each time a different strategy was used. Once the four simulations were complete, the total costs calculated for each strategy were compared and the strategy with the lowest cost was declared the best. Note that since these costs were represented as linear functions of ß, some strategies were

declared the best for certain values of ß, but not the best for other values.

4.2 First Comparison: Stable Conditions

For this comparison, the initial and boundary conditions for the simulation reflected the experiment operating under stable conditions. Prior to performing the comparison, it was expected that the MAXSTEP strategy would be the best strategy for small values of ß (see Sub-section 3.4.2), the MINSTEP Strategy would be the best strategy for large values of ß, the MEANSTEP Strategy would lie somewhere in between the MAXSTEP and MINSTEP strategies (see Section 3.4), and the NLP Strategy would always be close to the best strategy for all values of ß (see Section 3.5).

Table 4.1 lists the values of the various parameters used by the MEANSTEP and NLP strategies that were determined prior to commencing the comparison:

TABLE 4.1: Values of Parameters Determined Prior To First Comparison

PROGRAM	w_i	k_i	a_i	α_i
1	0.333	0.333	2.7	-0.968
2	0.333	0.333	21	-0.958
3	0.333	0.333	0.51	-0.929

Note that:

w_i is the weight assigned to program i to represent its contribution to the simulation results. It is used by the MEANSTEP Strategy.

k_i represents the contribution made by program i to the quality of the integrated results produced by a simulation. It is used by the NLP Strategy.

a_i and α_i are coefficients used by the direct cost function for the NLP Strategy. The calculations for these coefficients are shown in Appendix A.

The results of the comparison are shown in Figure 4.1. Note that for line representing the total cost incurred by each strategy, the y-intercept is the direct cost and the slope is the Mean Average Percentage Error.

The first thing to be noted about the results is that there are different strategies which yield the best result, depending on the value of ß. As was expected, the MAXSTEP strategy was better than MINSTEP for low values of ß, and the MINSTEP strategy was close to overtaking MAXSTEP as ß became large. The MEANSTEP strategy behaved better than expected: it yielded a total cost which was better than either MINSTEP or the MAXSTEP for most of the domain of the graph (i.e., $5 < ß < 45$). It was not expected that the NLP strategy would always yield the best total cost; however, it was expect that the NLP strategy would produce a total cost that was close to the best strategy for that value of ß. It seems to do accomplish this for the range of ß chosen, i.e.,

$0 < \beta < 50$.

Clearly, for this set of initial and boundary conditions, the MEANSTEP Strategy or the NLP Strategy provided a strategy which is close to the best for all values of ß.

4.3 Sensitivity Analysis On the K_i's

To test the hypothesis that the k_i's might be better chosen, a sensitivity analysis was performed on the k_i's using the second set of initial and boundary conditions prior to doing the second comparison. The combination which yielded the best total cost was then used in the second comparison.

To begin the sensitivity analysis, ranges for the k_i's were determined using best judgment. The ranges used in this analysis are shown in Table 4.2:

TABLE 4.2: Ranges of k_i's Used In The Sensitivity Analysis

PROGRAM	Range for k_i
1	0.15 - 0.95
2	0.10 - 0.90
3	0.05 - 0.60

Using a simplex method for mixture sensitivities [15], the mixtures of k_i's shown in Table 4.3 were determined as the test points for the sensitivity analysis.

TABLE 4.3: Test Points For The Sensitivity Analysis			
Test Point	k_1	k_2	k_3
S1	0.85	0.10	0.05
S2	0.30	0.10	0.60
S3	0.15	0.25	0.60
S4	0.15	0.80	0.05
S5	0.33	0.33	0.33

The results of the sensitivity analysis shown in Figure 4.2.

The narrow band in Figure 4.2 tends to indicate that the cost for the NLP strategy is insensitive to the mixture of k_i's. As a results, S5 was chosen as the values of k_i's used in the second comparison.

4.4 Second Comparison: Unstable Conditions

For this comparison, the initial and boundary conditions for the simulation reflected the experiment operating under unstable conditions. The strategies were expected to behave in a similar manner to the way they did in the first comparison. However, the costs for the strategies were expected to increase because the simulation was operating under unstable conditions and, therefore, more data transfers

were expected.

Table 4.4 lists the values of parameters used by the MEANSTEP and NLP strategies, determined prior to commencing the comparison:

TABLE 4.4: Values of Parameters Determined Prior To Second Comparison

PROGRAM	w_i	k_i	a_i	α_i
1	0.333	0.333	3.1	-0.896
2	0.333	0.333	23	-0.920
3	0.333	0.333	0.50	-0.937

Note that the calculations for the a_i and the α_i are shown in Appendix A, the w_i were simply made to represent each program contributing equally to the simulation, and the k_i were determined in Section 4.3.

The results of this comparison are shown in Figure 4.3.

The behaviour of the three heuristics, in this comparison, are as expected: the MAXSTEP strategy is better than MINSTEP for smaller values of ß, the MINSTEP is better than the MAXSTEP for larger values of ß, and the MEANSTEP yields a cost that lies in between the two other heuristic for the range of ß ($0 < ß < 50$). The NLP strategy did not perform as expected. There are few reasons that could explain why

this was the NLP Strategy behaved unexpectedly in this comparison. First, it was assumed that the k_i's were constant. This assumption may have to be relaxed. Second, the LQ_i's were assumed to be quadratic. This assumption was based on knowledge about the behaviour of the numerical methods used by the programs [McDonald 1990], but may have to be re-evaluated in future research. Third, the MAPE method, used to measure the loss of quality in a program's results, may not be suitable for this particular application. It was chosen because it was an easy method to apply to the problem. Due to time constraints, there was no evaluation of other methods that could be used on the problem. Future research should address evaluating different methods.

Clearly, for this set of initial and boundary conditions, the MEANSTEP strategy should be used as it provides near-to-the-lowest cost for the defined values of ß.

4.5 Summary of Results

In this chapter, the four strategies developed in Chapter 3 were quantitatively compared using the method described in Section 4.1. Two comparison were performed, representative of different operating conditions, and a sensitivity analysis was done on the k_i's. The strategies performed as expected except

in the second comparison where the NLP Strategy behaved unexpectedly. Reasons for this unexpected behaviour could be attributed to the assumptions made in formulating the non-linear program and were discussed in detail in Section 4.4.

To conclude this chapter, it should be stressed that the objectives of the comparisons were met: they demonstrated that a system which is representative of a project team executing a project can be used to evaluate and compare different strategies, and they showed that no single strategy is the best in the studied situations.

Figure 4.1
Stable Conditions

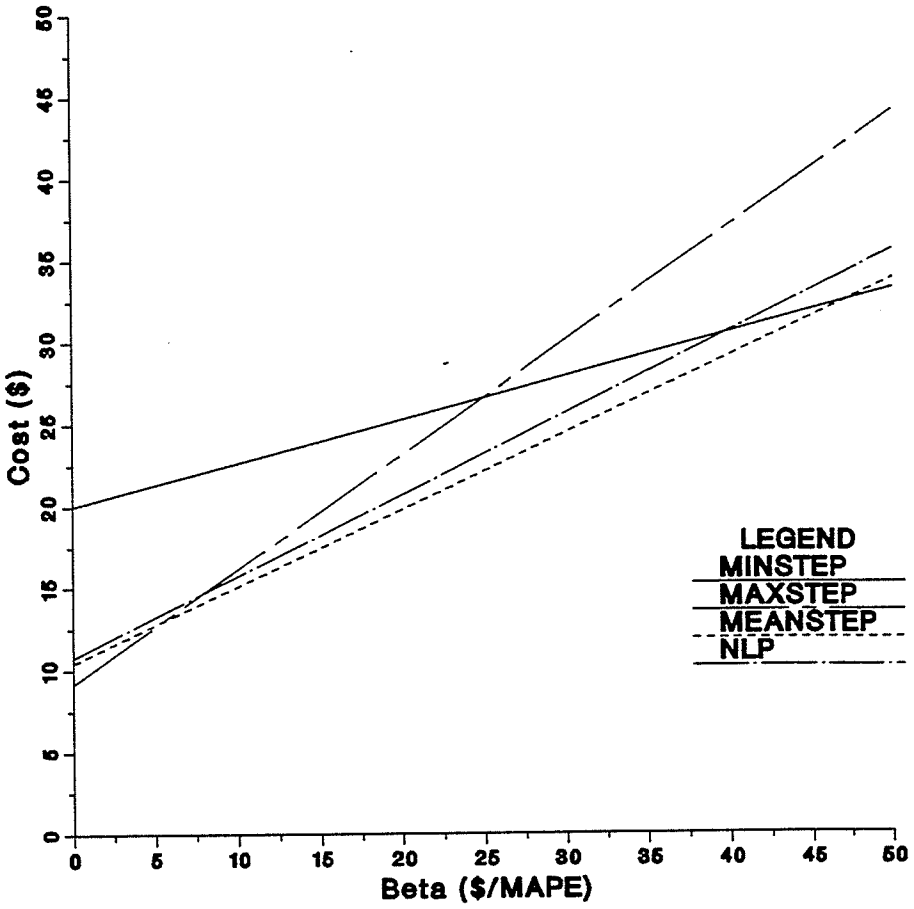

Figure 4.1: Comparison of Strategies Using Stable Operating Conditions.

Figure 4.2
Sensitivity Analysis of KI

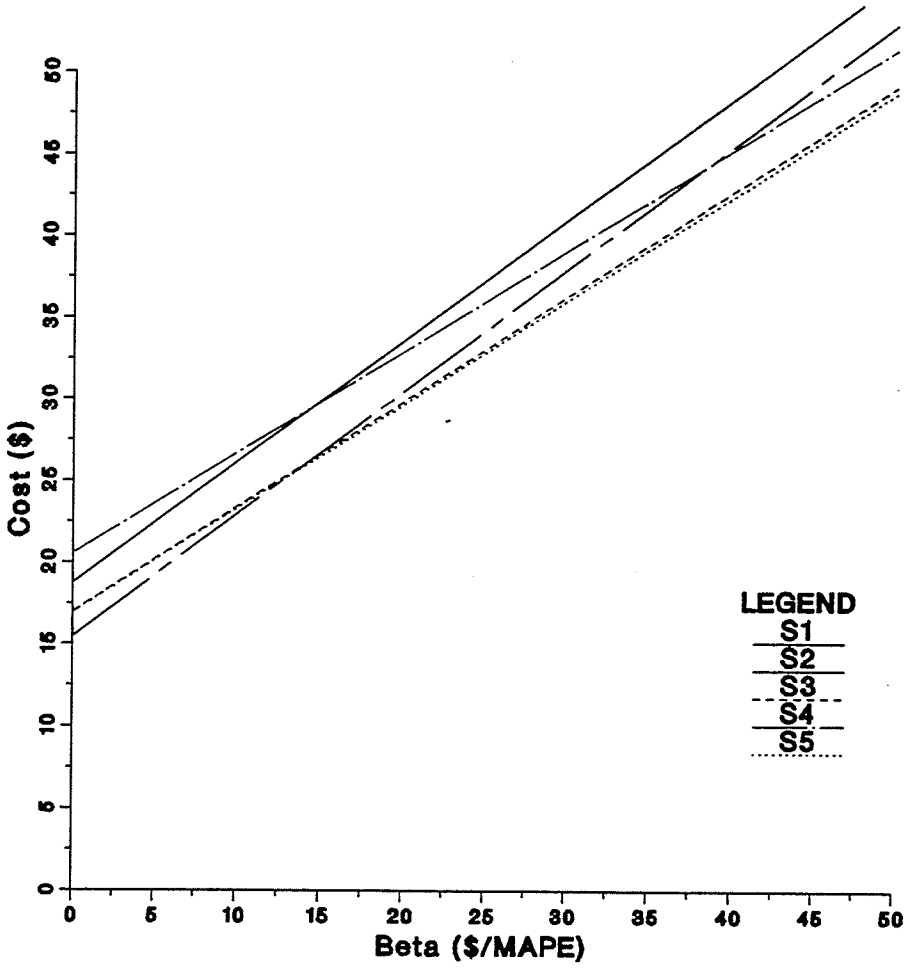

Figure 4.2: Sensitivity Analysis of The k_i's Using Unstable Operating Conditions.

Figure 4.3
Unstable Conditions

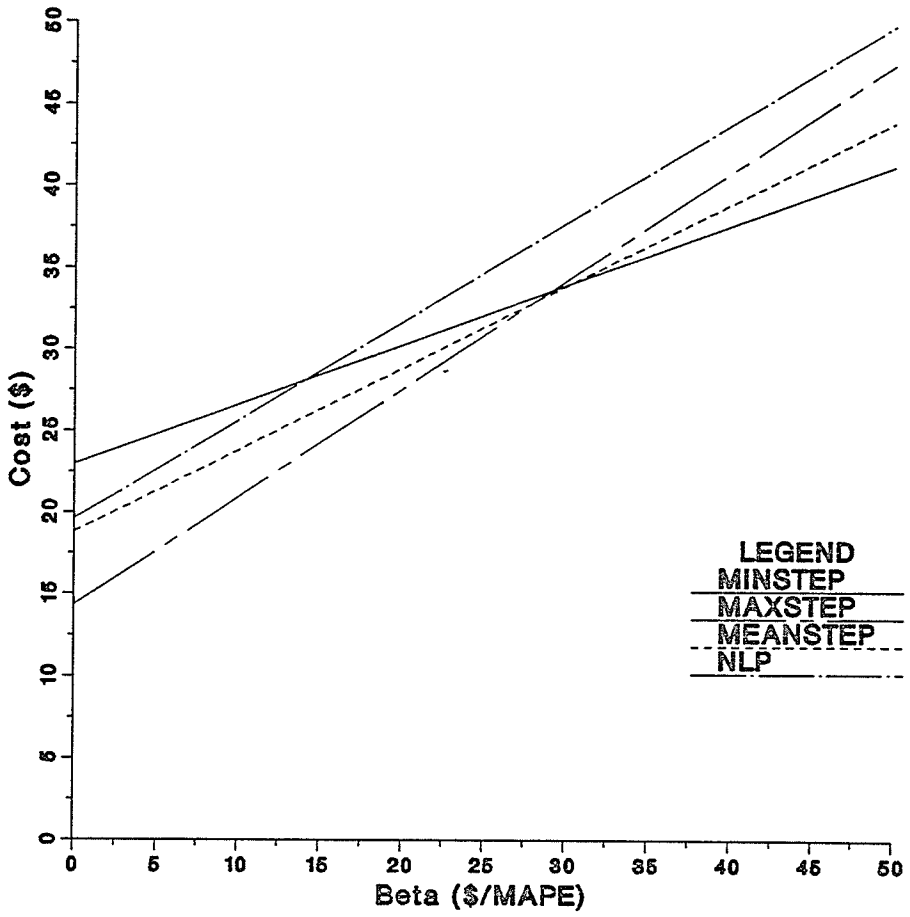

Figure 4.3: Comparison of Strategies Using Unstable Operating Conditions.

CHAPTER 5

CONCLUSIONS AND SUGGESTED FUTURE RESEARCH

5.1 CONCLUSIONS

With today's corporate trend toward global competition and Multi-Organizational Enterprises [Kelly 1982], holding project meetings may involve gathering members from all around the world. These meetings can, therefore, consume a large portion of a project's budget. Literature reviewed in the area of project management suggested that little research had been done on developing generalized strategies for scheduling project meetings [Meredith 1985; Frame 1987]. The literature did analyze this problem in a qualitative manner, to some extent, but any quantitative discussion was restricted only to specific examples.

To shed more light on the problem, a system analogous to a project team executing a project was suggested. This system was an integrated simulation package which used distributive problem solving techniques to model a large scientific experiment. The analogy is that programs in this package represent members of a project team and that transferring data synchronously represents having a meeting. Literature

reviewed in the areas of distributive problem solving and distributive processing provided more discussion on quantitative methods to schedule data transfers but the discussion was in the context of specific examples and not on any generalized strategies [Smith 1983; Durfee 1988].

To proceed with the research, four simple strategies were developed and tested on this analogous system. These developed strategies were general enough that they could be used on systems that have characteristics similar to those described Example 1.1 and Example 1.2. Three of the strategies used heuristics to provide a solution to the problem; the fourth used non-linear programming techniques. The results obtained from this study suggest that more research should be devoted to the problem of scheduling synchronous data transfers and to scheduling project meetings.

In the context of the integrated simulation package, three specific observations were made.

It was demonstrated that generalized quantitative strategies could be developed for this system and that these strategies could be compared in a quantitative manner to determine which one provides the best solution for a given problem.

Depending on how important the quality of the results of the simulation were, a strategy could be the best in some instances and not the best in others. In one comparison, when the importance of the quality of results (measured in terms of cost) was low, one strategy was the best; when the importance of quality was high, another strategy was the best (see Section 4.4). This indicates, in the context of this study, that there was no single strategy that was always the best, implying that more research should be done on developing generalized strategies for synchronous data transfers.

When developing the three heuristics, it was realized that a more dynamic strategy may provide a more robust to the number of applications on which it could be used and, therefore, it would be more preferable. A fourth strategy was introduced which used a more rigorous approach to solve the problem. It was not expected that this solution would be the best for all cases; however, it was expected that to always provide a near-to-the-best solution, thus providing a robust strategy to solve a given problem. The results of the two comparisons suggest that this strategy does fulfil its

objective for certain cases, but not for all cases. One reason for this may lie in some of the assumptions (discussed in Section 3.5) that were used to formulate the non-linear program. These assumptions may have to be relaxed in order to provide a more robust strategy. Another reason may be how the LQ_i functions are derived. Future research on this strategy should include other methods for deriving these functions.

In the context of scheduling project meetings, from the research done on the integrated simulation package, the following observations are stated:

The frequency at which project meetings are scheduled may impact on the overall performance of the project.

A strategy for scheduling project meetings may work well in certain circumstances, but may not work well in others.

More research should be done to develop generalized strategies for scheduling project meetings, as it may provide ways to improve project performance.

5.2 SUGGESTED FUTURE RESEARCH

One topic which should provide interesting future research would be investigating, in greater detail, the relationship of an individual's quality of performance to that of the entire project. In this study, an initial attempt was made to represent this for the integrated simulation package in the Non-Linear Programming (NLP) Strategy using Equation 3.3. Further studies into how an individual's performance affects the performance of others and, ultimately, the performance of the entire project may provide managers with useful information for scheduling meetings.

The second topic would be expanding on the NLP strategy to make it a more robust model. In addition to relaxing some or all of the assumptions made in the formulation suggested in Chapter 3, further research should include looking at formulations requiring more constraints and the feasibility of solving large non-linear programs for this type of problem. For the comparisons done in this paper, which had only one constraint on the time between data transfers (τ), the NLP strategy did not behave as expected in one comparison (see Section 4.4). However, if a problem requiring more constraints was used, the NLP strategy may perform better

since it can cope with additional constraints which the other strategies can not.

APPENDIX A

CALCULATION OF THE COEFFICIENTS FOR THE DIRECT COST FUNCTIONS

Recall Equation **3.2**

$$DC_i(\tau) = a_i \tau^{\alpha_i} \qquad \text{where } a_i, \tau \geq 0 \; ; \; \alpha_i \leq 0 \tag{3.2}$$

which represents the direct cost incurred by each program for transferring data τ time units away from the current transfer. The coefficients, a_i and α_i, are assumed to be fixed and have been assigned values prior to commencing the simulation.

The method described below was implemented in both comparisons (see Section 4.2 and Section 4.4) to determine the values of the a_i's and α_i's for each program's direct cost function. To determine these values, seven simulations were performed, each using a different fixed value for τ to schedule data transfers. For example, one simulation scheduled data transfers every 0.1 simulation-seconds (sim-sec) apart, thus for the 120 second simulation, there were 1200 data transfers. When each simulation was complete, the CPU costs were determined for each program by taking the length of time needed to complete its portion of the

simulation and multiplying that number by a cost factor supplied by the Whiteshell Laboratories Computer Centre [McDonald 1990]. The CPU cost was then assumed to represent the direct costs incurred by that program.

Once all simulations were performed and there was a CPU cost corresponding to each of the seven different values of τ, a non-linear least squares numerical method (Press 1989, pp 521-528; Box 1969, pp 20-24) was used to determine the a_i's and α_i's for each program.

Table A.1 shows the CPU costs (in dollars) for each program and each value of τ using the initial and boundary conditions supplied in the first comparison.

TABLE A.1: CPU Costs Attained Using The Initial And Boundary Conditions of The First Comparison

τ	0.1	0.2	0.5	1.0	2.0	5.0	10.0
PROGRAM 1	24.76	12.55	5.07	2.66	1.49	0.90	0.80
PROGRAM 2	195.05	98.61	40.39	22.38	13.18	8.43	7.86
PROGRAM 3	4.34	2.24	0.95	0.50	0.31	0.18	0.15

After performing the regression analysis, the following values were obtained for the coefficients (expressed as a 95% confidence interval):

PROGRAM 1: $a_1 = 2.7 \pm 0.6$ $\alpha_1 = -0.968 \pm 0.002$

PROGRAM 2: $a_2 = 21 \pm 5$ $\alpha_2 = -0.958 \pm 0.002$

PROGRAM 3: $a_3 = 0.51 \pm 0.14$ $\alpha_3 = -0.929 \pm 0.002$

Table A.2 shows the CPU costs (in dollars) for each program and each value of τ for Comparison B.

TABLE A.2: CPU Costs Attained Using The Initial And Boundary Conditions of The Second Comparison

τ	0.1	0.2	0.5	1.0	2.0	5.0	10.0
PROGRAM 1	24.78	12.51	5.37	3.12	2.30	2.05	1.78
PROGRAM 2	192.91	98.05	40.80	23.14	16.61	12.29	9.61
PROGRAM 3	4.32	2.18	0.93	0.50	0.30	0.18	0.14

After performing the regression analysis, the following values were obtained for the coefficients (expressed as a 95% confidence interval):

PROGRAM 1: $a_1 = 3.1 \pm 2.2$ $\alpha_1 = -0.896 \pm 0.002$

PROGRAM 2: $a_2 = 22 \pm 11$ $\alpha_2 = -0.920 \pm 0.002$

PROGRAM 3: $a_3 = 0.50 \pm 0.14$ $\alpha_3 = -0.937 \pm 0.002$

These results were used by the Non-linear Programming Strategy to determine the objective function in the both comparisons.

REFERENCES

Agha, G., "Concurrent Object Oriented Programming", Communications of the ACM, Vol. 33(9), September 1990, pp 125-141

Avrunin, G.S., and Wilede, J.C., "Describing And Analyzing Distributive Software System Designs", ACM Transactions On Programming Languages and Systems, Vol. 7(3), July 1985, pp 380-403

Back, R.J.R, and Kurki-Suonio, R., "Distributive Cooperation With Action Systems", ACM Programming Languages and Systems, Vol. 10(4), October 1988, pp 513-553

Box, M.J., Davies, D., Swann, W.H., Non-linear Optimization Techniques, Oliver & Boyd Ltd., Edinburgh, Scotland, 1969

Chambus, F.B., Duce, D.A., and Jones, G.P., Distributive Computing, Academic Press, London, England, 1984

Durfee, E. H., Coordination of Distributive Problem Solvers, Kluwer Academic Publishers, Boston, Massecussets, 1988

Frame, J. D., Managing Project In Organizations, Jossy-Bass Inc., San Francisco, California, 1987

Fujimoto, R., "Parallel Discrete Event Simulation", Communications of the ACM, Vol. 33(10), October 1990, pp 30-53

Geihns, K., and Hollberg, U., "Retrospective On DACNOS", Communications of the ACM, Vol. 33(4), April 1990, pp 439-448

Gelertner, D., and Philbin J., "Spending Your Free Time", BYTE Magazine, Vol. 15(5), May 1990, pp 213-219

Kelly, A.J., New Dimensions In Project Management, D.C. Heath and Co., Toronto, Ontario, 1982

Kerzner, H., Project Management: A Systems Approach To Planning, Scheduling And Controlling (2nd Ed.), Macmillian of Canada, Agincourt, Ontario, 1984

Lee, S.M., Moore, L.J., and Taylor, B.W., Management Science (2nd Ed), Wm. C. Brown Publishers, Durbuque, Iowa, 1985

McDonald, B.H, Wallace, D.J, Hanna, B.N, and Dormuth, D.W., "INTARES: A Concept Code For Integrated Thermal-hydraulics And Aerosol Safety Analysis", presented at the CSNI Workshop on Aerosol Behaviour and Thermal-hydraulics in the Containment, Fontenay-aux-Roses, France, 26-28 November 1990

Meredith, J.R., Project Management: A Managerial Approach, John Wiley and Sons Inc., New York, New York, 1985

Milenkovik, M., Update Synchronization In Multiaccess Systems, UMI Research Press, Ann Arbour, Michigan, 1981

Nicolas, G., Briere, E., Lux, F., and Marguet, S., "Easy Explicit Coupling Between Two Large Codes Using CRAY UNICOS", Third International Conference On Simulation Methods In Nuclear Engineering, Montreal, Quebec, 18-20 April 1990, pp 784-796

Oberstore, J., Management Sciences: Concepts, Insights and Applications, West Publishing Co., St. Paul, Minnesota, 1990

Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T., Numerical Recipes The Art of Scientific Computing, Cambridge University Press, Cambridge, England, 1989

Smith, R.G., A Framework For Distributive Problem Solving, UMI Research Press, Ann Arbour, Michigan, 1983

Strategy of Experimentation, E.I. du Pont de Nemours and Co., Willmington, Delaware, 1975

Taha, H.A., Operations Research: An Introduction (4th ed.), Macmillian Publishing Co., New York, New York, 1987

Tanenebaum, A.S., Computer Networks (2nd ed.), Prentice Hall Inc., Englewood Cliffs, New Jersey, 1988

Tazelaar, J., "Desktop Supercomputing", BYTE Magazine Vol. 15(5), May 1990, p 204