A Global Approach To Project Optimisation

Thesis submitted to the University of Edinburgh, for the degree of Doctor of Philosophy.

By:

Anthony Martyn Harding

Department of Civil and Environmental Engineering, University Of Edinburgh.

For which of you, intending to build a tower, does not first sit down and estimate the cost, to see whether he has enough to complete it? Otherwise, when he has laid the foundation and is not able to finish, all who see it will begin to ridicule him, saying "this fellow began to build and was not able to finish."

Luke 14, verses 28-30, NRSV

Abstract

In recent years there has been a growing trend in major projects towards considering the whole project life cycle and it's costs. This analysis initially considered concept to commissioning, although more recently it has been expanded to consider concept to decommissioning. Within this project life cycle optimisations are performed to obtain efficient solutions. However, contemporary optimisation techniques still tend to focus on improving the performance of small subsystems within the project as a whole. An optimisation of the whole project across the entire project life cycle is not normally carried out.

This thesis therefore proposes a general methodology for such a global optimisation model, which will allow the whole project to be considered within a single optimisation. The strategy consists of a basic scheme for input of all the project data to an objective function, as well as the definition of its constraints. The objective function can map to a value relating to cost, time, performance or risk. This allows the most important criteria to be maximised or minimised, as well as constraining other important criteria.

The validity of the model is tested by applying it to the resource constrained project scheduling problem. This involves solving scheduling and resource allocation problems. In a large construction project these problems would be a subsystem of the global project model. By testing the global approach to optimisation on this problem using a genetic algorithm, and comparing it to locally optimising techniques, optimisation performance is shown to be improved by applying the global approach.

Finally, the global approach is successfully applied to a case study. The results of both the case study and the simple problems demonstrate the global approach to be both feasible and advantageous for large construction projects.

Declaration

This thesis is the result of research undertaken at the Department of Civil and Environmental Engineering, University of Edinburgh, and is submitted for the degree of Doctor of Philosophy. It is declared that this thesis, and the work contained therein, is original research work conducted solely by the author unless clearly stated otherwise, under the supervision of Dr. D A Ponniah. None of this work has been submitted to any university or institute for any other degree or qualification.

Previously Published Papers

Harding, A. M., Ponniah, D. A. *A global approach to project optimisation*, First Asia Pacific Conference on Offshore Systems: Mobile and Floating Structures, 10-11 December 1996, Kuala Lumpur, Malaysia.

Harding, A. M., Ponniah, D. A. *Experience with a global optimisation approach to project scheduling and resource allocation*, Offshore Europe '97: Continuous Change – Learning from the 21st Century, Aberdeen, 9-12 September 1997, Society of Petroleum Engineers.

Harding, A. M., Ponniah, D. A. *Experience with a global-optimization approach to project scheduling and resource allocation*, Journal of Petroleum Technology, April, 1998, pp116-127, ISSN 0149-2136.

Acknowledgements

In the completion of the work presented here and the preparation of this thesis the author would like to thank David Ponniah for all his help, guidance and support. Thanks are also due to James Patterson for providing the data for the 110 sample RCPS problems, and to Harry Fleetwood Bird for providing the data for the JIA car park project.

In addition, the author would like to express his thanks to all his friends and family, for their support. Particular thanks are due to Richard Boyd and Justin Robinson, whose willingness to give a second opinion was often invaluable in solving many of the problems encountered along the way.

Contents

List of Figures

List of Tables

Abbreviations

AoA	Activity on Arrow.
AoN	Activity on Node.
CPM	Critical Path Method.
FF	Finish-Finish
FS	Finish-Start
GA	Genetic Algorithm.
LCC	Life Cycle Costing.
MCS	Monte Carlo Simulation.
MEST	Minimum Early Start Time.
MINSLK	Minimum Slack.
PERT	Programme Evaluation and Review Technique.
PRU	Primary Resource Use.
RCPS	Resource Constrained Project Scheduling.
SF	Start-Finish
SS	Start-Start

Glossary of Terms

Activity on Arrow A graphical representation of a network where the project activities are represented by arrows, and precedence relationships by nodes connecting the activities.

Activity on Node A graphical representation of a network where

Activity Priority A numerical value which the scheduling algorithm uses to rank activities in the queue. Activities with higher priorities are placed higher in the queue, and will therefore be scheduled first.

Activity Selection Techniques Algorithms which determine how the scheduling algorithm determines which activities are scheduled and which delayed in a

Children New solutions generated by the GA each generation.

Critical Path Method The traditional method of calculating the duration of a project network and obtaining a critical path.

Crossover In the GA, the process of swapping values of project input variables of two children.

Feasibility Rate In the GA, the proportion of runs which are feasible, usually expressed as a percentage.

Feasible Region The full set of all feasible solutions.

Finish-Finish A relationship which dictates that the activity may not be completed before its preceding activity finishes.

Finish-Start A relationship which dictates that the activity may not begin before its preceding activity finishes.

Fitness	The fitness of a solution in the GA is a numerical value which measures how good a value of the objective function it possesses. The fitness of a population is a generalisation on the fitness of each of its members.
Generation	Unit of measure of execution of the GA. Each execution of the basic loop of a GA's operation constitutes the advancement of the GA from one generation to the next.
Genetic Algorithm	The optimising algorithm used for all the optimisation problems analysed.
Global Optimum	The solution to the project which represents the best possible project performance, as defined by some measure of project success.
Life Cycle Costing	A technique which not only provides the cost of including an item within a project, but also the cost of maintaining it throughout the operational life of the project.
Local Optimum	The mathematical optimum of a small subsystem of the project.
Minimum Early Start Time	The minimum time at which an activity may commence.
Minimum Slack	A scheduling heuristic which schedules activities whose value of float by network calculation is the lower ahead of those whose float is higher.
Monte Carlo Simulation (1)	A stochastic technique which permits the likely outcomes of a project to be evaluated when risk is considered.

Monte Carlo Simulation (2)	A simple optimising technique which selects a large number of possible solutions entirely at random, evaluates them and selects the best.
Mutation	In the GA, the act of giving a project input variable a new, randomly generated value.
Objective Function	The objective function is the project model which maps from the project input data to the particular project output criterion being optimised. The value of the objective function is the value of this project output criterion.
Parents	In the GA, a pair, or number of pairs, of existing solutions which will be used to generate new children for the population.
Population	A set of solutions which the GA manipulates in its search for an optimum or near-optimal solution.
Primary Resource	The resource whose rate of consumption within an activity determines its duration.
Primary Resource Use	The allocation of the primary resource to an activity per time period. This value determines the activity duration, and, when it is variable, permits the variation of an activity's duration.
Programme Evaluation and Review Technique	As CPM, but is sometimes referred to as methodology for implementing CPM as well as the technique itself.
Project Input Data	All the data which defines the project.

Project Input Variables	Project input data which is variable, rather than fixed. The project optimisation aims to find the optimum values of these variables in terms of the project output criteria.
Project Output Criterion	A measure of project success. This is usually a numerical value, and may be maximised or minimised as part of the global optimisation, or have limits imposed on it.
Resource Constrained Project Scheduling	Scheduling a project (determining when activities should begin and end) while ensuring that no resource constraints are violated.
Runs	In the GA, the number of runs completed is the number of times the objective function has been evaluated, which is equivalent to the number of times the scheduling algorithm has been executed.
Scheduling Algorithm	An algorithm which iterates through all the activities in the project and determined when they should begin and end, while ensuring that all precedence relationships are met and there are no resource conflicts.
Solution Set	A set of project input variables, with its corresponding set of project output criteria. This is also frequently abbreviated to **Solution**, particularly in the discussion of the results of optimisations.
Start-Finish	A relationship which dictates that the activity may not finish before its preceding activity has begun.
Start-Start	A relationship which dictates that the activity may not begin before its preceding activity has begun.

Chapter 1 Introduction

Construction project management is a fundamental element of civilisation. Most of our quality of life depends on large constructions, such as buildings, roads, power stations and water supply networks to name but a few. These are all products of construction projects which have required careful planning and control of their design, construction and execution in order to be successful. This planning and control constitutes project management, and project management is considered to be effective when this planning and control is performed successfully.

In times gone by the actual process of managing a project received little attention. However, in today's highly competitive construction industry, engineers and project managers are keen to increase their competitiveness by executing projects more efficiently. In addition to this, clients expect projects to be completed more quickly, more cheaply and to a higher quality. Because of these two factors, the ways in which better levels of project performance can be achieved has become a significant field of study, particularly in the last 50 years.

This research may be broken into two fields. Firstly, there is analytical, quantitative, research. This aims to measure and to model the project, and use mathematical techniques to create solutions to problems. Secondly, there is qualitative research. This, recognising that projects involve people and therefore have a strong relational element, focuses on creating an environment in which the project can succeed. This is achieved by looking at the whole project, the people involved, their relationships and their priorities.

In the last ten years or so the analytical side of project management – particularly those analytical techniques which have been developed more recently – has received increased criticism. This criticism has asserted that current techniques tend to be restricted to analysing small subsets of the project rather than the whole[Kerzner, 1995; Rodrigues, 1996; Walker, 1989], and that they often represent clever mathematical solutions to abstract problems rather than accurate models of what actually happens within a project. In addition to this, the fact that these techniques

consider only a small subset of the full scope for optimisation available within the project means that they often tend to find solutions which, though good solutions to the problem defined, are not good solutions for the real project at all[Pidd, 1989].

While the criticism of analytical techniques is valid, it must not be assumed that analytical techniques cannot play an important part in project planning. Indeed, analytical techniques play a vital part in project planning. In the face of this criticism it is necessary not to do away with analytical techniques, but to create a new paradigm which is not restricted by the shortcomings of the existing techniques. In order to avoid the criticism which has been raised against analytical techniques in the past, such paradigm is required to fulfil the following criteria:

1. The technique must be capable of numerically modelling the whole project, from concept to decommissioning. It must also be capable of supporting different measures of project success.

2. The technique must be capable of modelling all projects as accurately as possible.

3. There is a trade-off between computational effort and how good a value the optimisation will be able to find. A reasonable balance must be aimed for, where sufficiently good solutions may be obtained without an excessive and expensive amount of computational effort being required to do so.

This thesis proposes such a paradigm: a global approach to project optimisation. This approach looks at the project globally. That is to say, rather than employing a number of optimisations which all act on small subsystems of the project (local optimisations), the whole project is optimised using one single, global project model.

This thesis defines the core framework for this model, and a mechanism for creating, from this framework, an accurate analytical model of the whole project.

The model framework uses a resource constrained scheduling algorithm as its core, but has been defined in such a way as not to restrict it to modelling any particular stage of the project's execution. Nor is it restricted to particular types of project, analytical techniques or measures of project success. This core model is defined,

along with the mechanisms for applying it to any particular project or type of project. It is also explained how the flexibility provided by using this core model allows the project to be modelled as accurately as possible. Provided techniques are selected which reflect the actual behaviour and interrelationships in the project, the global project model created will be accurate.

The aim of project optimisation has also been redefined to fit in with this new paradigm. Rather than considering the optimum solutions as the absolute mathematical optimum of the system in terms of a single measure of project success, the optimum is considered as the best solution which may be found given the constraints on the project and the available time and computational resources. Thus, rather than defining a true mathematical optimum which would require unrealistically large amounts of computational resources to find, and which might represent a solution which is not acceptable in terms of the true measures of project success, a realistic and useful optimum solution is defined.

1.1 Testing the model

In order to test this theory two case studies were performed. The first was on a simple set of 110 projects which had been used in the literature to test resource constrained scheduling algorithms. The global approach was applied, and a model formulated. A computer program which allowed this model to be implemented, and optimisation to be performed, was created.

These models were optimised by minimising the only measure of project success which existed for these projects: execution time. In order to show how the global approach performed when the true mathematical optimum was sought, the amount of computational resources used by the optimisation was limited. The results of the optimisations performed using the global approach were compared to those which were optimised locally. The results of these optimisations showed how the global approach to optimisation yielded the best solutions. This increased performance was observed even with very small amounts of computational effort.

It was also shown that the relationship between project size and optimisation execution time was a linear one. This showed that to create a global model of a major project, and to subsequently optimise it, was a realistic goal. Further confirmation of this was obtained by observing the results of the second case study.

The second case study involved the modelling of a significant portion of the construction phase of a real project: a new car park at the Johannesburg International Airport. The model was prepared using the same steps as for the first set of problems. A number of additional analytical techniques were identified in order to model the project properly. These were incorporated into the model, and added to the computer program.

For this project two important measures of project success were identified: cost and time. Therefore, two optimisations of the project were performed. One minimised cost, and one minimised time. The results of these optimisations were compared to the results of a line-of-balance (local) optimisation. As with the first case study, the global approach obtained better solutions. Indeed, the superiority of the global approach in this case study was even more marked than in the first case study, particularly when minimising time.

It was also necessary to address the fact that a solution which is minimum in terms of time is not necessarily acceptable in terms of cost, and vice-versa. Indeed, analytical techniques have, as has already been mentioned, received criticism for focusing on only one measure of project success. Thus, it was shown how the global model could be used to determine the trade-off which exists between different measures of project success. The trade-off between cost and time was evaluated, and it was shown how the project planner could select a solution which was optimal in the sense of the global approach. That is to say, the best solution in terms of both time and cost could be selected.

The development of a global approach to project optimisation represents a significant improvement in the technology used in the analysis of projects. The model is applicable to any project, and yet sufficiently flexible to be an accurate model of any

particular project to which it is applied. It is also capable of obtaining better solutions than existing techniques, as is demonstrated by its application to the case studies.

The potential benefits of this model are huge. However, the implementation of the global model for a major project is a significant undertaking. The numbers of analytical techniques which must be integrated into such a model are much more than were required for these case studies. Therefore, in order to ensure that the integration of analytical techniques may be done properly, further research will be required. The likely major components of this future research are identified both from what has been said in the literature, and by careful analysis of the results of the case studies.

Chapter 2 Literature Survey

Construction project management of one sort or another has been practised since man first made buildings at the dawn of civilisation. The oldest of the seven wonders of the ancient world, the great pyramid at Giza, Egypt, is an enormous and intricate building which was built around four and a half thousand years ago. In order to build such a structure an intricate design would have been laid out, and then the many thousands of men who executed the project would have been organised in some way to ensure that the final building was constructed in accordance with this design. Although the term had not been coined then, the ancient Egyptians were practising construction project management. Throughout the many centuries since, all the great civilisations of the world have planned and organised the construction of buildings. As technology has increased the field of construction projects has increased to include things such as dams, roads, aqueducts, canals, railways and factories to name but a few. All of these projects have needed to be planned and organised properly in order to be successful. Construction project management is therefore a fundamental element of civilisation.

The purpose of project management is to manage the people and resources (which include time and money) in the best way possible throughout the project. However, until recently there has been little mention by historians of how the execution of projects was planned and managed. Much attention is devoted to aesthetics or new technology, and the architect or engineer is usually commended accordingly, but little is mentioned of the management of the men and resources used to build them.

It is only really this century that project management has become a field of study. This has been brought about by social, political and economic uncertainty, rapidly increasing technology, project size and complexity, as well as an explosion of the subjects covered by scientific research. Research into project management has observed the behaviour of projects in an attempt to identify how, generally, projects should best be planned, organised and executed. It has shown how techniques can be employed to improve project performance by, for example choosing the best contractual arrangements, improving the efficiency of the work force or reducing

errors in design. Many aspects of the project can now be explicitly quantified thanks to the vast array of analytical techniques which have appeared in recent years. While this study has been able to improve the performance of construction projects, this has only served to increase the clients' expectations. In this highly competitive industry it is demanded that construction projects be completed more quickly, to a higher standard and for less money.

Because of the large size and complexity of today's projects, the process of managing a project involves much careful planning and monitoring, as well as the management skill required to make sure that human and physical resources are used as best they can be. Techniques which have been developed to facilitate the management of projects fit into two main categories: analytical techniques and non-analytical techniques. Analytical techniques are employed to determine and control measurable aspects of the project's execution. Non-analytical techniques focus on the non-measurable aspects of the project, such as relationships and the perceptions and priorities of the parties involved in the project.

2.1 Current Practice in Project Management

The central purpose of project management is to manage the people and resources (which include time and money) in the best way possible throughout the project. However, this "best possible way" can be very difficult to define. There are usually a number of different parties involved in a project (client, consultant, contractor, etc.), who tend to have different aims[Becassi, 1996; Flanagan, 1989; Kerzner, 1995]. The contractor and subcontractors are usually concerned with completing the construction in the most profitable way, and the management and clients are typically concerned with project's overall success across the whole project life cycle. This overall project success includes maximising, monitoring and verifying life cycle performance[Frame, 1994; Hand, 1997; Holmes, 1989]. The best possible life cycle performance can be desired to maximise profitability, or can be an end in itself.

Because an approach which considers the whole project is so important to the client, Life Cycle Costing (LCC) has become increasingly popular[Flanagan, 1989]. This

method of analytical planing involves considering the long term implications of all design decisions, costing their effect over the whole implementation life of the project. This is more beneficial for the client than basing all design decisions solely on whether or not it meets a set of design criteria.

Developing on the practice of LCC, Holmes[Holmes, 1989] proposed the means for a maintenance policy/strategy to be implemented. It was argued that a good maintenance strategy was more important to project performance than straightforward life cycle costing (LCC), because changes in technology and specification often invalidate the information used in the original LCC analysis. LCC was therefore recommended only for use as a general prediction. A good maintenance strategy would ensure that maintaining the building could be performed smoothly and efficiently. This strategy must be applied to the whole project life cycle.

2.1.1 Developments in Relationships Between Parties Involved

As well as an increase of a life cycle approach to the analytical side of project management, there has been a change in the way the different parties involved in a project, particularly the major parties, relate to each other. This has been both promoted and reported on in recent literature, outlining potential and actual success of contractual arrangements which allow all parties involved to align their goals within the project.

Firstly, in the area of safety, a report by the European Construction Institute[European Construction Institute, 1992] promoted working towards SHE (Safety, Health and Environment). This highlighted the need for SHE objectives to be considered throughout the whole project life cycle. Safety health and environmental issues should be evaluated and designed for and incorporated into contract documentation. It was also maintained that other contractual arrangements should promote SHE, rather than being detrimental to SHE objectives. This promotes the joint responsibility of all parties involved in the project to ensure safety throughout the project life cycle.

In the oil industry the sharing of responsibilities has gone even further in some projects. Because of the nature of these projects, the overall outcome can be very easily measured in terms of investment and reward. Recently this has lead to situations where the risk and rewards of the project are shared in some way between the client and the contractor. These "Alliance" type projects were formed to help resolve conflicts between the aims of the client and contractor by making sure that it is in both their interests for the whole project to be a success. This allows focusing on the relationships between the involved parties, rather than on the transacted goods and services, and encourages the contractor to be involved in the whole project life cycle. Such contracts have been tried in the North Sea[Hemmens, 1997], and also elsewhere[Campbell, 1997].

Hemmens [Hemmens, 1997] reported on the Britannia project in the North Sea, where an alliance concept had been used to form a new company (Britannia Operator Ltd.) from two holding companies. These two holding companies were to share the construction and operation costs, as well as sharing the profits made by the venture. This also allowed production and project optimisation to be linked to the aims of the final project outcome.

Campbell et al.[Campbell, 1997] reported a project in Australia where the alliance approach allowed project execution to go ahead despite considerable time constraints. The alliance necessitated a commitment to open communication, win-win situations and fast conflict resolution. More of the risk/reward was taken by the contractor. The presence of the contractor early in the project allowed options for the project to be studied in parallel. This was facilitated by considering the project as having a fixed start-up date, which reduced the time spent in the project feasibility phase.

2.2 Analytical Methods in Project Management

While getting the relationships between the parties involved in the project cam make a significant impact on the success or otherwise of a project, it is also necessary to use analytical techniques. Modern projects are highly complex, involving huge

numbers of activities, many of which operate in parallel and which can sometimes be difficult to understand in themselves. This is far too large for the project planner to comprehend unaided, so analytical methods are employed to help quantify the information.

Analytical methods can broadly be defined as quantitative methods which facilitate the planning of how the project will be executed, and the controlling of it in such a way that its execution is as close to that plan as possible. The process of planning has received much attention in the literature, particularly methods of analytical planning, which aim to determine critical values for certain aspects of the project (e.g. how long they will take and how much they will cost). Similarly, the analytical techniques employed to monitor a project's progress against its planned progress, and the means by which this progress can be controlled, have been studied in some detail.

There are many different types of analytical planning and control techniques available to project managers. These have been developed for a variety of different types of project and for a variety of different aspects of projects. These broad categories will be discussed in this section. This discussion will include some mention of the criticisms of some of these methods which have been raised in the literature.

2.2.1 Project Planning, Control and Monitoring

Although the practices of planning, controlling and monitoring are separate in terms of what they aim to achieve and when in the project they are implemented, they tend to use very similar analytical techniques. The broad areas/types of technique are often common to all three. These broad fields of technique will be discussed in detail.

2.2.1.1 CPM/PERT And Subsequent Scheduling Models

CPM (Critical Path Method) and PERT (Programme Evaluation and Review Technique) were developed by the American Navy, in co-operation with the management consultants Booz, Allen & Hamilton, in order to accelerate the Polaris missile programme[Stires, 1962]. PERT operates by breaking the project down into

distinct activities. It then uses a simple logic to determine the earliest and latest times activities involved in the project can start and finish. The implementation of this method was the single biggest contributor to the Polaris missile's becoming operational three years ahead of schedule.

There are many limitations to CPM and PERT as sufficiently realistic models of projects[Kazanji, 1991; Madhaviji, 1991]. The most frequent criticism is that resource handling contains many unrealistic assumptions[Kavanagh, 1985; AbouRizk, 1997; Arora, 1989]. PERT was designed for a project where there were very few resource constraints, and has subsequently been applied to projects with considerable resource constraints. When resources are constrained, the traditional network logic, including the notation of float, becomes invalid[Marshall, 1996]. For this reason a new type of scheduling was developed: Resource Constrained Project Scheduling (RCPS). This the addition of resource constraints increased the complexity of the problem considerably, and it was no longer possible to use a fast, simple algorithm to minimise project duration. Therefore various mathematical models have been applied. These will be discussed in the section on optimisation

Despite these criticisms, PERT and CPM are still very widely used, as they are simple both to implement and to use. Because of this much attention has been given to techniques which can enhance or build on these traditional (non-optimising) techniques.

Many people have been critical of the preoccupation with time of both the mathematical techniques and those RCPS techniques which are built on the traditional model. Arora[Arora, 1989] was critical of the fact that they all tended to consider activity durations as fixed and failed to consider things like crashing activities. This was expanded upon by Kavanagh[Kavanagh, 1985], who was also critical of PERT's focus only on the time and technological restraints on the project. He proposed a new model of construction which he called SIREN. SIREN increased the detail of project modelling beyond PERT/CPM by considering the project as a queuing system. It allowed several types of data which are beyond the scope of

PERT to consider to be implemented within the schedule. This included data from both the activity and project levels.

Kavanagh's criticism of the relationships used in PERT were expanded upon by Leachman et al.[Leachman, 1993]. They, however, focused on the current use of overlap relationships. Overlap relationships exist where only a certain amount of an activity needs to be complete before the following activity may commence. They were critical of earlier techniques which attempted to model this by applying a minimum fixed time period or lag between the starting of the activity and its dependent. However, this model is only valid if activity durations are not variable. If activities are measured in terms of resource application rather than elapsed duration then modelling will be more accurate. This method was used a variable resource model, which used a network flow model.

Miskawi[Miskawi, 1993], in developing a model which could include variable durations, tried to move away from the strict logic of the traditional CPM network. The emergent scheduling methods were, it was argued, too computationally demanding. Therefore a method for reducing the overhead required for calculating the schedule was proposed, based on vectors. This vectorisation was achieved by using the calendars outputted from networking models to create a schedule of resource use. Previously these calculations would have been done on a spreadsheet, based on interpolated values from previous projects, and would not have been able to calculate the effects on the schedule of project extension/compression. The vectorisation of this schedule allowed these effects to be re-evaluated with little computational effort, and was performed using Lagrangian multipliers.

While these techniques all constitute a valid extension of the project model, Raz et al.[Raz, 1996] were critical of the way some of these queuing project models were implemented with CPM concepts which they made redundant. Often, a value for float would be given, based on the float calculated purely by network logic, and without any consideration of resources. In remedy to this, they proposed performing a backward pass of the scheduling logic, similar to the backward pass employed by

traditional networking techniques, in order to obtain a more realistic value for the float of an activity.

Criticism of the preoccupation of CPM with identifying the critical path also arose from considerations of risk. Soroush[Soroush, 1994] asserted that due to the uncertain nature of activity duration estimation, there are many paths in a project which may become critical. Instead, it was proposed to find the most critical path, which is the path most likely to be critical. Monte Carlo Simulation and Dynamic Programming were rejected because they are too demanding and time consuming., and a heuristic approach was used.

Jaafari[Jaafari, 1996] went even further in considering the application of risk within projects. It was maintained that, rather than having to be carefully redefined, the term criticality had actually become a redundant for RCPS and risk analysis. As with Kavanagh's work the project was modelled as a queuing process, and this was expanded upon by considering risk elements in the project. This meant that different ways of structuring the project and allocating resources could not be considered. It was also argued that current techniques tried to impose precedence constraints on the project which reflect the strategy for dealing with resource constraints rather than simply including technological restraints. In order to consider these factors within the project, Time And Priority Allocation Scheduling (TAPAS) technique was developed. This model rearranged activities into parallel chains using only technological constraints. Fixed Activity distributions with a corresponding fixed value of resource use were implemented.

While much of the research mentioned so far was explicitly aimed at individual projects, a number of researchers began to look at the multi-project environment, where the planners are concerned with optimally distributing finite resources between a number of projects which are all active at the same time. A multi project environment can exist where a very large project is subdivided into subprojects, or where a contractor has several projects underway at the same time. Yang et al.[Yang, 1993] considered the application of the modelling of resources within project scheduling to the multi-project environment. A method of efficiently scheduling

many projects within a constrained resource environment was proposed. This type of problem is characterised by a limit on the total resource use by all projects. The algorithm proposed projects resource demands of all projects in the immediate future, and attempts to distribute available resources among projects according to need.

2.2.1.2 Activity Duration Estimation

The estimation of activity duration is an essential component of obtaining an accurate measure of project duration. Despite its importance, however, it has received little systematic attention[Hendrickson, 1987]. The main aspects of deterministic activity duration estimation will be covered in this section. Non-deterministic factors, those factors which have a random element to them and are outside the control of the planners, will be discussed in section 2.2.1.3.1.

The duration of an activity may be determined by a number of different factors, including allocation of resources, crew output rate[Moselhi, 1993 (2)], learning curves[Moselhi, 1993 (2); Kavanagh, 1985], activity cost (the amount of money the planners are willing to invest in an activity[Phillips, 1996; Phillips, 1977; Kavanagh] and the availability of space [Thabet, 1994].

Rather than considering resource allocations as a linear scale, Talbot[Talbot, 1982], argued that it was more realistic to consider a number of number of "resource modes" by which an activity may be undertaken. Each mode defines a possible way of assigning resources to an activity, along with the duration associated with the execution of the activity in that mode. This also allowed the situation where an activity may be executed by two different types of crew to be considered.

Hendrickson et al.[Hendrickson, 1987] proposed an expert system for the estimation of activity durations. The paper discussed all those major deterministic influences on activity duration which had been identified in the literature. These were all incorporated in an expert system which started at a low level estimate, based on the manpower allocation, and then adjusted it for higher level considerations, such as work conditions, learning curves, etc. This allowed an accurate estimate of activity duration to be obtained for the deterministic case. Risk was not considered in any

way, and no attempt to analyse the effects of the values of certain input variables on the variability of the activity's duration was made.

Thabet[Thabet, 1994] was concerned that the current techniques for activity duration estimation failed to address the activity's need for space on a construction site. It was explained that space is often at a premium on construction sites, and that that needed to be effectively modelled. The availability of space has an effect on the productivity of crews within an activity. In previous literature productivity has tended to be estimated from historical data[Hendrickson, 1987], and have ignored that fact that lack of work space is detrimental to activity progress. This is because space is required for operating activities and storage of materials and equipment. When this is not available in sufficient supply, the activity cannot operate as efficiently. In order to address this space was modelled as an ordinary resource within a scheduling model. This permitted the scheduling logic to consider the availability of space and adjust the productivity, and hence the effect of space on activity duration.

2.2.1.3 Forecasting

Project planning - both in the design and various execution stages of the project - is not only concerned with finding out what will happen when and what resources will it require. It is also concerned with finding a forecast of the projects outcomes: how much it will cost, how long it will take, how much revenue it will earn, etc. A reliable forecast of project completion is essential for a number of reasons. Firstly risk in a project is high where insufficient analytical effort has been devoted to creating an accurate financial plan[Isahara, 1992]. Good forecasting allows the early identification of project delay and hence allows more time for recovery. It also allows the trade-off between the cost of project compression and the cost of project slippage to be evaluated[Ahuja, 1985]. There are two types of forecasting. Firstly there are those methods which attempt to evaluate project outcome before the construction phase has commenced, and tend to be purely analytical. Secondly, some methods attempt to evaluate the final project outcome based on the difference between the actual performance to date and that which had been expected.

2.2.1.3.1 Purely Analytical Techniques

Perhaps the major factor in the reliability of analytical forecasts of project performance is risk. Many assumptions have to be made about the project's requirements and the environment in which it will operate.

Ahuja et al.[Ahuja, 1985] identified seven common and quantifiable sources of risk which would detrimentally affect the project by increasing the project's demand for either resources or time. Consideration of these sources of risk showed that values within the project network which might previously been considered as deterministic should be considered as probability distributions. The use of PERT to evaluate the impact of these variables was considered to be too simplistic. Therefore a Monte Carlo Simulation (MCS) is used.

Woolery et al.[Woolery, 1983] were critical of methods which considered risk only at an activity level, and ignored the fact that an activity's not performing as planned may be due to both activity level and project level risks. These project level risks were typically expressed as "modifiers" to the activity's inherent probability distribution, which allowed project level risks to be coordinated throughout the activities. These project level risks could then also be linked to time (seasonal risks).

Diaz et al.[Diaz, 1993] reviewed the use of five nondeterministic methods: Program evaluation review technique(PERT); probabilistic network evaluation technique (PNET); narrow reliability bounds (NRB); Monte Carlo simulation (MCS); simplified Monte Carlo simulation (SMCS). PNET and NRB require all paths for calculation and are therefore require more space for calculation, making them a memory intensive technique. MCS requires around 10,000 calculations to calculate, making it a time intensive technique. PERT was found to be the most liberal estimation method, and the MCS and SMCS the most conservative for duration estimation.

Cox[Cox, 1995] saw the use of stochastic methods to be too demanding for regular use. A statistical approach is used. It iterates through all nodes in an activity on

arrow network, calculating the mean and standard deviation for each, to arrive at an approximate solution for the mean and standard deviation for project duration.

AbouRizk et al.[AbouRizk, 1997] also found formal statistical methods to be very demanding. They used a stochastic simulation method to link CPM, uncertainty and productivity. A neural network was then used to generate a forecast of project duration, which reduced the computational intensity of the problem.

Rather than using a quantitative probabilistic analysis of risk within projects, [Yeo, 1991] suggested that sensitivity analysis should be used. The use of traditional spider plots, which involves plotting the variation of uncertain variables against the corresponding variation in project outcome, was found to be limiting as it failed to take combinatorial effects into account. It was suggested to use the most optimistic, most pessimistic and mean values for every uncertain cost variable. This gives a better idea of how the uncertainties affect the project as a whole, and indicates which variables have the most impact on the project's outcome.

Moving away from low-level technical considerations of risk, Isahara[Isahara, 1992] identified a number of major risks which effect projects, particularly capital projects. These risks were divided into three main categories: pre-completion risks (those risks which affect the design and construction of the project), operational risks (those risks which affect the operation of the project) and ongoing risks (those risks which affect the project as a whole). These risks played a very large role in the uncertainties surrounding the project, much more so than the low level, technical, risks. It was argued that many of these uncertainties were not necessarily due to inherent uncertainties, but down to insufficient analytical effort. Thus the level of uncertainty could be reduced considerably by better analysis. As well as providing a more accurate forecast of final project outcome, better analysis would also allow the effects of unlikely events to be evaluated and taken into account while preparing or updating the financial plan.

2.2.1.3.2 Performance Based Forecasting

After a project has entered the construction phase it is possible to compare the current performance of the project to the expected performance. Based on this comparison, it is possible to project the final outcome of the project by assuming that the level of performance relative to that planned remains constant. The most common method of forecasting involves comparing how much work has been done and how much it has cost with how much was expected to have been done and how much it was expected to cost. This information is then used to estimate the final cost and completion date[Humphreys, 1991].

Teicholz[Teicholz, 1994] aimed to improve upon this relatively simplistic approach. Characteristics of an ideal forecasting method were identified. Firstly it should not require any data which is either difficult or expensive to collect. It should be simple enough to be incorporated into the cost system. The analytical technique should be accurate and unbiased, providing information in a way which is up to date on current project performance, without being unstable. A linear projection of sliding moving average cost is recommended. This technique uses the most recent project performance as the assumed value of future performance and allows any possible slip in project performance to be identified in time. However, it is recognised that such a method should not confine itself to looking so recently that it fails to provide stable forecasts.

2.2.2 Process Control and Monitoring

The analytical techniques applied to projects are generally specifically designed for projects. However, there are many ways in which a project can be viewed as a process which transforms something into a new state[Knoepfel, 1989].

This is demonstrated by Hackman et al.[Hackman, 1989]. They proposed a general model which would allow the modelling of any production system. They identified all the elements common to a process of any kind, and created a framework which could be tailored to fit specific problems. The applicability of this general

framework is shown by a number of examples, including the modelling of a CPM/RCPS problem.

The scheduling of repetitive processes is often not considered in projects, which ignores the repetitive nature of many aspects of a project. When a series of activities of a similar nature exist, a crew of men usually moves from one of these activities to the next, and all the resources are required at a constant rate. There are also other considerations, such as learning curves, which dictate that a repeatable activity will be performed more slowly to begin with, until the crew learn how to perform the activity more efficiently[Humphreys, 1991; Kerzner, 1995].

Arditti[Arditti, 1986] proposed the use of Line Of Balance (LOB) methods to schedule repetitive processes within a multi-project environment. This was employed in order to ensure that the programmed rate of completed units could be met. This in turn would allow a constant rate of repetitive work, a balance of the labour force and the cost benefits of repetitive work to be maintained. It was argued that a traditional network analysis which included resources and was performed only at the process level, rather than for the whole project, would fail to maintain a balance of the labour force. The use of LOB was a much less demanding method of analysing the whole project than resource constrained project scheduling.

The traditional implementation of line of balance methods, however, has been criticised because all its repetitive units to be identical in size and scope, which is a very uncommon scenario[Moselhi, 1993 (2)]. Even where such repeatable units exist, there is usually a learning curve, which means that activities early in the process will be executed more slowly than activities later in the process. This applies equally within a project and across a multi-project environment. LOB also has no way of incorporating any non-repeatable aspects of the project into it's modelling of project resources. Thus, any conflict between those activities which lie outwith the repeatable blocks and those activities within the blocks will not be considered. These factors limit its scope, and so it has received little attention in the literature in comparison to other techniques.

2.3 Optimisation

Optimisation generally involves seeking the best possible solution to a problem. In a mathematical sense this involves finding the maximum or minimum value of a function. There are many approaches to finding this value, for many different types of functions. Many techniques are well documented in ordinary textbooks[Adby, 1982; Russell, 1970; Humphreys, 1991]. However, many other techniques have been developed to cope with situations where traditional techniques were found to be inappropriate.

Traditional approaches to optimisation have tended to focus on finding the best value possible, the true mathematical optimum. This kind of optimisation relied on very specific mathematical definitions of the problem being modelled, and was often required to make many assumptions. They also required very specific methods of solution, which could often be very demanding. Nelder et al.[Nelder, 1965] were one of the first authors to accept that traditional approaches would be too demanding for use in certain situations. They proposed a method of optimising linear programming (LP) models which converged on the optimum, but did not find it. However, it was able to obtain a value which was very close to the optimum. Its advantage was that it was not as computationally demanding as exact methods, as it did not involve the solution of many large matrices. Nevertheless, the implementation still required assumptions to be made in order to generate the LP model in the first place. Further refinements to these mathematical models would always be constrained to having to formulate models in this very specific way.

Because these Mathematical techniques were so specific, a number of new methods of solution arose. Perhaps the most significant techniques were those which were able to search randomly through the possible combinations looking for good solutions. The most basic form of this optimisation technique is the Monte Carlo method, in which a large number of randomised solutions to the problem are evaluated, and the best selected. While these methods would not be expected to find the true mathematical optimum, they would be able to find solutions which were close enough to the optimum to make their use justifiable. However, it was also the

case that, ini many situations, such an unbiased random search could tend to require a large number of function evaluations to find a sufficiently good solution.

Because of this, methods were developed which would restrict their search to areas where good solutions would be likely to come from. Perhaps the simplest extension of the Monte Carlo model was proposed by Conley[Conley, 1988], who reported on the use of a multistage Monte Carlo method. This uses a ranking system to preference certain selections in the random selection process in the optimisation. Its applicability was demonstrated on a shortest route problem.

Lee et al.[Lee, 1996] gave an overview of three near optimal solution methods which were used to solve Resource Constrained Project Scheduling problems (RCPS). Three methods were described: Simulated Annealing, Tabu Search, and Genetic Algorithms (GAs). While all these optimisation techniques had been proposed in the past, this is the first time they were used in the field of project scheduling.

Simulated Annealing makes changes to an existing solution. If that solution represents an improvement in the objective function, then it is accepted and becomes the new existing function. Otherwise the changed solution has a finite probability of being accepted. This probability is influenced by how well progressed the optimisation is (probability decreases over time), and the difference between the new value of the objective function and the existing (probability increases with proximity).

Tabu Search makes small changes to an existing solution, to create several new ones, selects the best, and moves to this new solution, irrespective of whether or not it is better than the existing. A list of the most recent solutions used is maintained, and the search is not permitted to return to these values. Other constraints are also applied to enhance the optimisation.

Genetic Algorithms are perhaps the most widely used of these techniques. Their application to many different types of problem is well documented. The method involves maintaining a population of solutions which "evolves" from one "generation" to the next. At each generation a number of parent pairs of solution are

selected. Then the solutions are crossed over in each pair. This involves generating two new solutions which are identical to the parents except that some of the values in the solutions have been swapped over. After this some values of the new solutions may be changed slightly, or mutated. These new solutions then replace existing solutions. The exact behavior of the crossing over, mutation and selection of parents and replacements can all take many forms[Chambers, 1995].

A further criticism of traditional optimisation techniques was raised by Stuckman et al.[Stuckman, 1990]. They argued that traditional models focussed too much on idealised representations of the problem, whose optimal solutions did not represent solutions which were feasible in reality. They sought to bridge the gap between the solution which a function/simulation based optimisation may arrive at, and the best real solution to the problem. Applying this principle to design, function/simulation based modelling was complemented by real experimental models for design optimisation.

2.3.1 Optimisation of Projects

Optimisation has been applied to projects in many different ways. Indeed, one of the first analytical techniques to be applied to projects, the PERT/CPM model, is an optimisation technique: it yeilds the minimum time to complete a project with precedence constraints. However, its assumption that resources were unlimited, while realistic for the problem for which the technique was originally designed, was found to be unrealistic when the problem was applied to other problems. This gave rise to he field of Resource Constrained Project Scheduling (RCPS).

2.3.1.1 Resource Constrained Project Scheduling

Within RCPS there are three resource types: renewable, non-renewable and doubly constrained[Boctor, 1993; Slowinski, 1981]. Renewable resources such as manpower are available in a fixed quantity per time period at the end of that time period they will be available for use again in the next time period. Non-renewable resources are resources which may only be used once. A project will usually have a stock of these resources, and using the resource will reduce the size of the stock

accordingly. This applies a limit to how much of this resource may be used over the whole project. Doubly constrained resources are essentially a class of non-renewable resources. However, as well as having limit on the total amount of the resource available, there is also a limit on the amount of resource which may be used per time period.

2.3.1.1.1 Exact Approaches

The solution of the precedence network without resource constraints was relatively easy to calculate exactly. When RCPS was first studied, the aim was to find an exact approach which involved minimal computational effort. The earliest solution methods were based on either linear programming or integer programming models[Patterson, 1974]. However, unlike the PERT/CPM model, the RCPS solution techniques tended to be far more computationally demanding, so many solution techniques were investigated with the aim of reducing these computational demands.

Davis et al.[Davis, 1971] attempted to solve the problem by breaking the project into a series of one period activities. This allowed the formulation of the problem as a shortest route problem, or A-Network. This type of method allows nonconstant job requirements (when the resources required by an activity are not evenly distributed throughout the duration) and supports preemptive scheduling. This is where activities may be temporarily halted to allow another, more critical activity to be scheduled, and is common practice in the construction industry.

Schrage[Schrage, 1972] also developed a method for solving the RCPS problem for the preemptive case. As with Davis et al., each activity was broken down into jobs with a duration of one time period each, and the resulting network was solved as a RCPS problem using emplicit enumeration. However, it was found that permitting preemption did not typically yield substantially lower project durations.

Patterson et al.[Patterson, 1974] used a zero-one integer programming model. This involved generating relationships between all activities, which dictated which should be schedule first. This set of activity relationships was then optimised to find the

shortest duration using an integer programming model. Its performance was found to be comparable to that of other contemporary methods.

Talbot et al.[Talbot, 1978] realised that the efficiency of many techniques which had been designed for the RCPS tended to use a lot of computational resources eliminating solutions in areas where no optimal solution could be found. They used network cuts to allow the removal large groups of solutions which could not contain a solution which is better than one already identified as part of the optimisation. This allowed the efficiency of an existing algorithm to be improved considerably.

Patterson[Patterson, 1984] later compared this technique with two other techniques which had been developed at around the same time. These two techniques also allowed the elimination of large portions of the complete set of solutions. In order to evaluate the three methods a large set of 110 different RCPS problems, which had been used in various literature in the past, were assembled. While branch and bound (Stinson) was found to be the quickest, the three methods were all of comparable speed.

Although interest in the exact solution of RCPS problems was high, and much research was published in the field, it failed to achieve widespread use in real projects. This was partly because of its many assumptions, but mainly because of its high computational demands, particularly for projects with a large number of activities.

To this end, Simpson et al.[Simpson, 1996] sought to improve the performance of some of the exact approaches by implementing them for parallel computing. Modification of Talbot's implicit enumeration algorithm was performed to make it suitable for parallelisation, and it was also proposed to use a heuristic solution as a good starting point for the search procedure in order to speed the process up still further.

While most of the research into RCPS has been concerned with minimising time, Robinson[Robinson, 1975] was concerned with the cost-time tradeoff in resource

constrained projects. By dividing the project into paths of activities, and applying a dynamic programming algorithm, time could be minimised for a fixed cost.

Slowinski[Slowinski, 1981] also argued that RCPS models should consider cost as well as time. A method of solution was devised for the pre-emptive case which included cost considerations. However, rather than reviewing a trade-off between cost and time, the method proposed used LP to solve the problem both for minimising time and minimising cost.

Expanding of the theme of extending exact solution of RCPS problems beyond the simple minimise time objective, Talbot[Talbot, 1982] suggested a model which was more flexible in allowing optimisation to be performed on the most important management objective. A general scheduling model was presented, with activities having several resource modes. This model could then be defined in terms of a single project outcome. It was specifically applied to minimising completion time and minimising overall cost, formulated as an IP model and optimised using emplicit enumeration. The advantage of this model was that it could be used to find either optimal solutions or, with reduced computational effort, near optimal solutions.

Easa[Easa, 1992] attempted to add an increase in realism in modelling cost in RCPS. Rather than using a simple calculation of cost, NPV was used. This was maximised using a linear programming model. It assumes that cost is accumulated directly through activity execution, and that payment is made according to work completion. The model was also capable of minimising overdraft change.

2.3.1.1.2 Heuristic Approaches to Minimising Time

Once research into RCPS had intensified it became clear that exact optimal approaches would be too complex to use in computational packages[Arora, 1989]. The computational demands of using such methods in a project with a reasonably large numbers of activities would simply be too great. A simpler algorithm was required. For this reason many attempts have been made to produce efficient heuristic algorithms which allow near optimal solutions to be arrived at with minimum computational effort.

One of the earliest heuristic techniques proposed was by Weist[Weist, 1967]. The minimum slack heuristic (MINSLK) was used, where activities with the least float are prioritised over activities with a higher float. When selecting activities to be scheduled those of a higher priority will be scheduled ahead of those of a lower priority whenever there are insufficient resources to schedule both. Critical activities were scheduled at max resource allocation (minimum duration) if resources were available. If critical activities could not be scheduled then the scheduling logic would attempt to borrow resources from other activities to try to schedule the activity, in order to reduce time slippage.

Davis et al.[Davis, 1975] compared eight heuristic rules on 57 non-preemptive resource constrained networks with known optimal durations. These rules would all dictate that certain activities would have a priority over others. If the scheduling logic found a situation where two or more activities were competing for limited resources, then the one with highest priority would be scheduled first, and then the next highest, and so on until either all activities are scheduled or no more activities may be scheduled without violating resource constraints. MINSLK was found to be the most efficient method of those tested.

Using the same model of using rules to dictate an order of preference in scheduling activities, Khattab et al.[Khattab, 1991] expanded the work done in identifying such good scheduling heuristic rules. It was proposed to use eight rules which had been identified as efficient, and incorporate them all into one single program. This program scheduled the project eight times, using each rule separately. It then selected the best of the eight schedules generated. This proved more efficient than using one single rule.

Many good heuristic rules have become available which help a scheduler prioritise activities. However, Moselhi et al.[Moselhi, 1993 (1)] identified a way of making use of the fact that there were a large number of ways of scheduling a group of activities. Rather than considering activities individually, they suggested a means of prioritising combinations of activities rather than individual activities. At any point, or time frame, there will be a number of activities which are ready to be scheduled.

If not all activities may be scheduled then sets of activities are identified. These are sets which may be scheduled without violating resource constraints. The scheduling of any one of these sets will cause some activities to be delayed. This has a detrimental impact on the project. Therefore the set with minimum impact on the project is implemented for that time frame, and all those activities are scheduled. This method was able to consistently outperform other heuristic rules.

While the method of Moselhi et al. for increasing the efficiency of heuristic rules was effective, Boctor[Boctor, 1993] expanded on the use of heuristic rules by allowing the activities' duration to be non-constant. Twenty one resource scheduling rules were identified for a constrained scheduling model where activities were given several possible modes of execution, each with different durations and resource requirements. This allowed an activity to be scheduled in a different mode in situations where other techniques would not be able to schedule the activity, and hence minimise project execution.

Elmaghraby[Elmaghraby, 1993] also considered the possibility of varying the activity's duration. Dynamic programming was used to reduce project duration. This was done by considering two types of variable: one allowing extra resources to be added to activities on the critical path, and one allowing the optimal allocation of resources to parallel activities.

Lee et al.[Lee, 1996] decided that the use of such simple heuristic rules, while it allowed good solutions to be found quickly, ignored much of the scope for optimisation which was available within the problem. Instead of a simple rule, they made use of search heuristics. While these techniques would be more computationally demanding than the simple rules, they would be expected to find better solutions without requiring the level of computational effort associated with the exact approaches. They used an arbitrary priority number to determine which activities are prioritised for scheduling in a resource constrained situation. To find the optimal values of these priority numbers, three heuristic search techniques were used: Simulated Annealing, Tabu Search, and Genetic Algorithms (GAs). For the Genetic algorithm a weighted random parent selection method was used, both for

selection and replacement. This method makes better solutions more likely to be selected as parents, but less likely to be selected for replacement.

While Lee et al.[Lee, 1996] were concerned with using search heuristics rather than rules, Kurihara et al.[Kurihara, 1985] moved away from PERT based techniques altogether. They used NLP to optimise the project, using a Graphical Evaluation Review Technique (GERT) network. This was employed to minimise the execution time of a project of uncertain job order and job execution time. This allowed a near optimal solution to be obtained while considering risk.

Yau et al.[Yau, 1990] observed that some of the assumptions made in the analysis of the resource time trade-off are quite simplistic, particularly when it comes to evaluating the cost of allocating extra resources. A method for project compression was proposed which aims to preserve the current project schedule, hence avoiding costly reschedules which might be obtained by other techniques. The reduction in time is achieved by making use of surplus resources and assigning them to activities which are holding the project up. Because no additional resources need be assigned, this method achieves project compression without any increase in cost.

2.3.1.1.3 Heuristic Approaches to Minimising Cost

The feasible schedules generated by time optimising RCPS algorithms are not necessarily cost optimal[Moselhi, 1993 (2)], and those implemented in practice do not generally consider things like varying activities' durations[Arora, 1989]. As with the exact solution, heuristic near-optimal solutions to RCPS models which included considerations of cost were investigated.

One of the first to recognise the link between time and cost in RCPS was Phillips[Phillips, 1977; Phillips, 1996], who proposed a means of reducing project duration at minimal cost. This method used a cut search method, which, using a network flow model in an AoA network, allowed reduction in duration to be acheived at minimal cost.

Igelmund et al.[Igelmund, 1983] proposed the use of what they called "preselective" strategies, which could be mathematically identified and implemented in a cost optimal heuristic solution. These strategies would aim to reduce the cost of the project in the same way heuristic rules had aimed to reduce time, but with the added benefit of considering risk as well.

Patterson et al.[Patterson, 1990] sought exact solutions to minimising cost on RCPS schedules. However, in order to reduce the computational effort required the schedules were evaluated using an existing heuristic rule. An emplicit enumeration model was used to minimise NPV on these heuristically evaluated schedules. This used a backtracking algorithm in the cost evaluation and used MINSLK for near optimal schedule times.

These methods for optimising costs required completely new solution techniques to be implemented. Despite being heuristic, near optimal solutions, they were still very specific and mathematical, and therefore were not widely implemented in practice. Rather than creating another such new technique, Arora[Arora, 1989] simply expanded on some of the simple models which had been proposed for minimising time by adding cost to them. In order to reduce the computational effort required to evaluate these models, it was shown how the model could be paralellised. To this end a distributed program with no shared variables was used. This involved breaking the project down into subprojects. Once the subprojects were defined, interface activities between subprojects could be identified. Activities were broken down into vectors, and the MINSLK rule was used for resource constrained scheduling.

Davis et al.[Davis, 1992] also attempted to expand the realism of existing techniques. It was argued that resource constraints should be relaxed rather than rigorous, because the ceiling can usually be exceeded, although this does cause an increase in cost. Therefore the trade-off between resource ceiling and duration was evaluated within their model.

Another factor to be considered within cost optimisation was introduced by Moselhi et al.[Moselhi, 1993 (2)], who sought to incorporate minimise cost by minimising

crew idle time and maximising learning curves. These effects had been considered before, but only implicitly by some of the methods which sought to minimise time. A model which took these factors into account was developed, and dynamic programming was used to solve it.

Akkan[Akkan, 1996] was also concerned with overtime scheduling. A method of using finite real time scheduling was developed. This allowed the scheduling of work orders without making substantial changes to the planned schedules of previously planned work orders. It is argued that to reschedule existing planned work orders would be unfeasibly expensive.

The development of a cost model by Li et al.[Li, 1993] arose from a concern about current methods which aimed to reduce project duration without being careful not to cause costly disruptions in the schedule. While an increase in the cost is almost inevitable if one aims to reduce the duration, the practice of creating a solution which requires a lot of schedule replanning can generate unnecessarily large increases in cost. Therefore the method suggested involved identifying the longest paths, and finding the most cost optimal way of reducing each of the paths in turn, starting with the longest.

Li[Li, 1996] was concerned that many of the cost optimising techniques which had been developed were too demanding for use on large scale project. Concern was also expressed that these techniques were often unable to consider the project in terms of real management objectives. A method was proposed which looked at large scale projects, and attempted to optimise a project in terms of several output criteria by finding optimal start times for all subprojects. This involved finding the relationships between the schedule of each subproject and: weather, resource supply and investment allocation. It uses Monte Carlo (risk) Simulation on an LP model.

2.3.1.1.4 Combined Objective Optimisation

The development of a model which tries to look at real management objectives by Li[Li, 1996] arose from one of a number of limits in the realism and applicability of contemporary optimising techniques. These weaknesses are common to almost all of

the many optimising methods which have been proposed for different aspects of projects. It has been recognised that not all project outcome will reduce to cost[Davis, 1992], and that there are many goals within a project which are important. Optimising with a single objective assumes that there is only one[Norbis, 1988], and is therefore restricted in its usefulness.

As well as not being able to consider real optimising objectives, the scope for optimisation is also quite restricted within many current techniques. Although the global optimum of a system often bears some relation to the optima of the submodels, providing they have been optimised in terms of the same variable as the output[Williams, 1993], optimising many component parts of a complex project system will not necessarily yield the optimal solution for the whole system[Guang-Yuan, 1992; M'silti, 1993; Stuckman, 1990]. Thus, current optimisation techniques are restricted because they only consider small portions of the whole project.

Norbis et al.[Norbis, 1988] addressed the general problem of single objective optimisation by using a multi-level heuristic to find feasible near optimal solutions for a multi-objective RCPS model. This allowed solutions which were compatible with a set of common management objectives to be generated.

M'silti et al.[M'silti, 1993] were concerned with the absence of any practical decision support method for optimising project performance in terms of more than one criterion. It used NLP to define a reference point, which represents a realistic ideal project performance, in terms of all the important output criteria. The optimisation sought to get as close as possible to this point. This, it was proposed, would form the core of an expert system. The fact that similar techniques often failed to solve real problems in industry was also raised. On this basis it was recommended that the technique proposed be developed in the field, rather than by research. This, it was hoped, would ensure its applicability to real industrial situations.

Guang-Yuan[Guang-Yuan, 1992] solved the multiple goal problem in a different way. Rather than getting as close to the ideal situation as possible, it was proposed to use a means of either optimising cost at a given level of quality, or maximising

quality at a given cost. This technique was aimed at setting realistic project goals pre-design.

Expanding on the concept of multiple objectives, Davis et al.[Davis, 1992] used a goal programming method to resolve the trade-off between resource overutilisation and project completion time. This is a more formal mathematical method than the two methods previously proposed, and has been used widely in other fields of optimisation.

2.4 Systems Theory

Much effort has been made in the field of project modelling and optimisation. The literature reviewed here represents only a small portion of the entire field of research in this area, and many different aspects of projects have been subject to this research. However, such techniques do not represent the only way of analysing a project. Another field of research relating to the management of projects has been receiving increased attention in recent years: the application of systems theory to projects. The principal problem with existing planning methods is that there are few techniques which have implications for the whole project[Walker, 1989]. Therefore the application of a systems approach within project management has been proposed so that planners are able look at the project in its entirety, rather than considering it piecemeal[Kerzner, 1995; Rodrigues, 1996].

Knoepfel[Knoepfel, 1989] advocated the use of a systemic approach to project management. This involved viewing a project as something which transformed a system to a new state. Quality is then the measure of how close the project comes to the desired new state, and the benefits of this new state depend on the nature of the project, although they are often difficult to quantify. In order to implement a systems approach fully, it was recognised that a project information system was required which would allow information to be collected from any level of the project system. This would allow the project as a whole to be analysed.

Buchanan[Buchanan, 1991] proposed how systems theory might be used to ensure that the management focus on aspects which are important to project success. He was critical of the current focus on analytical techniques, and argued that cost control, which is usually used as a method of reducing the likelihood of project overrun, may actually cause projects to overrun. He proposed a change of management focus away from this analytical technique towards what he describes as the process agenda. The project management team is expected to be: technically competent and experienced in the field of the project (content agenda); competent with techniques of planning, scheduling, monitoring, etc. (control agenda); competent in team building, communications and consultation, influencing and negotiating skills, management of enthusiasm and resistance (Process Agenda). It is by focusing more on this process agenda that the type of project overrun which a focus on analytical techniques can cause may be avoided.

This process agenda was discussed in the more specific field of risk analysis by Ward et al.[Ward, 1995]. Concern was expressed that risk analysis was not usually applied to the PLC. As part of looking at the whole project they proposed considering process risks. These are those risks inherent to the management process, rather than risks only inherent in the physical nature of the project. Using current analytical methods which tend to focus only on technological risk, these process risks will usually go unnoticed until they manifest themselves much later in the project, by which time they cannot be effectively managed. If these risks were to be identified early in the project, it was argued, then they could be much more effectively managed.

Rodrigues et al.[Rodrigues, 1996] applied system dynamics in a more general way by considering the management structure, with specific reference to the analytical techniques employed within it. They maintained that implementation of system dynamics to the project's management structure would create a more holistic approach than current, analytically based, methods. The system dynamics approach, while not providing the level of detail of traditional planning methods, would provide a flexible system which would facilitate the analysing of the whole project in a

practical, albeit low level, way. The application involved using methods which concentrated on human interaction, and considered the project as a whole rather than as small components. Building on this, some ways in which systems dynamics might be incorporated usefully into traditional analytical methods were also outlined. This demonstrated that the holistic, higher level systems approach could be used to increase the effectiveness of the lower level analytical planning.

The use of systems dynamics alongside analytical techniques may be very useful for helping to provide a more realistic and manageable measure of a planned project's performance. However, Becassi et al.[Becassi, 1996] were concerned that measuring a project only in terms of things like cost or time did not constitute an accurate measure of project performance. They provided a much less restrictive general scheme for assessing different success/failure factors for projects. This not only uses systems analysis within the project analysis, but adopts the systems approach for the evaluation of the project's outcome. It is often assumed that the success of a project depends upon good analytical planning. However, success is perceived differently by different parties involved in the project. Four main groups of factors were proposed which relate to: the project; the project manager and project team; the organisation; the external environment. It was shown that in a real project all these factors are interrelated and must be managed together to ensure project success.

2.4.1 Expanding The Field Of Analytical Techniques

The application of the field of systems analysis has been helpful in terms of identifying an overdependency on low level analysis and the need to consider the project as a whole both in terms of its quantitative aspects, which analytical techniques attempt to measure, and the qualitative aspects. Despite all this criticism of the appropriateness of analytical techniques, however, it still commands much attention in the literature. This is because criticism has not ever aimed to say that the project planners should not use analytical techniques at all. Rather, the criticism has been focussed more on the failure of analytical techniques to facilitate the planning process as much as they potentially could. Many of the problems raised by the application of systems theory are being addressed through some changes in approach

to the use of analytical techniques within projects. These new approaches aim to do away with the short-sightedness of those techniques which have been developed and used in isolation.

Arora[Arora, 1989] attempted to provide a framework for considering how contemporary analysis fits into such a holistic approach. Three stages of analytical planning were identified: time analysis (precedence networking), resource analysis (RCPS feasible schedules) and cost analysis (optimal RC Feasible Schedule). The purpose of these analyses is twofold. Firstly, it gives a schedule for the work[Kerzner, 1995; Lockyer, 1991], and secondly it provides information which allows decision makers to more informed decisions[Karkaria, 1987]. Moving from the first to the third of these levels of analytical modelling involves an increase in the level of detail considered in the model, to which the law of diminishing returns applies. This law states that to increase in accuracy or realism of the model involves an exponential increase in the level of detail of modelling required to obtain it[Mirham, 1972]. This increase in detail can also be expressed as an increase in effort required in order to obtain an optimal solution.

Stepniak[Stepniak, 1990 #78] also argued for a framework which allows the adoption of a holistic approach to analysis, and proposed an object oriented project management system to be used to create this. By applying object orientation, different portions of the model could be viewed as different objects. The objects would contain data which define the object, functions which describe the object's internal behaviour, and functions which define the interrelations between it and other objects. A method for assessing the objects which contained most scope for optimisation (which would allow them to be prioritised in any optimisation process) was also outlined.

Brown[Brown, 1992] was concerned about the wide use of classical probability theory which, even in the stochastic case, depended upon accurate measures of probability which are very difficult to obtain. Rather than using these traditional techniques which suggested that fuzzy sets might be used within analytical systems.

This would allow uncertainty to be measured without the need to make any quantitative assumptions about the data used.

All of these techniques recognise the need for new, holistic approaches to project analysis. However, Hallefjord et al.[Hallefjord, 1993] were concerned about the level of computational demands which would be required by the implementation of such techniques. They argued that projects are too large to be modelled all at once in one large, complex model. In order to solve this problem it was shown how subsystems of the project could be modelled individually and then aggregated by using an approximation to the modelled behaviour. Optimisation could then be performed at a the project level using the aggregated models of the subsystems with far less computational effort.

While the expansion of analytical techniques in this way can be useful in making better, more applicable groups of analytical techniques to be available, it does not necessarily follow that they will be used in practice. This can be seen by looking at how many of the developments in analytical techniques have actually been implemented in the industry. Despite the fact that there are many analytical techniques available to project managers, in practice they are often limited to what is commonly available within the software domain[Kennington, 1980; Lockyer, 1991].

Because most analytical techniques are difficult to perform without the use of a computer program, project planners are usually restricted to using those techniques which are available in the software domain. This software, however, usually contains only a restricted number of analytical techniques, rather than the wide range which might be expected. While this could be put down in part to the criticisms raised of such techniques, some of the blame should be assigned to the software industry for not allowing for the implementation of such techniques. In addition to this criticism, this type of software tends also not to allow sufficient flexibility to accurately model many specific projects[Pidd, 1989]. Humphreys summed up the situation as follows:

> *"...the industry has become almost preoccupied with sophisticated scheduling software packages for time planning, while very few are available or used to*

facilitate other types of planning. The strangest part of all is that these critical path based scheduling programs, which are predicted on strict, logical interrelationships of activated with fixed durations, are being applied to an industry characterized by extreme variabilities and uncertainties."[1]

Heindel et al.[Heindel, 1996] also expressed concerned about the gap between the level of technology within contemporary project management software and that which has been presented in the literature. However, as well as being critical of the fact that software does not use up to date techniques and has no capability for modelling many higher level management concerns, contemporary software development was criticised for focusing on improving usability and presentability of output, rather than on technological advances.

It was argued that a new method would need to be made available, a new paradigm was required which was capable of meeting the demands conventional software packages were not. To allow this next step in project management software to be achieved, a "compound application" was proposed. This would be based on a relational database, and be composed of many different integrated applications. Each of these applications would be able to manipulate information in the common database in a way most appropriate to its function.

2.5 Conclusions

In this study of the current practice of project planning and control a very wide number of topics have been covered. These first of these was how the consideration of non-quantitative aspects of projects were affecting planning and control. It was shown how the way projects were beginning to be considered in terms of the entire project life cycle, rather than only as far as the completion of its construction, and how attempts were being made to address the conflicts of interest which often exist between the parties involved.

---

[1] 1 Humphreys[Humphreys, 1991] p469

After assessing the current state of the qualitative side of project management, attention was turned to the analytical methods used in project planning and control. The development of scheduling techniques, from the original PERT/CPM model has been outlined. The associated functions of estimating and forecasting have also been discussed, with criticisms.

From these early analyses of projects techniques of modelling and optimising were developed. To begin with, these were time based approaches, but later developments also looked at cost and other measures of project performance in some detail. These techniques included both exact approaches, which were often criticised for being too demanding to compute, and heuristic rules, which were criticised for being too simplistic and not being close enough to the optimum. Because of these criticisms, heuristic approaches to optimisation were applied to these problems. These yielded better solutions than the heuristic rules without requiring the level of computational resources associated with the mathematical optimisations.

It has been discussed how systems analysis has been critical of the analytical methods applied to projects. This has principally been for two reasons. Firstly, analytical techniques tend to be focused upon specific problems, and usually fail to recognise the interrelationships between different parts of the project. Secondly, the tendency of project planners to focus on obtaining good results from such techniques has tended to obscure the need for qualitative aspects of the project, such as relationships between different parties, to be properly considered.

It has also been shown that while both these criticisms are valid, they do not invalidate the need for analytical planning. Indeed, some methods proposed which used systems analysis aimed to increase the effectiveness of analytical planning by placing it in its proper context within the project as a whole. Leading from this, some techniques which allow the holistic approach recommended by systems analysis to be implemented within the analytical side were discussed.

While the development of these techniques was significant, it was shown how the development of such techniques would be irrelevant unless they were implemented

within project management software. While software of this nature does not exist, it was shown how the need for next step in software had been demonstrated, and a possible solution assessed.

This chapter has shown how analytical techniques and optimisation have been developed, and how, arising from systems analysis and trends in the management of projects, a new need for a more holistic optimising technique has been created. The need for, and requirements of such a technique will be discussed in the following chapter.

Chapter 3 The Need for Global Optimisation Within Projects

It has been shown in the literature survey that there are many techniques already available for project optimisation. However, these methods all have limitations which are discussed in this chapter. These limitations show the need for a new, global approach to project optimisation, which is proposed in this chapter. The aims of such an approach are set out, and from these aims physical requirements of a global model are derived.

3.1 Limitations of Existing Methods

Project optimisation has become an increasingly important field of study in recent years. This is fuelled by a desire to complete projects more quickly, for a reduced cost and to a higher quality. Many of the techniques arising from this research have been simple mathematical optimisations, aimed at solving specific problems within specific stages of projects. However, the majority of these techniques, despite their sound theoretical basis, have not been applied to real problems[Heindel, 1996]. There are a number of reasons for this.

1. The optimisations often take the form of models which consider varying only a few aspects of the project[Walker, 1989], and ignore its implications for other aspects of the project.

2. These optimisations tend to have only one of two goals: to minimise project cost or project duration. However, there are usually many more important factors which dictate project success than these two values[Norbis, 1988]

3. The true mathematical optimum for the theoretical models often represents a solution which is unfeasible for the real project which the model is set up to simulate[Pidd, 1989].

4. Techniques which search for a true mathematical optimum tend to be very demanding on computational resources, particularly for very large projects. Any

savings they might make are offset by the cost of computing them, if it is even possible to compute them within a reasonable time scale at all[Li, 1993; Yau, 1990; Norbis, 1988].

Because of these criticisms, much attention has also been focused on the use of heuristic rules to improve project performance. Some of these have been employed in practice. However, they are limited because they fail to capitalise on the scope for optimisation which is available, and are also, like traditional methods, focused only on a few aspects of the problem.

3.2 Aims of the New Global Approach

These criticisms of both true mathematical optimisations and heuristic rules could suggest that the use of a numerical optimisation is limited within projects, and should only be used sparingly. Indeed, excessive dependence on numerical techniques and optimisation has been criticised for being directly responsible for projects failing to perform as planned[Buchanan, 1991; Becassi, 1996].

It is certainly true that there are many human and management aspects to creating a successful project outcome, and that optimisation (in its broadest sense of seeking the best solution which can be found) should not be restricted to the numerical modelling side of project planning. Nevertheless the use of numerical optimisation can improve project performance provided it is used appropriately.

The implementation of such numerical modelling currently facilitates decision making by providing an idea of how the project may best be structured, resources best allocated, activities best prioritised, etc., as well as providing an accurate account of how certain decisions will affect the optimal outcome of the project. Such analysis is a necessary part of project planning. It is only detrimental to projects when it is depended upon too much.

However, it is clear that current numerical techniques are not realising the full potential of numerical optimisation available to projects. A new approach to optimisation is required which can maximise the use of this potential. This approach

needs to be applicable to real projects, and able to find as good a numerical solution to a project optimisation as possible within time and computational limits. In order to set targets for such a project model, a set of clear aims must defined.

3.2.1 One Single Global Optimisation

It is commonly recognised in optimisation that optimising only small subsections of any system does not necessarily constitute finding the true optimum for that system[Guang-Yuan, 1992; M'silti, 1993; Stuckman, 1990]. Indeed, with a system of any complexity, it is expected that a series of "localised" optimisations will not find the global optimum. Projects, which are composed of highly interdependent subsystems, are no exception.

For example, one optimisation technique may focus on changing the orders in which certain activities are scheduled in a resource constrained project so that the project completion time can be reduced, while another technique may do so by changing the duration. Both of these are optimising subsystems of the project. The order in which activities are executed is one subsystem, and the activity durations and associated resource allocations is another, and these two subsystems are interdependent.

The current practice of optimising only these small subsystems should not be expected to find any global optimum. Not only will making one subsystem function as efficiently as possible in terms of some local criteria ignore its effect on the project as a whole, it will definitely not consider its impact on the optimising potential of other subsystems.

The only way the true project optimum can be found is through a global project optimisation. This would involve looking at the whole project within one single optimisation. In order to do this, a numerical model which is capable of incorporating all project variables must be generated. While it must be recognised that in real problems it may not be feasible to use absolutely all variables which exist within a project, it is important that as many variables as possible are included in order to maximise optimising potential.

3.2.1.1 The Whole Project - Defining the Project Life Cycle

In existing research, as was discussed in the previous chapter, project optimising techniques have been restricted to either the construction or the design phases of the project. Incorporating all the project variables within one optimisation, however, is not simply restricted to one or other of these phases. A true global approach to optimisation should consider the whole project life cycle.

The project life cycle is traditionally considered to be the phases through which a project passes between concept and completion (commissioning and handover). However, a common cause of conflicting interests within a project is the fact that the project manager is primarily concerned with the success of the project only up to what he sees as completion, and fails to understand fully the needs of the customer beyond this point.

The whole project, as the customer sees it, begins with a need. The project sets out to meet this need and ends once it has met the need, with the decommissioning of the project at the end of its useful life. The aim of the project is not necessarily to ensure completion of construction and commissioning on time and on budget, but to ensure that the project succeeds in meeting the need of the customer in the most efficient way possible. For this reason the whole project life cycle, from concept to decommissioning, should be considered.

The whole project life cycle can be broken down into seven distinct phases, which all have distinct purposes and characteristics.

1. The concept phase. In this phase the need is addressed, and solutions to that need are sought. A number of possible solutions are usually considered, which are expanded upon in the feasibility phase. Alternatively, it may allow the need to be resolved without the completion of a project. In this case the project will progress no further.

2. The feasibility phase. In this phase one or, more usually, several options are considered in more detail than the concept phase. This usually involves coming up with more accurate cost estimates, performance criteria and time factors for

these options, as well as an approximate design. For this phase to pass into detailed design, it is usually necessary to select one option. This phase can also reveal the need for re-evaluation at the concept stage, or reveal an alternative means for resolution of the need.

3. The detailed design phase. As well as producing a detailed design of the solution, this phase is also concerned with planning, including contractual arrangements, project scheduling and procurement of materials. If the project is acceptable, then it will progress to the construction phase. It is very uncommon and undesirable for a project to be reconsidered at this stage of the project, as considerable resources will have already been committed. This is particularly so in the later stages of this phase. In some fast track projects, the construction phase may already have commenced before this phase is finished.

4. The construction phase. When the project enters the construction phase, feedback of the whole project to previous stages becomes impossible, because part of the project already exists physically. However, feedback of parts of the project to the design phase, although obviously very undesirable, may prove necessary if mistakes in the design are revealed on construction which need to be amended. This phase is usually the most intensive, requiring many different types of materials and manpower to be present in a confined space for a short time. This phase's need of careful management has lead to its receiving much attention in the literature. Much has been said about the management, analysis and optimisation of this phase.

5. The commissioning phase. This stage involves the inspection and testing of all aspects of the project to ensure that it is able to meet the customer's need. This may reveal that parts of the project have not been designed or constructed to the required specification. Some project feedback would be required should any of these components have to be redesigned or reconstructed.

6. The implementation phase. This is usually the longest and most costly phase of the project, and involves the operation of the project. This is the phase of the

project which will demonstrate that the 'need' has been met. Again, it is possible for parts of the project to be fed back to the design phase, as elements of the solution may become inadequate over time. However, unlike the construction and commissioning phases, this is not necessarily to do with errors or oversights in the design or construction. The implementation phase is generally much more subject to a changing environment than any other phase, particularly to advancing technology. Sometimes a partial redesign and reconstruction of small component parts of the project is required. This partial feedback is often inevitable in this stage for certain types of project.

7. The decommissioning phase. This phase takes place when the solution has reached the end of its functional life, and involves the freeing up of all resources taken up by it.

Figure 3.1 illustrates how these stages fit into the project as a whole, and also shows the main groups of analytical technique commonly employed in the planning and execution of each stage.

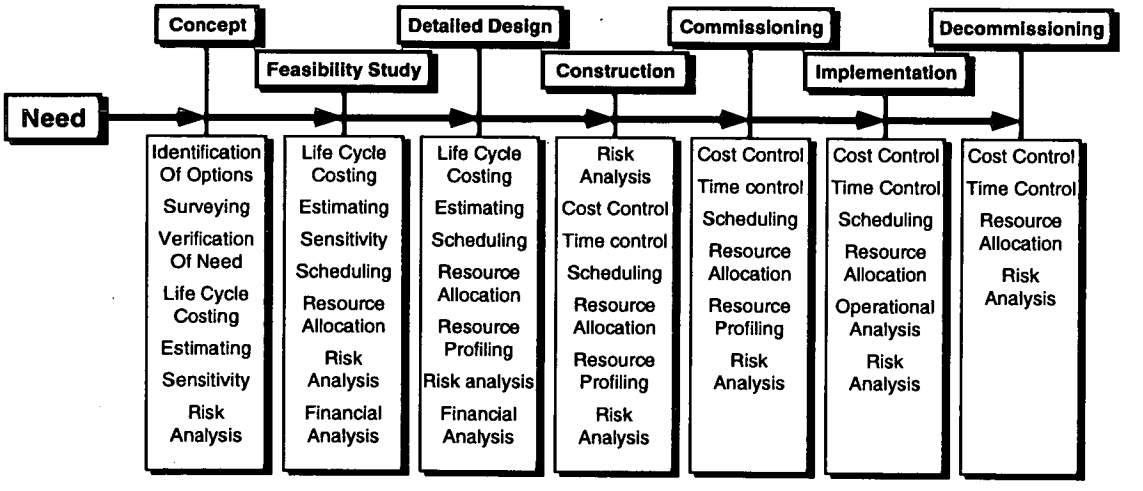


Figure 3.1: The Project Life Cycle

3.2.1.2 Expected Scope for Optimisation Throughout the Project Life Cycle

As the project passes through the various stages of its life cycle, the scope for improving project performance by optimisation becomes more restricted. At the start of the project the scope for optimisation is very large. There are many possible

solutions to the problem. As the project progresses beyond feasibility, one particular solution is opted for, and so scope for optimisation is restricted to fit within that solution. A great deal of optimisation is then performed in the detailed design phase, and continues to be performed throughout the project life cycle. However, as each stage progresses, the amount of optimisation which may still be performed is reduced, due to restrictions imposed during previous phases. By the time the decommissioning stage is reached, there will be very little further optimisation that can be performed. This is shown in Figure 3.2.

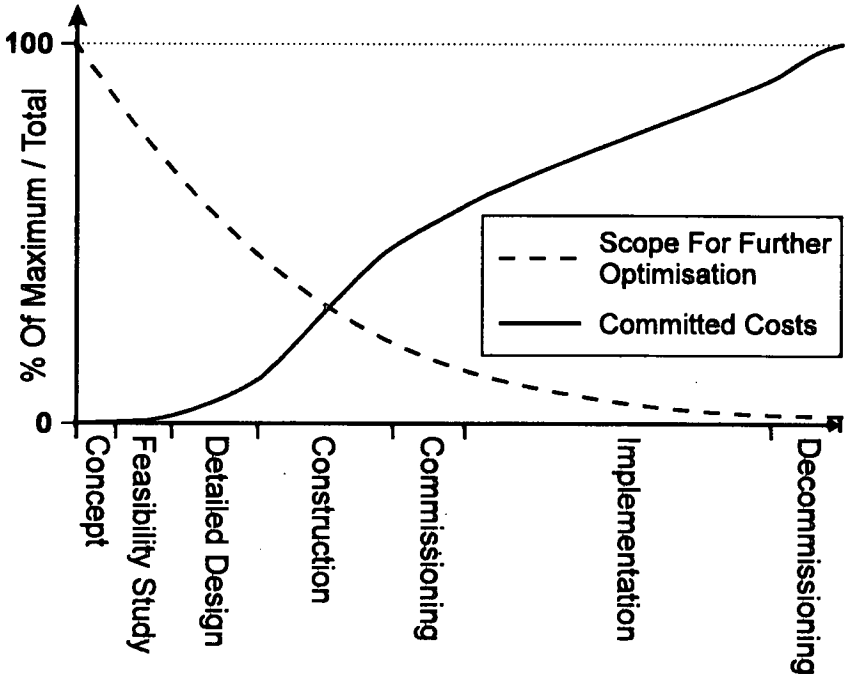


Figure 3.2: Committed Costs And Scope For Further Optimisation In The Project Life Cycle

As the project progresses, the amount of impact an optimisation may have is being reduced, as the values of many important variables are becoming fixed. Despite this, the detail to which the project is analysed will tend to increase, at least as far as the construction phase.

This analysis of the project is introduced with a simple model of the project at the concept phase. As the project progresses, it becomes more defined. Further analysis

is performed and, as the forms that various aspects of the project will take become more clearly defined, the amount of information which defines the project increases. This comprises breaking the simple model down into smaller subsystems. Each time the subsystems of the project are expanded, and the project is analysed in more detail, a whole new set of variables is introduced. These variables are inherent to the new subsystems being introduced, and could not have been considered before. Because this expansion of the project model is usually very rapid at least until the beginning of the construction phase, the number of variables introduced through the introduction of these new subsystems would be expected to exceed the number of variables fixed over that period. When this is the case, the number of variables available for optimisation will actually increase. Thus, while the impact an optimisation may have on the project performance is reduced, both the number of variables involved and the complexity of the optimisation is increased.

Because of this continual change of state of the model, optimisation cannot be performed at only one point within the planning process. A global optimisation which seeks to be relevant to the whole project must be performed repeatedly throughout the project, at each stage the project is analysed in more detail. This will ensure that the optimisation provides as accurate an indication as possible of how critical variables may be best set.

3.2.2 Accurate Representation of the Project

These critical variables are set by a decision making process, and the fixed values of these variables represent the results of the decisions made. It is important that the decision makers have information which is as accurate as possible. This not only applies to the information used for analytical planning, but also to the values achieved by an optimisation. In order to help the decision makers to make the best decisions for the project, it is essential that the optimised analysis uses the most accurate information available. This means using the most up to date information. Thus the concept of repeatedly performing the project optimisation with the most up to date information is not only consistent with the changing state of the project model, but also with providing a timely, accurate optimisation.

As well depending upon the accuracy of the information available, the accuracy of the model, and hence of the optimisation, depends upon the accuracy of the modelling methods which use this information. Obtaining accurate information depends greatly on the type and stage of project being optimised. Global optimisation's aim is to use this information to provide an accurate optimisation of the project. Though the accuracy of this information is vital, it does not form a part of the optimisation itself, and is therefore not relevant to defining the aims of a global project optimisation

The methods of modelling used within the project, on the other hand, are a key factor in optimisation. In order to perform an accurate optimisation, the model must be as accurate a representation of the processes and relationships involved in the real project as possible.

The generation of such a numerical model could take many forms. The number of modelling techniques available for projects varies considerably. Many different methods have been suggested in the literature, and many different methods are used in practice. Those suggested in the literature have tended to be restricted to one method of modelling, particularly where the research has been concerned with optimisation. In practice a number of techniques tend to be used. The main categories of analytical technique used to model projects in practice are shown in Figure 3.1, on page 45.

Despite the fact that the selection of one method for all projects has been common in the optimisation literature in the past, it is not realistic to try to select just one technique which will be appropriate for all types of project. Given their wide and varied nature, it is likely that the most appropriate way of modelling projects would vary considerably from one project to another.

As well as varying from one project to the next, any individual project is made up of smaller subsystems which may all be quite different in form. This implies that that, within any individual project, different component parts may lend themselves to

different types of modelling. This is why so many different types of analytical technique are used in practice.

In order to represent such a project as accurately as possible within a project model which might be used in a global optimisation, all these techniques must be used together. Thus, if the optimisation is to be as accurate as possible, it must be sufficiently flexible to allow a number of techniques to be used within the same model.

3.2.3 Finding an Acceptable Optimum

The use of so many different analytical techniques, and the wide range of variables to consider, makes the model to be optimised very large. Indeed those smaller, more localised techniques which sought the mathematical optimum have often been criticised for being too demanding on computational resources. If this is extrapolated to the global project optimisation it might seem that it is simply unfeasible to even attempt a global optimisation. Even if it were possible to employ an optimisation technique which was able to find the true mathematical optimum, it is almost certain that the time and computational resources required would be far too high for it to be used in practice.

It is possible, however, to use a near optimal or heuristic optimisation technique. This type of technique seeks out good near optimal solutions. This solution may not be the true, mathematical, global optimum, but it may be considerably better than any solution which might have been found using conventional techniques. Thus using a global optimisation in this way would not necessarily be concerned with finding the mathematical optimum. Rather, it should exploit the increased scope provided by the use of the global model to produce solutions which improve considerably upon the solutions found using conventional techniques.

One difficulty of using such numerical techniques, however, is that it can become difficult to know how much optimisation to perform. The value of the best solution found by a numerical technique is usually related to the amount of computational effort expended within the optimisation. If more resources are available (more time,

greater processing power, etc.), then the solution found will usually be better. For any project there is a trade-off between the computational resources used and the value of the best solution found. It is important that some balance is found where the value of the best solution represents a considerable improvement without the optimisation demanding too much time and resources for computation. Therefore the optimisation should look for the best solution which may be found using the time and computational resources available.

3.2.4 Multiple Objectives

In a project there may be any number of measures of performance. These may include measures of time and cost, as well as other indicators of project performance. These will be selected according to the priorities of the project team and/or the client. All these factors are important in determining the project's success. The traditional approach to project optimisation is to optimise in terms of one value. This means selecting only one of these important factors, and optimising in terms of it. This has been extensively used in the past, but has also been criticised for ignoring its effect on other critical factors which determine the project's success. Also, most of these techniques optimise only in terms of time or cost. While reducing the time taken to construct a project and/or the project cost is almost always desirable, there may be other critical factors which are actually more important.

Therefore the project optimisation should aim to be able to consider all important measures of performance at the same time. This includes both measuring project performance in terms of its important factors, and ensuring that improvements in one factor are not unnecessarily detrimental to other factors.

3.3 Requirements of a Numerical Model

In summary, the need for a new approach to project modelling and optimisation has been explained, and the aims of such an optimisation have been defined as follows:

1. Representation. The approach must be capable of numerically modelling the whole project, from concept to decommissioning, and of including all (or as many

as possible) of the project variables within the optimisation. It must also be capable of supporting different measures of project success.

2. Accuracy. The approach must be capable of modelling all projects as accurately as possible. This can be achieved by allowing the model to incorporate many different types of analytical technique.

3. Effort required to optimise. There is a trade-off between computational effort and how good a value the optimisation will be able to find. A reasonable balance should be aimed for, where sufficiently good solutions are being obtained without an excessive and expensive amount of computational effort.

Now that these aims have been identified it is possible to define a model to meet them. This model will be defined in the next chapter.

Chapter 4 A Numerical Model for Global Project Optimisation

In the previous chapter the need for a new approach to project modelling and optimisation has been explained. The aims of such a model have also been identified as follows:

1. Representation. The approach must be capable of numerically modelling the whole project, from concept to decommissioning, and of including all (or as many as possible) of the project variables within the optimisation. It must also be capable of supporting different measures of project success.

2. Accuracy. The approach must be capable of modelling all projects as accurately as possible. This can be achieved by allowing the model to incorporate many different types of analytical technique.

3. Effort required to optimise. There is a trade-off between computational effort and how good a value the optimisation will be able to find. A reasonable balance should be aimed for, where sufficiently good solutions are being obtained without an excessive and expensive amount of computational effort.

The development of a numerical model to meet these aims can be divided into two logical parts. The first is the general model, which incorporates those characteristics which all projects have in common. This defines a general case for which general expectations of behaviour can be defined and catered for within the model. This creates a core model onto which the second part of the model can be constructed. The second part of the project model is that which incorporates those characteristics which are specific to any particular project or type of project. Those aspects specific to the model of a particular project are discussed in later chapters, where the general model is applied to specific cases.

In this chapter the general model is developed. This begins with the definition of those aspects of the model which are common to all projects. It then builds on this by defining how analytical techniques should be incorporated into the general model

to develop the specific model. It also provides a general definition of how optimisation should be applied to a project.

4.1 Definition of Terms

In order to be clear on the aspects of developing a suitable general format for a project model which might meet the requirements described, terminology is defined for the different aspects of a project model.

- Project input data. Project input data represents all the information required to create a representative model of the project. Project input data includes activity level considerations such as durations, resource usage and costs; as well as project level considerations such as resource limits and breakdowns, financing and overheads.

- Project input variables. Project input variables are the project input data whose values can be directly controlled by the project planner. Most project input data will be fixed inherently, probability distributions, or a function of other data. However, some will be directly controlled by the project planner. The project planner seeks the best values for these variables. In current practice, this is determined using engineering judgement or, at best, using some sort of local optimisation. It is the aim of global optimisation to help the project planner to chose the best possible values for these variables.

- Scheduling Algorithm. The scheduling algorithm is the core of the project model, as it determines when everything will happen. The project can, at any stage, be represented as a series of activities. The scheduling algorithm dictates when these activities are scheduled and the consumption of resources by each activity, usually ensuring that consumption will never exceed an imposed maximum.

- Activity Selection Techniques. The activity selection techniques form part of the scheduling algorithm. They constitute any techniques employed to determine how the scheduler will schedule some activities in favour of others in a resource constrained situation. This will be explained more fully later.

- Pre and Post Scheduling Algorithms. A Scheduling logic on its own does not really constitute an accurate model of a project. The project input data will often not be in a form which the scheduling algorithm can use directly and the information generated by the algorithm will often need to be translated into more meaningful indicators of project performance. The pre-scheduling algorithms are the analytical methods used by the model to convert the diverse input data into the specific information types required by the scheduling algorithm. Some of the data generated by the pre-scheduling algorithms may also be used as output. The post-scheduling algorithms generate more meaningful output using project input data in combination with output from the scheduling algorithm. Such output data might include resource histograms, costs, etc.

- Project output data. The project output data is a vast set of data, and comprises all the various values calculated either directly by the scheduling algorithm, or by the post-scheduling algorithms.

- Project output criteria. The project output criteria are the most important of all the project output data. They represent the characteristics of the project which are most important to the planners. These values are the measures of project success which will be monitored as part if the optimisation.

- Solution Set. This term is only used when considering the optimising of the project model. A solution set for a project is a set of project input variables, and the corresponding project output criteria generated by the model.

4.2 Definition of The Model

The definition of a valid project model begins with a framework. There are a wide range of analytical techniques which will be used in modelling the project. These techniques behave in different ways and often have different types of output. The model needs a core onto which these other techniques are built. This forms a framework for gathering the information generated into a meaningful form. This

framework needs to be flexible, and to provide a structure from which the analytical techniques can operate.

The core of the model proposed is a precedence network. This provides information on two of the most common forms of interaction or conflict between project subsystems: precedence relationships and resources. Precedence relationships define how certain aspects of the project may require other aspects of the project to be in a certain state before they can operate. Conflicts can also arise between two subsystems which compete for the same limited resource(s). If a scheduling algorithm which considers resource requirements is used, then it can ensure that these conflicts are properly accounted for. Other considerations of conflict and interaction between subsystems could also be added to this basic scheduling algorithm for specific projects.

The scheduling algorithm provides an indication of when certain activities begin and end, and what resources they use over that period. Because this data is always calculated, for any project type, the analytical techniques used as part of the specific global model can be divided into two types: pre-scheduling algorithms and post-scheduling algorithms. Relevant project input data would be used by the pre-scheduling algorithms to generate network data, such as activity durations and resource requirements. This would permit a variety of types of input data and analytical techniques to be used, as appropriate for the type of project. The scheduling algorithm would then provide an accurate measure of when activities begin and end. It would also be able to avoid the violation of any resource ceiling, a process which could use a variety of rules. Then the post-scheduling algorithms would be able to use both project input data and output of the scheduling algorithm to generate project output criteria. The project output criteria are those values which measure the performance of the project. They will be discussed in more detail in 4.2.5. The whole process of the model calculating project output criteria from project input data is illustrated in Figure 4.3.

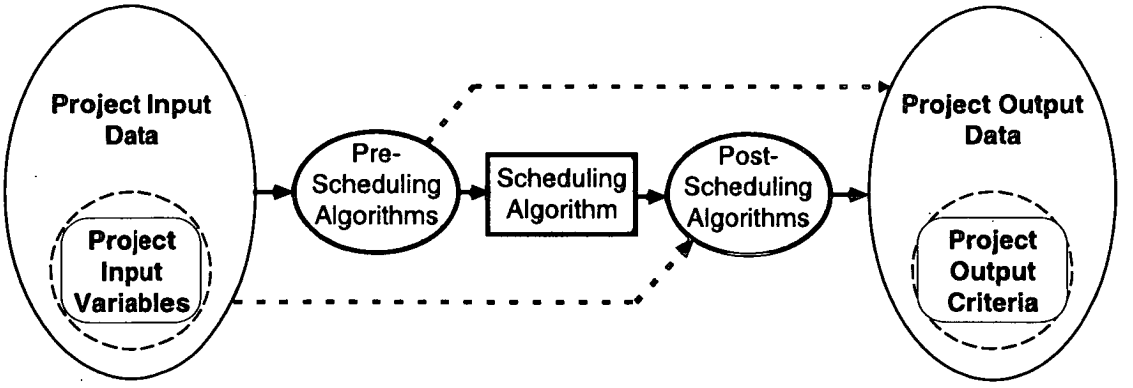


Figure 4.3: Basic Format Of The Model

4.2.1 The Scheduling Algorithm

The scheduling algorithm needs to be able to determine when activities begin and end, and what resources they will consume. There are essentially two ways of scheduling the project. One moves through the activities one by one working out when they can be scheduled (activity based scheduling). The other steps through the project time period by time period, observing the resource use and scheduling activities (time based scheduling).

With both time based and activity based scheduling, activities may be prioritised such that activities of a high priority are scheduled ahead of those of a low priority. This means that important activities can be given a reduced chance of being delayed due to the limited availability of resources. It is also possible to consider the use of rules which permit the increasing of available resources in order to allow the activity to be scheduled earlier than would be possible otherwise. These constitute extensions to the scheduling algorithm which are applied to specific projects, and are therefore not part of the core project. Nevertheless, it is important to consider how appropriately they could be applied to either type of algorithm.

4.2.1.1 Activity Based Scheduling

Activity based scheduling works through the project activity by activity. Any activity may only be considered once all its predecessors have been scheduled. Once this is done its precedence relationships define the earliest period at which the activity may be scheduled. The algorithm then checks that scheduling the activity at that period

would not violate any resource constraints. If it does not then it is scheduled at that period and the resource use profiles are adjusted accordingly. However, if insufficient resources are available to schedule the activity then it is scheduled at the next period for which adequate resources are available.

As the algorithm progresses, it must maintain a set of activities whose predecessors have all been scheduled. This set could be large, and would consist of many activities whose earliest possible schedule times, according to precedence relationships, vary a great deal. When these are sorted into some sort of order of priority, they are not necessarily a set of activities which would all be scheduled at the same time.

This kind of modelling has received criticism for not considering the project as a queuing system[Kavanagh, 1985; Jaafari, 1996]. In reality, it is argued, a project is not just a series of activities with precedence constraints, but a series of activities which queue for resources. At any time period there are a number of activities which may be scheduled. In a project, there are situations where activities which are in progress or about to be scheduled all interact with each other in some way. At this time period there are a number of different combinations of activities which may be scheduled. Scheduling these activities is a case of selecting the best combination of activities to schedule at this period, rather than selecting the best activities in themselves. Therefore it is important to be able to assess the full set of activities waiting to be scheduled, rather than considering these activities one at a time. With activity based scheduling this is difficult to do, as the algorithm is designed around scheduling one activity at a time. However, this is much easier with time based scheduling, as the algorithm actually requires this set of activities to be generated anyway.

4.2.1.2 Time Based Scheduling

In time based scheduling, the scheduling algorithm iterates through the project, time period by time period. At each period the algorithm evaluates which (as yet unscheduled) activities are available to be scheduled, and adds them to a queue. Activities which are available are those whose predecessors have all been scheduled

and which can be scheduled at the current timestep without violating precedence constraints. The algorithm schedules activities from the queue in such a way as to avoid resource use exceeding the ceiling. The algorithm may also permit the temporary raising of a resource ceiling to allow important activities to be scheduled. Any activities which are not to be scheduled remain in the queue until the next timestep, when the algorithm will add any new activities to the queue and attempt to schedule the queue again. The algorithm continues until all activities have been scheduled.

The time based algorithm does consider the project as a queuing system. At any time period a queue of activities is always available. The question "What is the best combination of activities to schedule?" can be answered using this set of activities. How this question is answered may vary from project to project. The techniques which do allow this question to be answered are called activity selection techniques. In the literature, a number of activity selection techniques have been suggested. They could all be incorporated into the algorithm very easily, and constitute extensions to the algorithm which are applied for specific cases, rather than part of the core algorithm.

Normally, such techniques will use existing data from the scheduling algorithm as well as data from the pre-scheduling algorithms. However, some more complex activity selection techniques will need more information than this. This may mean that they require data which comes from post-scheduling algorithms. Some of these techniques will use data from post-scheduling algorithms which only operate on data generated by the scheduling algorithm up to the current timestep. These will be discussed in section 4.2.3. However, some techniques also exist which try to assess the impact of different selections of activities on final project outcomes.

These techniques tend to try to look at the final outcome of the project under different selections of activities, and select the best set of activities based on this analysis. This constitutes a local optimisation which finds heuristic solutions. Using such techniques takes some of the scope for optimisation away from the global optimisation by using that scope to perform its own local optimisation. This local

optimisation is unable to consider the interdependencies of its variables with other variables in the project. It will also not usually perform its optimisation in terms of a relevant project output criterion. Therefore this type of technique should not be used in the global model.

4.2.1.3 Data Required

The scheduling algorithm requires activities with durations, dependencies and resource allocations, as well as resources with ceilings and possibly also some breakdown of those resources into groups or teams. It also requires any additional activity selection techniques which are to be applied. Some of this data will come directly from the project input data. Some of it will be generated from other project input data by the pre-scheduling algorithms. Therefore the analytical techniques which constitute the pre-scheduling algorithms must be employed in a way which generates data of this type for the scheduling algorithm, as well as any other data required by the post-scheduling algorithms.

The data output from the scheduling algorithm will typically be activity start times, finish times and resource uses. Some more complicated activity selection techniques may also generate data as a part of their execution. Post-scheduling algorithms will use that data generated by the scheduling algorithm, as well as other data from the pre-scheduling algorithms and project input characteristics. This process generates the project output data

4.2.2 Pre-Scheduling Algorithms

The scheduling algorithm requires data of a quite specific type: activity durations, precedence relationships and resource requirements, as well as some project level information such as the resource ceiling. In addition to this, any extensions to the algorithm may require data of their own. Much of this data will be defined as part of the project input data. However, some values may need to be determined from project input data which is of a different form to that required by the scheduling algorithm. When this is the case pre-scheduling algorithms must be employed to make this conversion.

For example, an activity's duration may need to be calculated from the resources to be allocated to it, taking various measures of productivity into account. Thus one or more pre-scheduling algorithms would be used to determine the duration from this project input data.

The pre-scheduling algorithms may be simple algorithms. Sometimes, to get from the project input data to the input for the scheduling algorithm, a number of algorithms may be employed. Each algorithm in turn adds information to the project, using project input data and output from other algorithms. Thus at any stage of the project two types of information are available: project input data and output from all previously executed algorithms.

The selection of pre-scheduling algorithms will vary greatly from project to project. What is required is a set of algorithms which convert the project input data into accurate activity based information for use within the scheduling algorithm. Therefore the algorithms must be selected so that together they are able to use the project data in the form that it is in, and model the input for the scheduling algorithm as accurately as possible.

Provided a combination of algorithms is capable of using the project input data and of producing the required data for the scheduling algorithm, then it will fit into the model. This approach to modelling provides a flexible format which would be able to incorporate most analytical techniques, while still ensuring that they will fit properly into the global model.

4.2.3 Post-Scheduling Algorithms

Once all the pre-scheduling algorithms have been executed all the information required by the scheduling algorithm is available. There may also be a lot of information which is not required by it. This information will have been generated as a by-product of arriving at the complete set of information required by the scheduling algorithm. That is not to say, however, that this information is useless. It may also be used by the post scheduling algorithms, along with the output for the scheduling algorithm and other project input data. Therefore it is not only necessary to ensure

that the pre-scheduling algorithms selected will provide accurate input for the scheduling algorithm, but also that it will provide accurate input for any of the post scheduling algorithms.

The post-scheduling algorithms, as with the pre-scheduling algorithms, must be selected and implemented in a way which will allow the project model to generate accurate project output criteria. It should be able to use all the appropriate information available, including that output by the scheduling algorithm. The analysis performed should also be appropriate for the project type, so that it evaluates the project output criteria accurately.

An example of post-scheduling algorithms would be the algorithms which calculated the final discounted cost of the project. They would require various information including activity times and costs, resource costs and other costs which apply across the whole project.

Normally the post-scheduling algorithms will be executed once the scheduling algorithm has been fully executed. As has already been discussed in section 4.2.1.2, some activity selection techniques require data which comes from post-scheduling algorithms. To be more specific, they may require data from algorithms which require schedule data. Perhaps the best example of this is those selection techniques which, to operate at any time period, require up to date cost information for the project at that period. This could appear to be a circular problem. However, these selection techniques typically require information which is accurate only as far as the current time period. Therefore these particular post-scheduling algorithms would be required to be executed only up to the current time period.

With a time based scheduling algorithm the schedule data up until the current time period is final. That is to say, the data up until the current time period will not be changed during further execution of the scheduling algorithm. Therefore the post-scheduling algorithms required could be executed up until the current time period, and still have access to all the data they require. When these types of activity selection technique are employed, some of the post-scheduling algorithms will have

to be executed in tandem with the scheduling algorithm, rather than being executed only when it is complete.

For example, an activity selection technique might consider the cost of the project to date. Therefore, an up to date value for the project cost must be available at each time period. This requires that the post-scheduling algorithms which determine the cost of the project be updated after each time period.

4.2.4 Project Output Data

All the analysis performed on the project – the pre-scheduling, scheduling and post-scheduling algorithms – all generate data which defines characteristics of portions of the project, or the project as a whole. Some of this data is used by other algorithms, and some is simply generated as a by-product of executing the algorithm.

4.2.5 Project Output Criteria

The project output criteria are the numerical values that the implementation of the model is aiming to produce for the particular project. Their values provide some indication of the project's performance, and can be used both in monitoring and optimising the project.

There are a number of different roles which project output criteria can fulfil. They can provide a measure of an important aspect of project performance which can be maximised or minimised (optimised). They can also provide a measure of project performance which is simply monitored as a part of the optimisation, so the model can easily provide an indication of how certain aspects of the project are performing at the same time as an optimisation.

The actual project output data used as project output criteria will vary considerably from project to project. This is because projects often have different aims. All projects begin with a need, which it aims to resolve. The types of project output criteria required to measure how well the project is meeting this need will depend greatly on upon the nature of the project.

How this need will be defined, and how it may be represented within the project during optimisation cannot be addressed properly without proper consideration of the process of optimising the project. Therefore optimisation will be discussed in the next section, and project output criteria and project input variables will be defined as part of this discussion.

4.3 Optimisation

In order to define how the project output criteria will be used in practice, as well as identifying how the model may be optimised, it is necessary to briefly examine optimisation theory. Optimisation of a system involves searching for an optimum, which is defined in the dictionary as:

> **optimum** *op'ti-mem, n* that point at which any condition is most favourable: — *pl* **optima.** — *adj* (of conditions) best for the achievement of an aim or result; very best. [L, neuter of *optimus* best] [2]

Mathematically the system of optimisation is composed of two components. The first is the domain, or feasible region. The domain is the set of all the possible combinations of values for the independent variable parameters which define the system. This is a subset of n dimensional space, where n is the number of the independent parameters. This set of parameters may be infinite. In the case of project optimisation, the feasible region is all the possible solutions of the model whose performance would be acceptable, and whose execution is realistic, and the independent parameters are the project input variables (a finite set of input parameters).

The second component of the system is the objective function. The objective function maps from the domain to a single one dimensional value, and may be either

---

[2] The Chambers Dictionary, p1187.

linear or non-linear. In the proposed project model, the pre-scheduling algorithms, scheduling algorithm and post-scheduling algorithms together constitute the objective function.

$$\text{Feasible Region } D \subset \mathbf{R}^n$$
$$\text{Objective Function } f: D \rightarrow \mathbf{R}$$

This can be considered as a surface in n+1 dimensional space, and is sometimes called the optimising surface. Optimisation is the process used to find the vector in D which yields the unique maximum or minimum value of f on this surface, which is called the optimum, usually denoted x^*.

$$x^* = \begin{bmatrix} \min\{f(x): x \in D\} \\ \text{or} \\ \max\{f(x): x \in D\} \end{bmatrix}$$

4.3.1 Identifying the Objective Function and Feasible Region

The problem with identifying an objective function for project optimisation is that there are many performance indicators (project output criteria). They may include measures of time and cost, as well as other indicators of project performance, and will be selected according to the priorities of the project team and/or the client. All these criteria are important in determining the project's success. However, traditional optimisation requires that the objective function maps to one single value, which is either maximised or minimised. In project optimisation this corresponds to either maximising or minimising one project output criterion. If this is the case then an optimisation may tend to find solutions which, while producing very favourable values of the project output criterion being optimised, have undesirable values for other project output criteria.

This problem has been encountered in other optimisation problems where there is more than one objective. The field of optimising such models is known as multi-objective optimisation. Generally, the most common method of solving this problem is to apply some sort of weighting to each of the objectives, allowing them to be

amalgamated into one value. This technique is known as goal programming, and has already been used in resource constrained project scheduling[Davis, 1992].

In order to use this method within the model proposed here, it is important to be able to be able to apply weightings or priorities to project output criteria. The actual mechanics of how the weightings are generated will vary from solution method to solution method, but common factors do exist. Each of these approaches requires values of all project output criteria for any combination of project input variables. Many also require the rejection of any solution which contains project output criteria which lie outside some constraints imposed on them. These are therefore essential features of the global project optimisation model.

It must also support the imposition of constraints on the project output criteria. Constraints are imposed on the project to ensure that the solutions considered by the optimisation are restricted to ones which are realistic in practice. The inclusion of these constraints allows the optimisation technique applied to reject solutions with undesirable values of any project output criterion. The easiest way of implementing this is to impose maximum and minimum values for each criterion. Then if, while optimising, a solution is found in which violates the constraints imposed on any of these project output criteria, then it should be considered by the optimising algorithm as unfeasible. This means that the pre-scheduling algorithms, scheduling algorithm and post-scheduling algorithms together not only define the value of the objective function, but also play a vital rôle in defining the feasible region. In order to find out if any proposed solution lies in the feasible region, the algorithms must be executed.

When the model is optimised there is another set of constraints which is imposed on the project as well as those which measure different aspects of project performance. These are those which represent the fact that the values of many project input variables are restricted in some way. Project input variables must also have constraints placed on them. These typically exist in the form of maximum and minimum values which the variables may hold, as well as whether or not the set of possible values is infinite (a continuous set) or finite, and will be discussed more fully in section 4.4.4.

Constraints imposed on the model which do not represent constraints in reality, but exist to add an arbitrary structure to simplistic models should generally be avoided. Such constraints have already been identified for precedence relationships[Jaafari, 1996], and may also exist for other types of constraint which might be applied to a global model.

The methods of constraining projects proposed would be common to all projects. This means that the optimising techniques used could be quite general. All they would have to do would be to search in a feasible region which is defined by project input variables and project output criteria. Once programmed, the techniques could be used for any project, without any significant adjustment to the algorithms which drive it. Of course, where there is scope for the fine tuning of a technique then it may have to be re-tuned for different projects, but if this can be changed at run-time then the computer code remains the same, and is transferable. If an optimising technique had to be recoded for each new project then it may make the use of global optimisation too difficult. This means that the method of imposing the feasible region must be defined in some detail.

4.3.1.1 Defining The Boundaries

In an optimisation the optimising algorithm should, generally speaking, only find optimal solutions in those areas which lie within the feasible region. This means that it should lie within constraints imposed on each project output criterion. Traditionally there are two ways of dealing with this problem: hard constraints and penalty functions.

Where hard constraints are employed the optimisation is simply forbidden from looking in areas where the constraints will be violated. When penalty functions are employed, however, the value of the objective function is penalised by a certain amount, depending upon by how much the solution is infeasible. If the solution is only just infeasible, and is close to being feasible, then the penalty is small, but if it is not close to being feasible then that penalty will be higher. Thus a solution is not measured simply in terms of whether it is feasible or not, but in terms of how infeasible it is. If it is not at all unfeasible then it is feasible, and suffers no penalty.

If it is unfeasible, then its infeasibility is measured and a penalty to the objective function applied accordingly.

The application of this penalty means that the value of the objective function will appear to the optimisation to be worse than its actual value. This deters the optimisation from looking outside the feasible region without explicitly preventing it from doing so. The advantage of using this technique is that it introduces the notion that for the solution to be only just beyond the specified limit, but with a very good value of the objective function, may still be acceptable as an optimum.

When considering whether this penalty function method should be used for any particular constraint, or whether it should be a hard constraint, it is first neccesary to establish whether it is a hard constraint in reality. A constraint may be applied because the project planners feel it is undesirable for the optimal solution to be outside the specified field, although if a sufficiently improved value of the objective function could be obtained by going outside this feasible range a little then it might still be deemed as acceptable. This type of constraint can be realistically represented by the use of a penalty function. If, however, the constraint is a real constraint which cannot be violated then this constraint is a hard constraint. With this kind of constraint any optimal solution obtained by optimisation must lie within the specified feasible region. If it does not then the solution cannot be implemented, and is therefore invalid.

The selection of whether to use hard constraints or penalty functions on any particular project output criterion is not solely related to whether it is hard or soft, but can also be related to issues of optimisation performance. Sometimes the application of hard constraints can actually make certain optimisation problems very difficult to solve, particularly if the problem is highly constrained, as projects can be. This is particularly the case if the algorithm cannot start with a solution which is feasible. The algorithm can spend so much of its resources finding feasible solutions that the amount of resources applied to actually optimising the problem can be significantly reduced. If penalty functions are used then the optimisation is able to begin with infeasible solutions, and the optimiser will first reduce the value of its penalised

objective function by reducing the penalty and hence finding feasible solutions. The optimiser cannot do this as easily when hard constraints are employed because it has no way of measuring whether one solution is closer to being feasible than another. Of course, employment of penalty functions to define a hard constraint will always run the risk of finding optimal solutions which are just outside the feasible region, which, if a constraint is truly hard by nature, is unacceptable.

Perhaps one of the problems which has arisen in the past is that the decision of whether hard or soft constraints are to be used has been decided as a matter of strategy for all outputs from the optimisation. Rather than considering each output individually the decision is often made as to whether the constraints on the output are to be considered as hard or soft. Often this decision can be necessitated by the type of optimising technique being used. While this may be a reasonable approach for local optimisations where only one or two such constraints exist, it may not be feasible for a project in which a large number of project output criteria of many different types exist.

If all the project output criteria of a project are constrained using hard constraints, then it can inhibit the optimisation of the problem because the model becomes so highly constrained. It also represents a decrease in the accuracy of the model, because it fails to represent the fact that some constraints can be compromised slightly if it means finding a much better value of the objective function. Similarly applying all the constraints as penalty functions can run the risk of the optimisation returning a value which is infeasible in terms of one or more hard constraints. Therefore the ideal way of representing constraints is to represent constraints as they are - either as hard constraints or penalty functions. If the optimiser is capable of coping with both of these constraint types then the constraints on the project can be modelled as realistically as possible.

However, it must be recognised that if a project does contain a large number of hard constraints then the project may still be difficult to optimise because of these constraints. In this case it may be necessary to soften some of these hard constraints by representing them as penalty solutions. In order to do this as efficiently as

possible, how critical each hard constraint is should be identified. This would allow project planners to select the least critical constraints for representation with penalty functions.

It is also possible that, for certain optimisation techniques, once the optimisation is some way through it will have identified those areas from which feasible solutions are coming. At this point it may be possible to allow all those hard constraints which have been represented by penalty functions up until this point in the optimisation to be expressed as hard constraints without being significantly detrimental to the optimisation.

4.3.2 Aims Of The Optimisation

The definition of the feasible region in such a way as to maximise the ease with which the project may be optimised is vital. The optimisation, as has been discussed in the previous chapter, should look for the best solution which may be found using the time and computational resources available.

Finding an optimal solution for a problem as large as the global model of a complex project will not be realistic for a reasonable amount of computational effort. Many issues which would be considered within a traditional optimisation, such what the true value of the global optimum is and how close the optimisation is to it, are no longer of great importance. What is of importance to those planning the project is how much better a solution can be obtained by introducing the global approach when the amount of computational effort available to the optimisation is essentially fixed.

Optimisation techniques are usually measured by how much computational resources are required to obtain the optimal solution. However, with the global approach an efficient optimisation is the one which provides the best solution with a restricted amount of computational effort, irrespective of how close this value is to the true, mathematical global optimum.

Of course, all these problems, including the measurement of performance, the selection of the project output criteria and setting of their limits, are ones which will

need to be solved for individual project types. Although the identification of possible common factors and problems is useful, it is important to be able to apply these to a real, specific project model. Therefore the process of extending the general model which has been defined into a global model of a specific project or project type will now be developed.

4.4 Extending The Model

This definition of a general format for a global model provides a framework upon which a global project model can be built. It allows the creation of a model which may be tailored to a variety of situations. These would depend upon the type and stage of project, the type of information used and the preferred methods of analysis. Many different types of analysis could be implemented within the pre-scheduling and post-scheduling algorithms. This provides a means for all the information required to be represented. Given that many models used in optimisation have failed to consistently yield feasible solutions, it is important to recognise that the model may need to undergo considerable refinement before it may be effectively implemented in an industrial situation. Therefore the general model defined here is sufficiently flexible to allow many different types of information and analysis technique to be used.

In order to generate a model which is aimed at a specific project or project type, it is necessary to carry out the steps defined in sections 4.4.1 to 4.4.5. Because of the changing nature of the project, the model will need to change with it. The introduction of new subsystems will require the addition of new analytical techniques to the model to ensure that they are accurately modelled. This means that these steps will need to be repeated at different stages to make sure that the model remains an accurate measure of the project throughout all its phases. To do this from scratch each time an optimisation is required would be a very large and difficult task. However, once the model has been set up for the project in its initial state most of the work required for the model to represent the next stage will already have been done. Therefore, these steps only need be repeated for those aspects of the project which

will have changed. Also, once industrial experience with the model has been gained, it may be possible to identify ways in which this process could be accelerated.

It might be possible to generate a computer program or system of programs which are able to integrate a wide range of analytical techniques with the general model to form a specific implementation of the model. This might take the form of some kind of expert system, and could make the actual creation of the model quite easy once the component parts of the model have been identified.

There is potential for making this process quite straightforward. However, to assess the possible impact of such techniques would be speculation. The length of time required both to perform this process and use the results to generate the model is currently uncertain. Because of this uncertainty, it should be recognised that some degree of planning will be required to ensure that any specific implementation of the global model is ready to optimise by the time the project reaches the stage for which the associated global optimisation is planned. The generation of the model must be started in time for the model to be available once its stage of the project is reached. Therefore the process of generation should not necessarily be based on the current state of the project, but on the expected state of the project at the future stage for which the optimisation is required.

What is required is a process which flows logically from the stage of the project which the model is to be applied for, to the completed, ready to be optimised, global model of the project at that stage. Such a process is laid out in this section.

4.4.1 Identify Project Input Data

The project data is what defines the project at its current state. Therefore project planners who are to use the global model must be clear on what stage of the project any specific implementation of the model is aimed at, and what type of information will be available at that stage. Although the type of data must be known, the values and even quantities of various variables may be uncertain, but this is less important, as these values are usually only required at the start of the optimisation itself. The model should be able to read the data at the point of implementation, rather than

requiring it to be incorporated into the model at an early stage. This would allow it to use the most up to date information.

With any project at any stage there is a pool of data types which define it. Some of this data will be available intrinsically, while some will have to be collected or generated. Data collected is that which is found out by investigation, like a site investigation, for example. Data generation is where existing information is expanded by other means (e.g. by carrying out a design).

It is important that the project planners are careful to consider the control they have over the data used by the model. If they do not then it may result in data which can be collected or generated being considered as intrinsic. It may also result in data being collected or generated in a way which is inappropriate for the type of project simply because other possibilities were not considered.

For the intrinsic data, there is not real control over the type of data which will be available, but when data is collected or generated, care must be taken that the collection or generation techniques are the most appropriate for the project. If data which is not accurate or representative of the project is used, then the optimal solutions found are unlikely to be accurate.

4.4.2 Identify Project Output Criteria

The aim of the model is to generate the project output criteria which will be available for the optimisation. Therefore it is important to identify what aspects of the project it is desirable to monitor, constrain or optimise. Identification of the project output criteria then provides a clear aim for the analytical techniques which are to be employed as part of the model.

As has already been discussed, project output criteria should consider the whole project life cycle, and be relevant to the needs of the project client.

4.4.3 Identify Analytical Techniques

The analytical techniques are those techniques which are required to generate the project output criteria from the project input data. The criteria which analytical techniques should meet have already been defined in sections 4.2.1 to 4.2.3. Techniques should accurately model the project. This means that the most appropriate type of technique should be selected wherever a choice exists.

The final combination of techniques implemented should flow logically from the input data to the output criteria. These techniques should be applied by the planners to create a clear and consistent structure.

This structure is important so that the planners can see clearly how the flow of the model, from technique to technique, runs. If it is difficult to understand how any set of techniques proposed fit together, then it becomes difficult to say with certainty that that combination of techniques accurately represents the project. It is vital to the optimisation that the project output criteria are accurate. This means that project planners should select a set of analytical techniques which is straightforward, representative of the project and not difficult to understand, so that they can ensure that the project output criteria will be evaluated accurately.

As all algorithms should fit into the categories of pre-scheduling algorithms, post-scheduling algorithms and activity selection techniques, a basic structure already exists, and would therefore be used as a starting framework from which the more complicated structure of the completed set of analytical techniques is created.

The number of individual techniques which may be selected to fit into this structure will be wide and varied, and provided that they are appropriate to the project and the project output criteria, the selection of such a technique will usually be valid. However, techniques which perform any local optimisation should be avoided, as these will tend to reduce the scope for the global optimisation. Because they set values of project input variables themselves, these variables are not available for optimisation and the effectiveness of the global model is reduced.

4.4.4 Identify and Define Project Input Variables

In order to perform a global optimisation of the project it is essential for the model to have all its variable attributes clearly defined in a way which the optimisation can make use of. This can only be performed once the main analytical techniques have been identified, because the nature of the techniques employed may influence what is variable and what is not.

Generally speaking the nature of these variables will vary greatly from project to project, so it is desirable that all these variables are reduced to a general form which any project input variable may take. This avoids the difficulty of having to explicitly build different aspects of the model's input into the optimisation. The process of identifying variables and defining them in this general form is made up of three steps: identification, examination and definition.

4.4.4.1 Identification of the variables

In order to perform a true global optimisation it is important to identify all the project input variables which are available to a numerical optimisation. This means that project planners may have to adopt a very open minded approach to what project input data is considered as being variable. There may be variables which were considered as fixed in the past simply because no optimisation technique was available to allow them to be considered as variable. Therefore the planners should not restrict themselves to those variables which are currently used in local optimisation techniques. All variables should be considered as part of the global optimisation.

Some project input variables will be easy to identify. Some types of project input data will always be variable, and some types will always be fixed. When this is the case, these are inherent properties of that type of input data. Identification of whether inputs associated with these data types are variables is straightforward. If it is of one of these types, then it is possible to say without further analysis whether it is variable or not. This allows the variables associated with these data types to be identified

early in the planning process, before more detailed information on the structure of the model becomes available.

However, some types of input may not be inherently fixed or variable. In one instance of a particular type it may be variable, and in another not. It is dependant upon how it fits in to the structure of the project input data. In order to identify which of this type of data is variable and which fixed, more detailed information is required of the project.

Therefore, the identification of these will have to be performed later than for those types which are inherently fixed or variable. This means that while it may be possible to plan most of the model early in the project, long before it is required, it may not be possible to generate the whole model until it is much closer to being required.

Therefore, planners must not only consider identifying types of data which are and are not variable, but also types that might be. This would also include the consideration of how the data required to determine whether input is variable may be collected and processed so that the model may still be generated in time.

The set of variables generated by the identification may be very wide. However, it should be recognised that the inclusion of many types of variable may expand the feasible region considerably. If the actual benefits of considering any particular kind of input as variable, rather than considering it as fixed, are low, it may reduce the ability of the optimising algorithm to find good near optimal solutions. This is because the number of good solutions, as a proportion of the whole feasible region, have been reduced. This makes it more difficult for the optimiser to find a good solution. Where this is the case, it may be necessary to consider that particular input as fixed.

4.4.4.2 Examining all possible values

For each variable identified, it is important to identify the full set of values which the variable may reasonably take. As for the identification, finding the range of values for some variables may, by their nature, be very simple. However, some variables

may need a wider consideration of how they fit into the project before their possible values may be evaluated. This means that their range of values cannot be identified until the model is close to being implemented.

As with the identification of variables, planners should be aware of this when planning the inclusion of these inputs as variable. It should be ensured that the data required to define the set of possible values will be available in time for the model to be generated.

4.4.4.3 Defining the variables

Any single variable is a one dimensional value. Therefore it takes one and only one of a certain set of possible values. This set may be finite or infinite. The simplest representation of this type of variable has three components: a maximum a minimum and a measure of whether the set is continuous or not. The maximum and minimum are the maximum and minimum values that the variable may take. If the set is continuous then the variable may take any real number between the minimum and maximum values, for example: $\{x{:}1 \leq x \leq 5, x \in R\}$. If the set s not continuous then it has a finite number of values. In many situations it will be adequate simply to describe a common multiple. Thus the set would simply be a sequence from the minimum value to the maximum value (e.g. $\{10,12,14,16,18\}$). For example, if the scheduling algorithm needs to know which of two activities it should schedule first, there are two possible permutations. Either one activity is scheduled first or the other is. This may be represented by a set of two variables: $\{0,1\}$, which is a sequence.

Sometimes sets will be structured in one of these two ways inherently. If they are not then it is possible for the real set to be determined from an input set like this by a simple mapping procedure. This ensures that the process of input and output remains simple.

4.4.5 Identify Additional Algorithms Required To Fit Analytical Techniques Together

Once this structure of analytical techniques has been laid out, it may be necessary to perform some simple translations of data types to allow the output from the algorithm representing one technique to be used as input for the next. For example, the basic costing of manpower may be a direct cost for each time period. However, men may be getting their wages for the previous week as one sum the following Friday. Therefore any algorithm which requires cash flow data will have to convert from a generation of cost per time period to the actual outgoings each following Friday. Therefore it is important to ensure that data types used as input for analytical techniques are correct, and that any algorithms required to make the necessary translations are employed.

As was discussed in section 4.4.4.3, it is also possible that some project input variables are not in a form which is not easy to represent in the form required by the optimisation. Therefore some translation may also be required to generate the real value of a project input variable from a simpler representation which is used by the optimisation.

4.5 Conclusions

In this chapter, a model which fulfils the requirements of a global optimisation model has been defined. It is composed of two components: the general model and the specific model.

4.5.1 The General Model

The general model contains a general framework which is common to all project types. Definition of this part of the model has included project input data, the scheduling algorithm, project output data, and project output criteria. How the project input variables and project output criteria would be used during an optimisation has also been explained, and types of optimising technique which are expected to be suitable for a project model have been identified.

4.5.2 The Specific Model

The specific model includes the general model and represents the actual model developed for a particular project. The general model is used as a framework and pre-scheduling algorithms, activity selection techniques and post-scheduling algorithms are added to the model to create a model which is relevant to a particular project. These represent analytical techniques which are used to determine important measures of project performance (project output criteria). The expected behaviour of such algorithms has been discussed.

Once the general definition of the specific model was complete, it was then possible to propose a strategy for generation of the specific implementation of the model. This showed how project planners might build on the general model to create a model which was relevant to a specific project.

1. Identify project input data

2. Identify project output criteria

3. Identify analytical techniques

4. Identify and define project input variables

5. Identify additional algorithms required to fit analytical techniques together

Using this general model, and the strategy for creating the specific model, it is now possible to apply the global model to existing projects. This will be developed in the following three chapters.

Chapter 5 Application of the Global Model to the Resource Constrained Project Scheduling Problem

Now that the model has been defined in the general case and a means of applying it to a specific project has been defined, it is possible to apply the model to specific projects. In this chapter the model is applied to 110 resource constrained project scheduling problems assembled by James Patterson[Patterson, 1984].

The purpose of applying the global model to a project is to improve the value of the solution obtained by the optimisation. Using the global approach increases the scope for optimisation, making the best solutions the optimisation might find better. These solutions represent more efficient ways of executing the project, and often cannot be found using local optimisations because they do not exist within the feasible region for these optimisations.

The value of applying the global approach is evaluated by applying the approach to a simple project model. If the project were applied directly to an existing large, complex project model without any experience in the behaviour of global optimisation, it could be very difficult to know if the optimisation were set up in an effective way. In dealing first with these smaller and simpler problems it is possible to analyse the results and effectiveness of applying the global approach before applying them to a real project.

The application of the global approach to these problems demonstrates how implementing the global approach increases the scope for optimisation. This can be achieved by demonstrating that the global approach can yield results which improve upon the optima found using local optimisation.

In order to obtain these values of optima for the local optimisations it is necessary to perform optimisations that consider using only one type of project input variable. These correspond to the local approaches to optimisation that have been performed in the past. Then all these different types of project input variable must be considered

within one optimisation. This corresponds to optimisation using the global approach, and yields better results than the local optimisations.

In analysing these results it is also possible to show how care needs to be taken over how which variables are included within the global optimisation, and how the global approach might perform with larger and more complicated projects.

5.1 The 110 Problems Assembled By Patterson

The problems used were assembled by James Patterson[Patterson, 1984]. In order to evaluate the performance of three exact optimisation approaches he found 110 problems which had been used to evaluate the performance of optimisation techniques in previous papers. These problems were then used to compare the three exact approaches. These problems have since been used in a number of papers that seek to evaluate the performance of optimisation techniques for the resource constrained project scheduling problem.

Along with the data "optimal solutions" were provided. In the past these solutions have been used as the targets for the optimisations. Normally the problems would be solved by identifying an optimal order in which to execute the activities. This optimal order would yield a schedule with a duration which was the lowest possible for the project without violating resource or precedence constraints. These solutions are optimal provided activities' durations are considered as fixed.

5.2 Representation of the RCPS Problem Using the Global Model

The resource constrained project scheduling problems used all have the same structure. They all have three renewable resources, which have a ceiling on their use, and activities which require these resources in varying amounts. Each problem is supplied with optimum start times for all activities for the case where there is no variation in project duration, which will yield the minimum project execution time.

Because all projects take the same form, it is possible to use the same specific implementation of the project model for each problem. The development of the model from the general case to the model to be used is therefore defined.

5.2.1 Project Output Criteria

The resource constrained project scheduling problems considered were created for the purpose of optimising (minimising) project durations. No other output types have been considered in association with these problems. Therefore the principal project output criterion will be project duration, and the object of the optimisation will be to minimise the project duration.

As well as minimising this value, it will also be necessary to identify a maximum project duration as a constraint. This value is required for two reasons. Firstly, it is required as a hard constraint to set the upper limits for the Minimum Early Start Times. This will be discussed more fully in section 5.2.4.2.1. The second reason an upper limit on the project duration might be applied is to prevent the optimisation generating too many unfeasible solutions. This need will be discussed more fully in section 5.3.2.

In a real project it will probably be necessary to impose a maximum value of project duration on any optimisation which might be carried out. This would represent the maximum permissible value of project duration. Any value greater than this would be highly undesirable. However, for the problems under examination no such maximum presents itself. Therefore some simple rule is required to establish a realistic and reasonable maximum duration.

When the project is initially scheduled, the activity priorities are all set to the same value. Under these conditions the scheduling logic will make no attempt to reorder the queue. This means that activities appear in the queue in the order they become available. Thus the project may be scheduled under an initial case, and a duration obtained. As the aim of the optimisation is to minimise the project duration then the best solution the optimisation will find cannot have a duration higher than this initial duration. Therefore it should be possible to set this as the maximum duration for the

project. However, when the project is optimised, it may be difficult for the GA to find better solutions than the initial value starting from the initial solution. It may be required for the GA to obtain a more diverse population of slightly worse solutions before being able to find better solutions. Given this fact it might be unwise to set the maximum duration without allowing a little "room to move" for the genetic algorithm. However, a penalty function cannot be employed because the calculation of the limits on the Minimum Early Start Time variable require a hard constraint. Additionally, applying a penalty function to the project output criterion being optimised is meaningless anyway, as the criterion itself performs the purpose of a penalty function already. Therefore the maximum value of project duration is set to the initial duration plus two.

5.2.2 Project Input Data

The projects are all precedence networks with activities, some of which will require some or all of the three renewable resources that are available to the project. These resources all have a fixed availability per time period. The time periods in question are just abstract units that have not been explicitly defined, but they will henceforth be referred to as days, for clarity.

The project input data are defined as follows.

5.2.2.1 Precedence relationships

The precedence relationships defined are all start-finish relationships with no lag or lead.

5.2.2.2 Resource ceiling

The resource ceiling is the maximum that a particular resource may be used in any time period. It remains fixed throughout the duration of the project, and always takes an integer value.

5.2.2.3 Activity duration

The duration of the activity defines how long the activity will take to be completed at the current level at which the activity consumes resources. The durations supplied

with the problems are integers, and all subsequently calculated durations will also be assumed to be integers. This is partly because all supplied durations are integers, and this leaves no precedent for using non-integer durations. Furthermore the use of non-integer durations is questionable in any case if the durations really are in days. If an activity is completed part of the way through a day, to start its succeeding activity that same day will probably not be feasible. It is usually better to wait until the following day before beginning any of the succeeding activities. For these reasons activity durations will be integers.

5.2.2.4 Activity's total resource requirement

The resource requirement of an activity is the total amount of any resource the activity will consume throughout its progress. The activity will require one total requirement for each resource in the project, and that requirement can be 0, in which case the activity does not require any of that resource in its execution. The value of this input may be a non-negative real number or a non-negative integer, depending upon the nature of the resource. For some types of resource it is not meaningful to break then down into less than one unit (prefabricated beams or men, for example, would only ever be used in whole units). These resource types would take the form of non-negative integers. In a real project deciding whether the total requirement would be an integer or not would be dependent upon the type of resource. In this problem it is not possible to make such inferences. However, all supplied values of resource requirement are integers, so there is no precedent for using noninteger values. Therefore integer values will be used.

5.2.2.5 Activity's resource allocation

The resource allocation is the amount of a particular resource that the activity is allowed to consume per day. There is a resource allocation for any resource which the activity requires (any activity with a non-zero resource requirement). Like the resource requirement, the resource allocation may be a non-negative real or integer number. For these problems the values supplied are all integers and the total resource requirement has already been defined as an integer value, so it will be assumed that all resources are non-negative integers.

The activity's duration, resource requirement and resource allocation are interrelated. Thus for any activity, of all the values which are of these types, some will be project input data and some will be derived from the project input data by analytical techniques. This will be discussed more fully in section 5.2.3.

5.2.3 Analytical Techniques

The optimum solution has been the one whose project execution time is the lowest. The basic scheduling algorithm will ensure that the precedence and resource constraints are not violated. Therefore, to use the model to solve this particular project type, the only extensions to the model required are some way of permitting the variation of activities' durations and activity selection techniques (some way of dealing with activities which compete with other activities for limited resources).

5.2.3.1 Pre-Scheduling algorithms

The only pre-scheduling algorithm required is one to relate duration, resource requirement and resource allocation. Where resource requirements exist, the activity's duration (d), total resource requirement (R) and resource allocation (r) are all related by the following equation:

$d = \frac{R_i}{r_i} \forall i \in S$, where S is the set of all resources.

If it is assumed that R remains constant, then there are two possible variables, the duration or the resource allocation.

5.2.3.2 Activity selection techniques

There are two simple approaches to applying activity selection techniques within this model. The first is to simply identify a series of priorities so that the activities can form a queue in order of importance. This has been tried in the past in the form of heuristic rules such as the early start time (EST), where activities with lower values of EST are considered to be more important than those with higher values. This rule has had some success at reducing project duration, but usually fails to find the optimum solution. It is therefore proposed to use an arbitrary number to represent

the activity's priority. This number dictates how important the activity is. The higher its value in relation to the values of the other activity's priorities, the greater its importance, and the closer it will be to the beginning of the queue. This technique has been used in the past to find near optimal solutions[Lee, 1996], and has also been implemented within some project management software.

The second way of dealing with the resource constrained nature of the project is to allow some activities to be explicitly delayed until a certain time period. In the past the true mathematical optimum for the model was found by identifying configurations of activities such that the activities do not violate any precedence relationships, and resource usage never exceeds the maximum specified. This would specify that if all activities were started at specific dates then the project's duration would be at its minimum and resource constraints would not be violated. This means that the violation of resource constraints could be avoided by explicitly delaying certain activities beyond when they become available by precedence logic alone. This delay could be expressed for all activities in the form of what will be called the "minimum early start time".

Both of these techniques will be incorporated into the model used for the solution of these problems.

5.2.4 Project Input Variables

The problems under analysis are quite simple, and therefore there is little that is variable. There are two aspects of the project that are variable: the input for the activity selection techniques and the activity's primary resource use.

The selection of the best activities to be scheduled from a queue of activities ready to be scheduled has been a field of considerable study in the past. This has included various heuristic rules that have ranked activities or even different groups of activities in some order of priority. It has also included techniques that will stop or "pre-empt" activities with lower priorities, or elongate activities in order to schedule them at periods where resources are too constrained. Many of these methods can be very complex. However, they tend to be methods that apply a series of rules in a

fashion that will tend to yield near optimal localised solutions. They take advantage of the structure of the problem and attempt to obtain near-optimal solutions in terms of some predefined objective without the need for a full-blown optimisation. This is not compatible with the requirements of the global model, in which the full scope for optimisation must be available to the optimising algorithm, rather than being used up by localised techniques. What is required in the analysis of these problems is some means of employing priorities in a way that can be changed by the optimisation as it searches for the best solution. Thus activity priority numbers are used.

5.2.4.1 Activity Priority

The operation of activity priority may be defined as follows. Each activity has a priority number. At each time period the activities in the queue are sorted in descending order of priority, so that the activity with the highest priority will be scheduled first. The scheduling algorithm then iterates through the queue, scheduling those activities which can be scheduled, and leaving those that cannot. Note that the scheduler will not stop iterating through the queue when it encounters an activity it cannot schedule. This is because later activities may still be able to be scheduled, either because they require less resources than the activity that cannot be scheduled, or because they require different resources altogether.

In order to allow the optimisation to have complete control over how the scheduling algorithm sorts the queue, the value of the priority number is a project input variable. For the purpose of this problem it was given an integer value and allowed to vary between the values of 1 and 100. These are arbitrary values, but as the largest problem size is 51 activities, this does permit the selection of a unique priority number for each activity in the project.

This variable only need be assigned to activities which have resource requirements. Activities without resource requirements will always be scheduled as soon as they are available, irrespective of any priority number, because the only way activities cannot be scheduled is by requiring a resource which is not available in sufficient quantities. Therefore the application of activity priority project input variables to activities with no resources is superfluous.

While the ability to determine priorities between activities is very important, having only these variables as project input attributes cannot be guaranteed to yield an optimal solution to the RCPS problem with fixed durations. This can be demonstrated by considering problem 1 from the data set.

Activity	Start Times		Duration	Precedents	Resource Use		
	Optimal	Ready			1	2	3
1	0	0	0	0	0	0	0
2	0	0	6	1	1	0	0
3	0	0	4	1	0	0	0
4	0	0	3	1	0	0	0
5	4	4	1	3	0	0	0
6	4	4	6	3	1	0	1
7	6	4	2	3	1	0	0
8	8	8	1	4,7	0	0	0
9	14	6	4	2	0	1	1
10	6	6	3	2,5	0	0	1
11	9	8	2	4,7	0	0	1
12	11	11	3	10,11,6	0	1	0
13	14	14	5	12,8	0	0	0
14	19	19	0	9,13	0	0	0

Table 5.1. Problem 1 from the Patterson data set.

Table 5.1 shows the activities' precedence relationships and resource usage for all three resources. The ceilings for resources 1, 2 and 3 are 2, 1 and 2 respectively. The table also shows the optimal start times (marked 'optimal') for the case where there is no change in activity duration, as supplied with the data, and the times (marked 'ready') at which the activities will be available to the scheduler assuming that all their precedents are scheduled at their optimal start times.

The problem with using priority scheduling rules arises with activity 9. The scheduler considers this activity as ready to be scheduled at day 6 (assuming that activity 2 is started at its optimal start time). Activity 9 uses one each of resources 2 and 3. On this day, activities 7 and 10 would also be available. Activity 7 would have been delayed because activity 6 would have been scheduled in preference to it on day 4, leaving no spare resource 1 until day 6, when activity 2 is completed. This activity has no resource conflict between either of the other two activities, and will therefore be scheduled at the current day. Activities 9 and 10, however, both require one unit of resource 3, and only one is available. In order to be consistent with the

optimal start times it is necessary to schedule activity 10 on this day. This means that activity 9 must wait before being scheduled.

On day 8, two more activities become available: activities 8 and 11. Activity 8 requires no resources, and is scheduled on this day. Activity 11 requires one unit of resource 3, which is still fully used, and must therefore wait until one of the activities using it is completed. This occurs on day 9, when activity 10 is completed, freeing up one unit of resource 3. Again, in order to be consistent with optimal start times, it is necessary to schedule activity 11, leaving activity 9 in the queue. On day 10 there are no new activities in the queue, and activity 6 is completed, freeing up 1 unit of resource 3. This leaves the scheduler with only one activity in the queue of activities available to schedule, and all the required resources available to schedule it. The scheduler will therefore schedule activity 9 on day 10, before its optimal start time of day 14. Activity 9 uses 1 unit of resource 2, the ceiling, and will therefore prevent activity 12 from being scheduled until day 14, when activity 9 is completed. This causes the project to have a final duration of 22 days, rather than the optimal 19.

The existence of this problem with priority techniques lead to the consideration of another means of controlling the delay of activities in order to obtain optimal project durations: minimum early start time.

5.2.4.2 Minimum Early Start Time (MEST)

The minimum early start time constitutes a minimal start time for an activity. When employing this the scheduling logic will not make any attempt to schedule the activity before its MEST. Rather than determining which activities should be scheduled and which left in the queue in a situation where there is a resource conflict, use of MEST moves activities out of conflict with each other. Unlike activity priority, use of this method does provide the means of finding the true optimum for the problem with fixed duration.

This can be shown by considering the situation where all activities' MESTs are at their optimal values, i.e. their values are their start times for the optimal solution. On the first time day, day 0, only those activities whose optimal start time is 0 will be

available in the queue. It will be possible to schedule all these activities immediately, by the definition of the optimal start times. At the next time period any activities become available in the queue, those activities' optimal start times will be the current time period. As all the preceding activities have been started at their optimal start times, it must be possible to schedule all the activities in the queue. This will also be true at any subsequent time period. Thus all activities will be scheduled at their optimal start times, and the duration will be the optimum.

5.2.4.2.1 Limits

Unlike activity priority, the limits for the MEST cannot be set arbitrarily. The earliest value the MEST can take is the activity's EST by network logic. Any value earlier than this will be redundant, as the scheduling logic will never consider it until its EST in any case. This rule is sufficient for the case where activity durations are fixed. Similarly where activity durations are variable the minimum effective value of the MEST can be determined by taking the ESTs from a forward pass of the network logic with all the durations set to their minimum values.

Extending this, the maximum value could be determined by using the LSTs from a backward pass of the network logic. However, the project scheduled under resource constrained conditions will usually be longer than the duration determined by network logic alone, so it may be desirable to delay some activities beyond their LSTs in order to obtain the optimum duration. The solution is therefore to use a different value as the project duration used as the LFT at the start of the backward pass. Under traditional network logic, the EFT for the project is used as this value of LFT for the project. However, if the backward pass is performed using some known maximum duration for the project as its LFT, then the LSTs will be at their maximum values, since if the activities were delayed any further, then the project would exceed its maximum by network logic. Therefore any value of MEST which is higher than this value will definitely lead to unfeasible solutions, and is therefore not worth considering.

Perhaps the most significant difficulty in employing this backward pass method of finding the upper limit is determining the required value of maximum duration, the LFT for the project.. If the aim of the optimisation is to determine the optimum duration then perhaps the obvious value of maximum duration to use would be the value of optimum duration for the case where there is no change in activity duration. These were supplied along with the project data. If this value were used then the optimum value will certainly be in this range. However, the purpose of these optimisations is to try to anticipate the behaviour of the optimisation of real projects. In a real project no such "optimal values" are known. The only known limit on a real project's duration is the upper constraint applied to project duration by the project planners. Therefore it is necessary to use this upper limit, as determined in section 5.2.1, as the maximum project duration when obtaining the upper limits for the MESTs.

If the scheduling logic is to have the MEST activity selection technique available, then it is necessary to determine an initial value of MEST which will not interfere with other optimisations. When MEST is not considered the activities are added to the queue when they become available by precedence logic. Therefore if the minimum values of the MEST project input variables are used then the scheduling logic will still be able to add all activities to the queue when they become available. Therefore the initial value of MEST should be the minimum value.

So setting the limits is determined by first setting all the activities' durations to their minimum value, then making a forward pass and backward pass taking the project's LFT to be the maximum duration of the project. The lower limits of the MESTs are the activities' ESTs, and their corresponding upper limits are the LSTs. In addition to this the MEST's initial value should be the lower limit.

5.2.4.3 Activity Duration and Primary Resource Use

The duration of an activity, as has already been described, can be reduced to a simple equation:

$d = \frac{R_i}{r_i} \forall i \in S$, where S is the set of all resources.

The total resource requirement R_i remains constant for all problems, so the two variables are duration and resource use. If the activity requires more than one resource then there are several values of r_i which are unknown. However, only one value of r_i or d must be known in order to determine all the other values. Therefore there are two possible types of project input variable: duration and resource use.

To set the duration and then calculate the subsequent resource requirements is one possibility, although it is perhaps not the most realistic way. When a project manager wishes to reduce the duration of an activity perhaps the most common response is simply to increase the manpower assigned to that resource. This is because it is the allocation of some defining resource that really defines the activity's duration[Hackman, 1989; Weist, 1967] (although in a real project it is also subject to other factors[Hendrickson, 1987; Kavanagh]).

This defining resource may not always be manpower, but may be some other resource, such as plant, fuel or a raw material. Because of this the resource that defines the activity's duration is given the more general name "primary resource". The amount of the primary resource allocated to the activity, which will be referred to as "primary resource use" dictates its duration, and the duration is then used to dictate the allocations per time period of the activity's other resources. So if the total requirement and allocation of the primary resource are denoted R_p and r_p respectively then the equations for calculating the duration and requirements of other resources can be expressed as follows:

$$d = \frac{R_p}{r_p}$$

$$r_i = \frac{R_i}{d} \forall i \neq p, i \in S$$

This shows how the activity duration and resource allocations can be derived from the allocation of the primary resource. However, these equations assume real values,

and it has already been dictated that all three of these data types are integers. Therefore if any of the values found are non-integers they need to either be rounded up or down.

Firstly the duration needs to be rounded up. If it were rounded down then using the primary resource at its assigned rate would not meet the total requirement over the duration of the activity. Similarly, the resource allocations should also be rounded up, because if they are rounded down then the total resource requirement will not be met. The rounding up of the duration should be performed before the allocations of the non-primary resources are calculated, so that these values are calculated using the real, final value of the project duration, rather than a lower duration (which could lead to an unnecessarily high resource allocation for the non-primary resources).

Thus the calculation of the duration and allocations of non-primary resources from the value of primary resource allocation should be calculated as follows:

1. Divide the total requirement of the primary resource by its allocation to obtain the duration, rounding it up to the nearest integer.

2. Divide the total requirement of each of the non-primary resources by the duration, rounding them up to the nearest integer.

5.2.4.3.1 Limits

In a real project the definition of the limits on the primary resource requirement would be quite straightforward. There will usually be identifiable physical upper and lower limits on the allocation of resources to an activity such that any allocation outside these limits would simply not be realistic. These values would be dictated by the nature of the activity in question. For these sample problems, however, the nature of each activity is not defined so clearly. Therefore a simple means of obtaining upper and lower limits is required. The only limits which can be inferred from the project input data are that the lower limit must be at least 1, as 0 would result in an infinite duration and any number between 0 and 1 would be a non-integer. Similarly the upper limit must, at most, take the value of the resource

ceiling, as any higher value will make it impossible to schedule the activity at all. In addition to this value the upper limit must also not be greater than the total requirement of the resource. This value yields a duration of 1, so exceeding it cannot result in any reduction in duration. Therefore the upper limit is either the resource ceiling or the lowest primary resource allocation that yields a duration of 1, whichever is the lesser. As these upper and lower limits are the only limits that can be inferred, they are the ones used.

The scope for optimisation provided by the use of these very broad limits could be very large, particularly for the larger of the sample projects. It is unlikely that upper and lower limits applied for the sample projects would be realistic for a real project. The upper limit would not be expected to be as large as the total resource availability, because projects usually involve the sharing of a resource between quite a large number of activities at any one time. It would also not often be practical to assign only one unit of primary resource to an activity. This is particularly the case when the primary resource is manpower. Thus it may be possible that the large scope for optimisation introduced in the application of such limits on the primary resource use may be unrealistically large. This will be developed in the analysis of the results in the following chapter. However, in order to set the limits on the project it is necessary to select the primary resource in the first place.

5.2.4.3.2 Selection of Primary Resource

In a real project the primary resource will be dictated by the activity. There will be one resource that will dictate the duration more than any other. This will usually be manpower. However, in the sample projects the activities are supplied with only the project input data, so it is impossible to discern the primary resource in this way.

In the projects there were usually two or three resources allocated to any particular activity at different rates. In order to maximise the potential for optimisation the best choice of primary resource use would be the resource with the largest allocation of resources, as this would yield the greatest number of possible values for primary resource use. As was expressed in the definition of the limits, however, the number

of possible values as determined by the limits may already be quite large, perhaps significantly larger than that which is realistic if the projects were real. Therefore it is possible that further maximisation in the scope for optimisation could make the model less realistic.

In order to evaluate whether this is likely to be the case it is necessary to consider what happens to the field of possible durations when the resource whose requirement is the greatest is used as the primary resource. This can be done by expressing the increase in duration from the minimum to the maximum as a proportion of the minimum. These values can then be averaged for each project. These values are shown in Figure 5.1 for both the resource whose requirement is maximum, and the one whose is minimum. The mean variability in activity duration is plotted against the number of activities in the project.

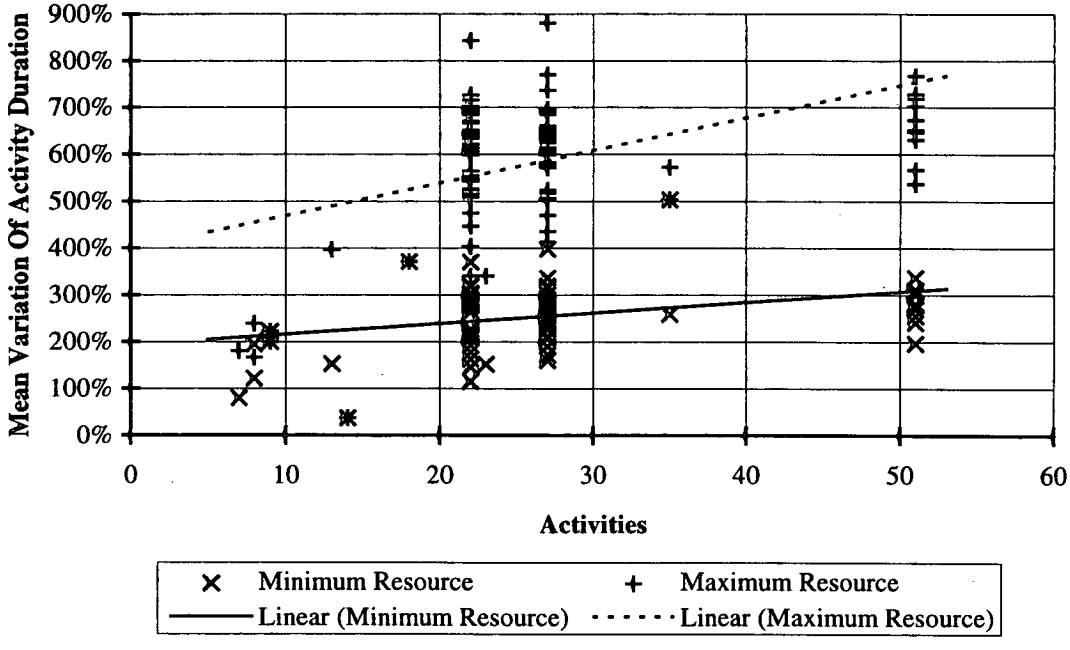


Figure 5.1. Comparison of activity variability for primary resource selections.

This figure indicates quite clearly that there is a considerable difference in the variability of the activities between the minimum resource and the maximum resource. Not only this but the trend lines show that while there is a lot of scatter there is a general trend for the activities to increase in variability as the project size

increases. Use of the maximum resource as the primary results in an average variability in activity duration of around 600%. An activity at maximum duration will be, on average, 7 times larger than the minimum duration. Using the minimum resource, however, this value is reduced to around 250%. Not only this but the trend lines suggest the variability increasing much more with an increase in project size.

In a real project the range of possible durations of an activity may, in some cases, allow a variation as large as 600%, but would certainly not be expected to on average. The activities in a project will usually vary by much less than this, and many activities will not be able to vary at all. Indeed, it is questionable whether an average variability of 250% is sufficiently low to be realistic for a project. However, as this is the lowest value available, it would be the best to use the resource whose allocation is least as the primary resource.

As well as causing problems with the realism of the model, the use of the resource whose allocation is the greatest could also cause a reduction in the efficiency of the optimisation. The lower limit on the primary resource allocation is 1. If the resource with the greatest allocation is used then the other resources will reach their minimum values before the primary resource. At this point any further reduction in the primary resource allocation will have the effect of lengthening the activity, but reducing in the consumption of only one resource. As all three resources are constrained the effect of this increase in duration will increase the overall consumption of the other two resources. This is likely to be detrimental to the project's execution time, because the aim of the optimisation is to squeeze the current resource consumption into as small a space as possible, and increasing the consumption will simply increase the amount of resource that must be squeezed into this space. Therefore to include this within the optimisation would be undesirable.

For these two reasons the resource whose total resource requirement was least was assigned as the primary resource, rather than the one whose requirement was greatest. This still gives considerable scope for optimisation without expanding the field more than is necessary for both realism of the model and efficiency of the optimisation.

5.2.5 Optimisation of the model

Optimisation of the resource constrained project scheduling problem with the objective of minimising time has been performed before. Although most the methods of solution developed in the past have tended to consider only the rearranging of activities in order to achieve optimal duration, some have considered varying resource use. However, the global approach to the solution of these problems, to optimise the two together, has not been considered. Thus by comparing the results of the optimisations of the three variable types alone to the results in combination (the global approach), it should be possible to identify some of the advantages and shortcomings of applying the global approach to project optimisation. In order to do this an optimising algorithm is required.

5.3 The Genetic Algorithm

Before the selection of a suitable optimising technique can be made, it is essential to consider what form the model will take, particularly when it is used to model real, complex projects. Projects can differ considerably in size and scope. This variation manifests itself not only in the size of the model, but also what pre-scheduling, post-scheduling and activity selection algorithms will be used, and the nature, number and constraints of both the project input variables and the project output criteria.

This large variation means that any technique which seeks to optimise the project model must be capable of effectively optimising a large model which is essentially a "black box" function. In order to be effective the technique needs to be flexible, so that it can be "tailored" to work well with any specific type of project. It must also be able to consider as much of the feasible region as possible, in order to find a solution which is a close as possible to the true global optimum.

A genetic algorithm would be expected to fulfil all of these criteria. They have been used with considerable success on a very large number of real problems. They have already been applied to these resource constrained problems by Lee et al.[Lee, 1996], who attempted to optimise them by using activity priority in combination with a genetic algorithm.

Genetic algorithms are very simple, yet there are many variables factors which determine the behaviour of the algorithm. This means that they can easily be tailored to specific problem types without any substantial change in the operation of the core algorithm.

5.3.1 Operation of the Genetic Algorithm

The operation of a genetic algorithm is very simple. Based on the concept of evolution by natural selection, the algorithm manipulates a "population" of possible solutions to the problem being optimised. This population essentially evolves towards a near optimal solution.

The algorithm uses a sequence of events which are very similar to the real, biological process of reproduction and natural selection. The algorithm changes the population by obtaining a number of new solutions, or "children", which are inserted into the population. Each time this is performed, the population is said to move on from one generation to the next. This is achieved by selecting a number of "parents" from the population for reproduction. These parents are then duplicated, or "reproduced", to create children. These children are "crossed over", which involves the switching of certain values of input variables between two children, and "mutated" which involves the selection of a new value for one or more of each child's input variables. These children then replace members of the existing population.

This operation is performed every generation, and maintains a continuously evolving population of solutions. Within this population there are two important factors. One is the "fitness" of the population (how good the values of the objective function are) and the other is the diversity of the population. What ideally happens in a genetic algorithm optimisation is that this population is diverse, having a lot of very different good solutions, before eventually converging to an optimal solution. This initially diverse population is required to allow as much of the feasible region as possible to be explored. Then, as the optimisation progresses, the average fitness of the population increases, and the diversity decreases because many parts of the feasible region will not yield sufficiently good solutions to remain in the population.

Eventually the population converges - all the solutions become very similar - as the optimisation gets close to an optimal solution, as solutions good enough to remain in the population will now only be found close to this optimum.

The efficiency with which the genetic algorithm finds these solutions is dependent upon a number of variables which exist within those algorithms which perform reproduction, crossover and selection. These include the methods used to select both the parents and the members of the population to be replaced, the frequency and position of crossover and the mutation rate.

5.3.1.1 Parent Selection techniques

There are essentially two types of selection technique, fit to fit pairing and random pairing. Fit to fit pairing involves matching parents in such a way that parent pairs have a very similar fitness. The process of selecting these pairs usually begins with the best solution, which is matched with the second best solution and so on, until the required number of parent pairs have been obtained.

Random selection can take two forms: unweighted and weighted. Unweighted selection is simple. The required number of parents are selected entirely at random from the population. This is usually not effective as poor solutions are as likely to be selected as good solutions, and the algorithm usually operates by making each generation slightly better than the next. For this reason, some kind of weighting is usually applied which allows better solutions to be given an increased chance of selection. This is usually done using a fitness function. The fitness function maps from a value of the objective function to a non-zero value of "fitness". Those solutions with a higher value of fitness correspond to solutions which are better. Note that in the case of function minimisation a lower value of the objective function maps to a higher value of fitness. The fitness function can be very simple – such as a linear sliding scale of fitness between the best and worst solutions – or really quite complicated. For this simple application, a linear sliding scale was used.

5.3.1.2 Crossover Points

A solution is a "string" of values of each project input variable, with corresponding values of project output criterion, including the one being optimised. It is this string of inputs, which corresponds to the chromosome in real genetics, which defines the value of the objective function.

When the algorithm performs the crossover operation the algorithm moves down this string of values in a pair of solutions until it reaches a "crossover point". At this point it continues to move down the string, but swapping the values of project input variable as it goes, until it reaches another crossover point (should there be one), at which point it stops swapping, but continues to move down. If it were to reach another crossover point it would begin to swap the values again and so on until it reaches the end of the string.

In some, more complicated genetic algorithms, attempts are made to try and group data together such that a number of crossover points can be identified which would not disrupt logical groups of data. Thus the crossover points can be limited to these points or given an increased probability of occurring at these points. However, for this simple optimisation it is difficult to identify any such groups of data, so the crossover points are evaluated entirely at random.

5.3.1.3 Mutation rate

It is through mutation that new values of project input variable are introduced into the population. Mutation occurs randomly at a fixed rate. If that rate is 0 then, over a large number of generations, the algorithm would eventually filter out all but one value of any particular project input variable. If that rate is 1 then the problem becomes a Monte Carlo random search, as all the values of project input variable will be set randomly by mutation. The best mutation rate is usually closer to 0 than to 1.

5.3.1.4 Replacement techniques

Once the children have been created by the process of selection, crossover and mutation, they need to be inserted into that population. In order to do this the algorithm must be able to select the existing members of the population to be

replaced. There are five main ways of replacing solutions: weakest individual, weakest parent, both parents, random and weighted random.

Weakest individual, as its name suggests, involves replacing the worst n individuals (solutions) in the population, where n is the number of children. This method is very effective at moving quickly towards an optimum by removing all comparatively poor solutions of the population. However, it can tend to do this at the expense of not maintaining a diverse population, and hence home in on a local optimum rather than the true, global optimum.

Weakest parent behaves very similarly to Weakest individual, but makes more of an attempt to ensure that population diversity is not lost. The weakest parent is replaced with the best child. Thus the "genetic information" in the parent is not necessarily lost as much of it will be retained in the child. Indeed, it is hoped that the best information from the displaced parent will be in the child, although it this may not always be the case.

The "both parents" replacement technique works similarly to weakest parent, except that both parents are replaced by the new children. This retains most of the information from the parents, with a small amount being lost to mutation, but it can sacrifice the fitness of the population. If the probability of a child's being better than its parents is low then this will usually result in a decrease in the goodness of the population each generation.

Random and weighted random techniques both operate in the same way as for selection, except that the fitness function is reversed for weighted random selection. This is necessary to ensure that poorer solutions have a greater chance of being removed from the population, rather than the other way round, which would almost certainly lead to a deterioration of the fitness of the population.

5.3.2 Dealing With Constraints

There are two types of constraints which would usually be employed within the optimisation - hard constraints and penalty functions. Therefore the genetic algorithm should aim to support both.

While penalty functions may require some thinking on the part of the project planners to define - as a fitting penalty to the objective function must be defined for each degree of unfeasibility - it poses no problems for the operation of the genetic algorithm. If, after reproduction, crossover and mutation, a constraint represented by a penalty function is violated, the penalty is simply incorporated into the value of the objective function and the algorithm proceeds as normal. The definition of constraints as hard, on the other hand, poses problems for the genetic algorithm. If the solution is found to violate a hard constraint then the algorithm cannot proceed as normal. It is necessary to reject that solution altogether. Nevertheless, this need not necessarily mean that two feasible children cannot be generated for the parent pair which initially spawned the unfeasible solution.

Rather than simply accepting that a certain parent pair has not produced two feasible children, it may be desirable to attempt crossover and mutation again. This need not be done for both children if one feasible child has been obtained. The single child's string of project input variables can be filled with values from one of the parents, the parent in question being swapped at each crossover point. While it is possible to continually repeat this process until two feasible solutions are found, it is possible that some combinations of parents are very unlikely to produce feasible offspring. If this is the case then a large number of function evaluations may be required to obtain two feasible children, which is inefficient use of the computational resources. In order to counteract this the number of attempts that are made is restricted to a finite value. In order for this to be implemented, the algorithm should permit the rejection of a solution and be able to re-attempt crossover until two feasible children are obtained, up to a limiting number of attempts.

5.3.2.1 Constraints For The Sample Problems

In this particular problem there is only one project output criterion which needs to be constrained: the project execution time. As this function is being minimised it is questionable whether any project output criterion constraints should be imposed at all. However, the imposition of constraints will be useful for two reasons. Firstly it will ensure that excessively long project durations are not considered within the optimisation. Secondly it will provide an upper limit on the project duration as was required or the setting of limits for the MEST project input variables (as discussed in section 5.2.4.2.1).

The use of a penalty function on this value would clearly be ineffectual, as any unfeasible solution already has, by definition, a very poor value of the objective function. In addition to this the setting of the limits on the MEST project input variables requires the definition of a hard constraint. Therefore a hard constraint will be used. However, while it is important that this constraint be quite low in order to have a useful limit for the MEST project input variables, it is also important that the value selected is not detrimental to the optimisation's progress. In order to evaluate this fully it is necessary to consider the initial population, and how it will change in the early stages of the optimisation.

The initial solution of the problem must be a feasible one. It may be possible to generate an initial population using a Monte Carlo random search, but this means that considerable effort might be expended in searching out a population of feasible solutions. If, on the other hand, the population is allowed to begin from a number of identical, feasible solutions, then the initial population will not require any significant amount of effort, and the algorithm can establish a diverse population from this starting point.

In the establishment of this diverse population, which represents solutions from different parts of the feasible region, the solutions will gradually move away from the initial solution. When this solution represents a local optimum it may be necessary for the value of the objective function to deteriorate slightly in order that this diversification may occur. This is essential because if this diverse population cannot

be created then the algorithm will be much less able to consider the whole feasible region in its search for a near optimal solution. For this reason the upper constraint on project execution time is taken as the initial duration plus two. This provides the optimisation with enough space to establish a diverse population, without being excessively large, and hence detrimental to the setting of the limits for the MEST project input variables.

5.4 The Computer Program

In order to perform the optimisation it was necessary to create a computer program which contained the implementation of both the objective function (model) and the genetic algorithm. The algorithms to be incorporated into the project were defined as follows.

5.4.1 Scheduling Algorithm

The scheduling algorithm is the core of the project model. It iterates through the project, timestep by timestep, and attempts to schedule all the activities in the queue. The queue consists of activities which are available to be scheduled at that time period.

A simple flowchart of the scheduling algorithm is shown in Figure 5.2.

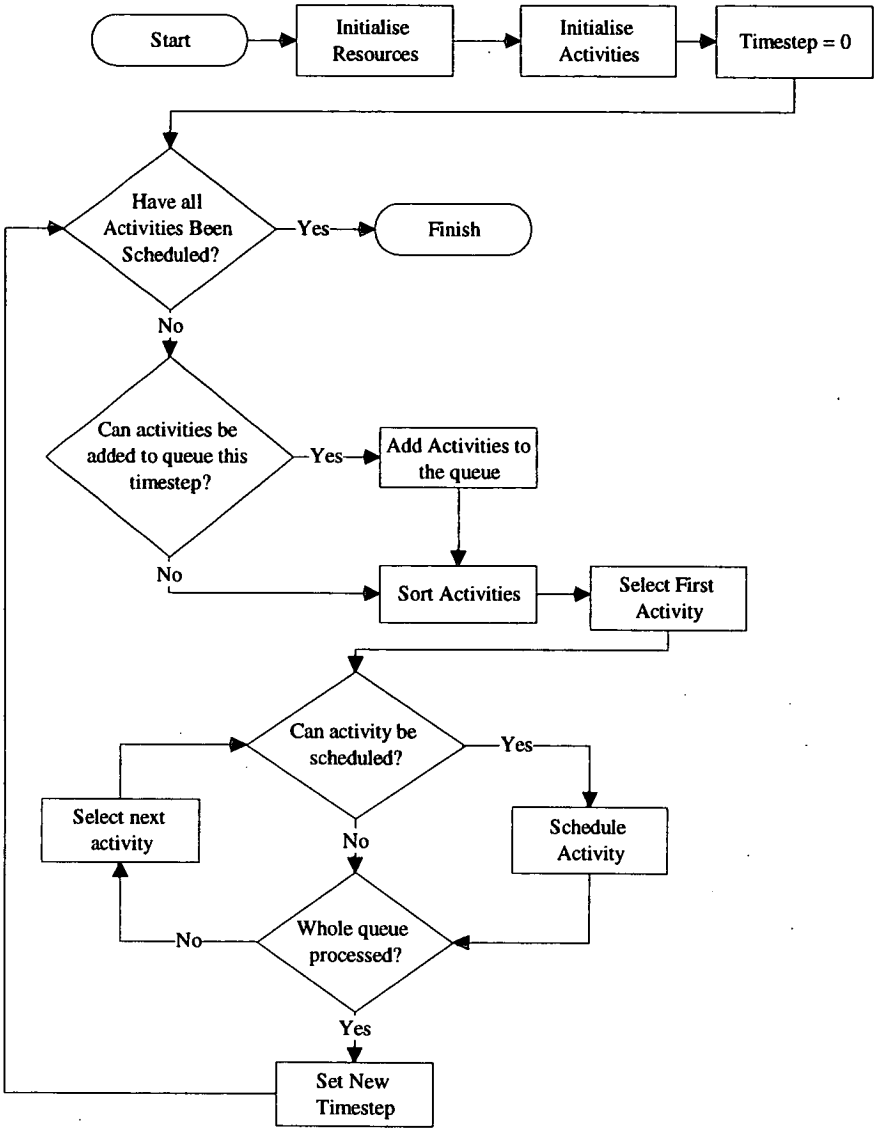


Figure 5.2. Scheduling logic.

Initialising the resources and activities consisted of processing all the pre-scheduling algorithms which were required in order to determine their characteristics. In this case this only required the setting of the duration and resource requirements from the primary resource use. In addition, it was necessary for the program to reset the activities and resources ready for scheduling. This included resetting the start times for the activity and the resource usage histograms. The resource histograms recorded how much of each resource was used each day.

The addition of the activities to the queue was performed by considering those activities yet to be scheduled whose preceding activities had already been scheduled. For each of these activities, if it could be scheduled in the current time period without violating its precedence constraints then it was added to the queue. Otherwise the activity was marked for addition to the queue at the first timestep at which the activity could be scheduled without violating its precedence constraints. In practice this meant maintaining three sets of unscheduled activities: the queue, the set of activities whose preceding activities had been scheduled (which will be referred to as the "precedence ready" set), and those activities whose precedents had not all been scheduled.

The precedence ready set was maintained by checking all the dependent (directly succeeding) activities to an activity when it was scheduled. Any of these dependent activities whose preceding activities had all been scheduled were added to the precedence ready set. In addition to being added to this set, their start times were given a provisional value. This was obtained by considering the precedence relationships and the minimum early start time. The earliest time which the activity could start, given each of the precedence relationships, was determined. This time and the MEST were compared, and the greatest selected. This was the earliest time at which the activity could be scheduled without violating the precedence constraints or the MEST. For example, if an activity had two finish-start relationships with a lag of 0 to two activities which finished on day 4 and day 7 respectively, and the activity's MEST was 6, then the provisional early start time would be set to 7.

The creation of this set of precedence ready activities made it much easier for the scheduling algorithm to determine which activities were eligible to be added to the queue. Rather than looking through the entire set of activities yet to be scheduled and not in the queue, the algorithm would only need to look through the precedence ready set of activities and extract those activities whose provisional start times were less than or equal to the current timestep.

Scheduling the activities was performed when an activity in the queue was encountered for which sufficient resources were available to complete it. It involved

setting the actual start time for the activity to the current timestep, and updating the resource histograms according to its consumption of them. For example, if 3, 4 and 6 units of a resource were being consumed over the three days that an activity would be in progress, and that activity used the resource at 2 units per day, then the new resource consumption would become 5, 6 and 8 units respectively.

As has been mentioned, the activity would then look at its succeeding activities to determine whether any of them can be added to the precedence ready set. However, there are certain circumstances under which these precedence ready activities would not be added to the precedence ready set, but directly to the queue.

Consider an activity which may only begin if another activity has begun, or is to be commenced on the same day. This corresponds to a start to start relationship with a lag of 0. If, when the preceding activity is scheduled, this succeeding activity is added to the precedence ready set then it will not be added to the queue until the next time period at the earliest. However, if its provisional start time is the current time period, then there is no reason why the scheduling algorithm should not consider it for scheduling in this time period. Therefore, in the situation where the provisional start time is less than or equal to the current time period, the activity is added to the queue, rather than the precedence ready set.

This requires the insertion of the activity into the queue midway through the processing of the queue. In order for this process not to invalidate the sorting of the queue, the activity must be inserted such that its priority is less than those activities ahead of it in the queue, and greater than those after it. However, this poses a problem. The algorithm moves down the queue, only removing those activities which are scheduled. If an activity is not scheduled then it remains in the queue, and will not be considered again until the next time period. The algorithm then moves to the next activity in the queue. Thus, midway through the processing there are a number of activities at the front of the queue which will not be considered again this time period. If the new activity were to be inserted amongst these then it would not be considered this time period either. Therefore the activity must only be inserted into the part of the queue behind the position of the activity being scheduled. This

would ensure that the activity is considered for scheduling this time period. The fact that this may mean that some of the activities which have already been considered may have lower priority numbers than this new activity is of no consequence, as the queue will be resorted at the next timestep.

The algorithm progresses in this manner until it schedules all the activities. Once this is done it is possible to obtain the total execution time for the project (he latest finish time for any of the activities). In future implementations, it will be at this point that the post-scheduling algorithms are executed to obtain other project output criteria.

A test of the scheduling algorithm can be found in appendix I.

5.4.2 Genetic Algorithm

The genetic algorithm has a very simple structure. It is simply required to repeat, for each generation, the processes of selection, crossover, mutation, evaluation, and replacement. The only aspect of this process which was unusual was the ability of the GA to repeat the crossover and mutation for those solutions which were not feasible.

All the rules for selection of parents and replacements were incorporated into the GA, so the user of the program would be able to select whichever rule was best for each. The crossover and mutation were controlled by having a crossover rate and a mutation rate. Each of these rates dictated how many genes (project input variables), on average, there would be between one crossover/mutation point and another. Thus, when the algorithm reached a crossover/mutation point during crossover and mutation, it would make the crossover or mutation, and then calculate the distance to the next crossover/mutation point. This calculation was made by obtaining a random number of steps to be incremented from a flat probability distribution where all the values from 1 to $(2 \times \text{rate} - 1)$ had an equal probability of being selected.

A simple flowchart of the GA is shown in Figure 5.3.

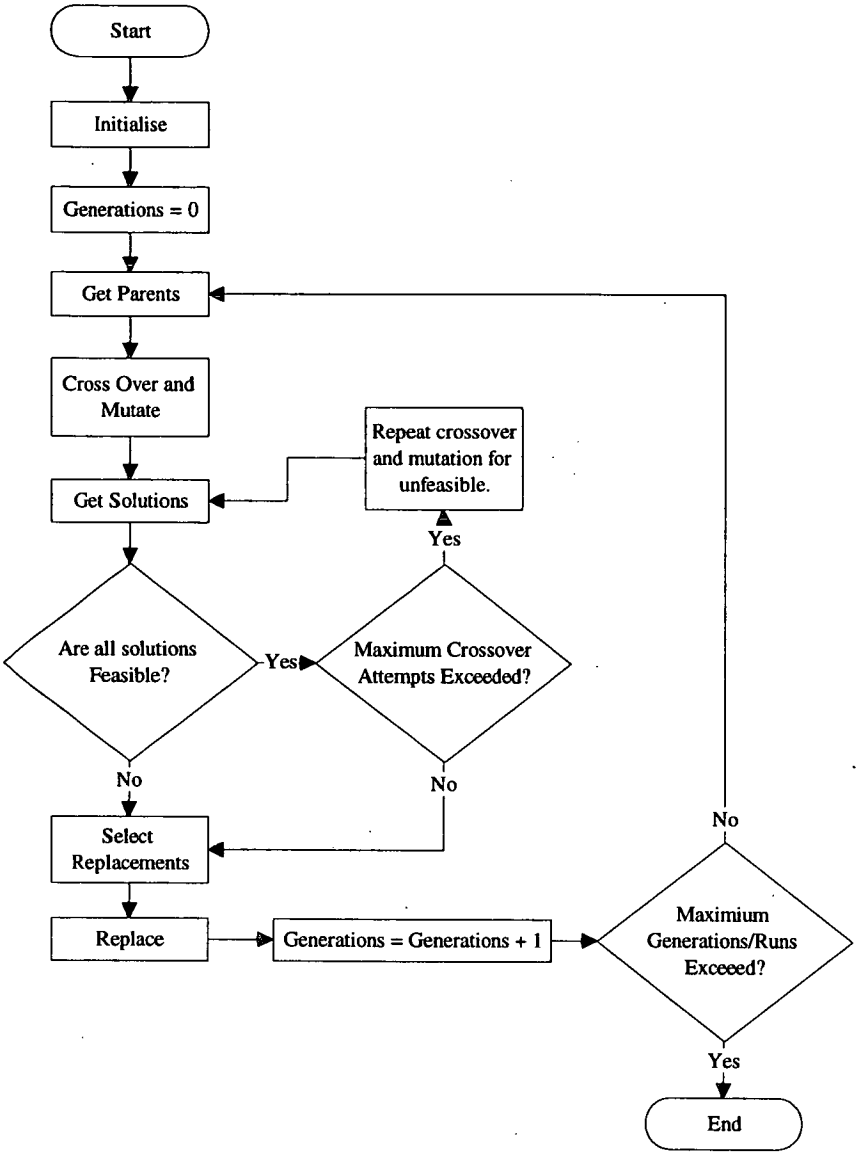


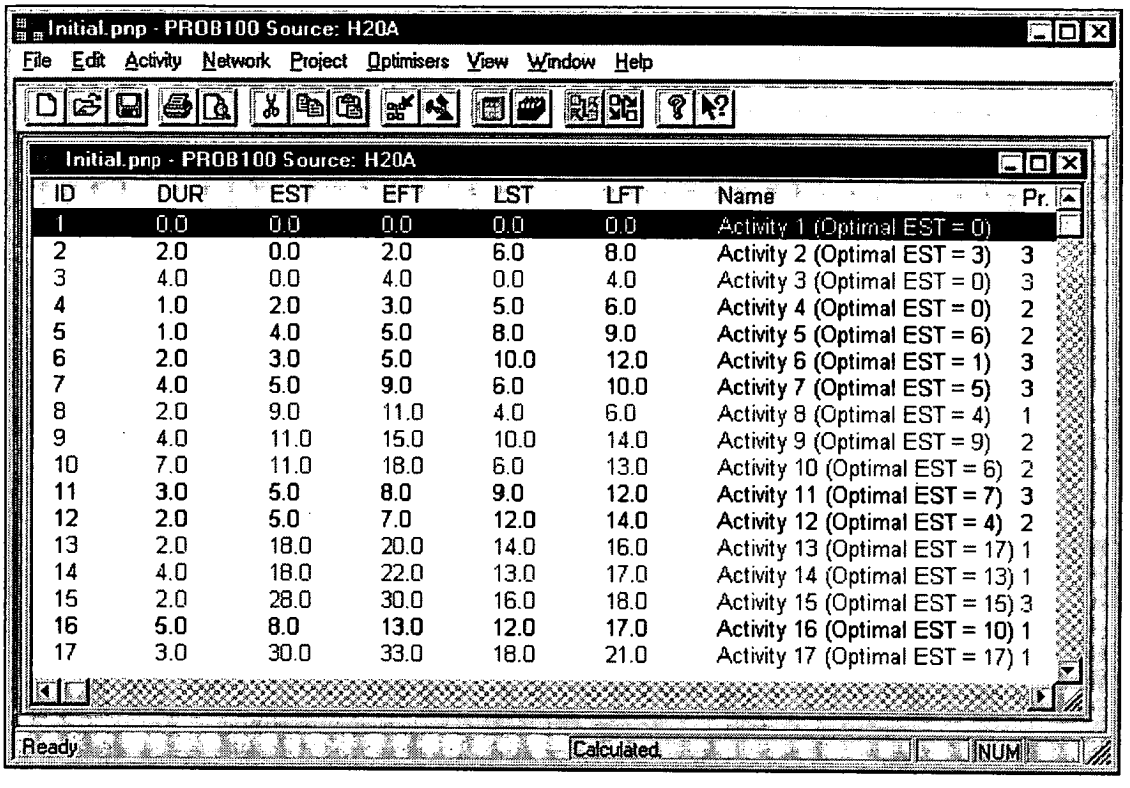
Figure 5.3. Flowchart of the GA.

5.4.3 Encoding The Program

The program was encoded in C++ as a Microsoft Windows® 32 bit application for an IBM compatible PC. In addition to the algorithms already discussed and their supporting data structures, a number of features were incorporated to facilitate the input, output and manipulation of data.

- The program was able to import the text files for the projects. This allowed the errors often associated with manual data entry to be avoided.

- The activities in the project were displayed using a simple graphical interface, shown in Figure 5.4, and access to all the project's information was provided through a series of dialogue boxes.



ID	DUR	EST	EFT	LST	LFT	Name	Pr.
1	0.0	0.0	0.0	0.0	0.0	Activity 1 (Optimal EST = 0)	
2	2.0	0.0	2.0	6.0	8.0	Activity 2 (Optimal EST = 3)	3
3	4.0	0.0	4.0	0.0	4.0	Activity 3 (Optimal EST = 0)	3
4	1.0	2.0	3.0	5.0	6.0	Activity 4 (Optimal EST = 0)	2
5	1.0	4.0	5.0	8.0	9.0	Activity 5 (Optimal EST = 6)	2
6	2.0	3.0	5.0	10.0	12.0	Activity 6 (Optimal EST = 1)	3
7	4.0	5.0	9.0	6.0	10.0	Activity 7 (Optimal EST = 5)	3
8	2.0	9.0	11.0	4.0	6.0	Activity 8 (Optimal EST = 4)	1
9	4.0	11.0	15.0	10.0	14.0	Activity 9 (Optimal EST = 9)	2
10	7.0	11.0	18.0	6.0	13.0	Activity 10 (Optimal EST = 6)	2
11	3.0	5.0	8.0	9.0	12.0	Activity 11 (Optimal EST = 7)	3
12	2.0	5.0	7.0	12.0	14.0	Activity 12 (Optimal EST = 4)	2
13	2.0	18.0	20.0	14.0	16.0	Activity 13 (Optimal EST = 17)	1
14	4.0	18.0	22.0	13.0	17.0	Activity 14 (Optimal EST = 13)	1
15	2.0	28.0	30.0	16.0	18.0	Activity 15 (Optimal EST = 15)	3
16	5.0	8.0	13.0	12.0	17.0	Activity 16 (Optimal EST = 10)	1
17	3.0	30.0	33.0	18.0	21.0	Activity 17 (Optimal EST = 17)	1

Figure 5.4. Graphical interface of program.

- In addition to the scheduling algorithm, the program was also capable of calculating the project duration using traditional network logic. This was useful for checking that the relationships had been imported correctly.

- Each of the three types of project input variable and their limits could be applied and set automatically, using the definitions of the limits in section 5.2.4. This allowed errors associated with manual calculation and data entry to be avoided.

- Access to the settings for the optimisation was provided by a dialogue box, which also contained the settings for logging the optimisation.

- The random number generator used by the genetic algorithm was re-seeded at the start of each optimisation. This ensured repeatability of the optimisations.

- The optimisation was able to log its progress. This permitted the best value found by the optimisation to be logged at finite intervals. These intervals were expressed as a number of runs.

The program was compiled, and the optimisations were run using an IBM compatible PC with an AMD K4 (equivalent to Intel Pentium II) 266MHz processor.

5.5 Conclusions

The model, having been defined for the general case, has been applied to the RCPS problem. There were four stages to the creation of this specific model: defining the project output criteria, defining the data, defining the analytical techniques and defining the project input variables.

The project output criterion was time, and its limit was set as the initial duration of the project plus two time periods. This was to ensure that the optimisation did not permit projects which were too inefficient to be considered, and to permit calculation of the upper limit for the MEST project input variable.

Only three analytical techniques were required. One of these was a pre-scheduling algorithm to calculate the activity duration and resource allocations based on the allocation of the single, primary resource. The other two algorithms were activity selection techniques. The first dictated that activities available to be scheduled at any time point be arranged into a queue ordered by their priority numbers. The second dictated that an activity be delayed to a time period called the minimum early start time (MEST), should it become available before that time period. These two activity selection techniques were vital to ensuring control over how the schedule is arranged.

Three project input variables were defined. The first of these was primary resource use (PRU). This allowed the optimisation to control how resources were allocated to an activity. The limits on this variable were dictated by a simple algorithm. Some

doubt was cast as to whether these limits were realistic, but it was not possible to obtain more realistic values. The second project input variable was activity priority, which allowed the priorities of one activity over another to be varied. The third was MEST, which permitted activities which are in conflict to be moved out of conflict. It was shown that using the priority project input variable would not necessarily include the global optimum solution to the problem, whereas the MEST variable does.

Once the model was defined, the operation of the genetic algorithm was described. Three selection algorithms were identified: fit-fit, random and weighted random. Five replacement algorithms were identified: weakest individual, weakest parent, both parents, random and weighted random. Constraints would be enforced strictly, with unfeasible solutions not being inserted into the population. Crossover and mutation would, however, be repeated for each parent pair in the attempt to obtain two feasible children, up to a predefined maximum number of attempts.

The model and genetic algorithm were used in a computer program which was able to perform the optimisations. The scheduling logic and genetic algorithm were described. Some additional features of the program were also briefly discussed.

Once the model and optimisation had been defined, and the program completed, it was possible to perform the optimisation of the 110 problems. The results of this optimisation are presented and analysed in the following chapter.

Chapter 6 Results of Genetic Algorithm Optimisation

Once the general model for the global approach had been defined, and the model for the RCPS problems derived and coded into a compute program, it was possible to actually perform the optimisation. The optimisations were performed in terms of time, and used the following project input variables: primary resource use (PRU), priority and the minimum early start time (MEST). The results of these optimisations, which include both the global approach and (for use as a comparison) three local approaches, are presented in this chapter.

Furthermore, the results are analysed, and implications for the application and behaviour of the global approach to real projects are assessed.

6.1 Function And Measurement Of The Optimisation

The primary purpose of performing the optimisations of the 110 sample problems was to demonstrate whether or not there was a significant advantage in employing the global approach to project optimisation. The secondary purpose was to identify any characteristics in the optimisation which might have some bearing on the performance of the optimisation of a real project.

Traditionally optimisations have usually been measured by the time it takes them to reach an optimal solution. However, this measure of optimisation performance is not applicable to project optimisation. Finding an optimal solution for a problem as large as the global model of a complex project is not realistic for a reasonable amount of computational effort. Therefore the true measure of the global optimisation is whether it can obtain better solutions than localised optimisation techniques using the same amount of computational effort. Therefore many issues which would be considered within a traditional optimisation, such as when the population of the genetic algorithm begins to converge and what the true value of the global optimum is, are no longer of great importance. In order to verify the validity of the global approach it is essential to be able to compare the performances of the global approach

to local approaches. This comparison must be made at various different amounts of computational effort.

The normal means by which such comparisons would be made is time. All the optimisations would be performed on the same computer and the time taken by the optimisation is taken as a measure of computational effort. For this analysis, however, to compare the optimisations of different projects in terms of the optimisation execution time is not necessarily a valid comparison. Some problems are much larger than others and will therefore take much longer to optimise. This is firstly because the objective function is larger. This means that fewer executions of the objective function can be made in the same space of time. The second reason is that there are more project input variables for the larger problem. This increase in variables makes the problem more difficult to optimise, and more executions of the objective function would usually be required to reach the same closeness to the optimal solution.

Due to these two facts the smaller optimisation will usually be more progressed after a finite amount of time than the larger. Thus to compare the progress of two significantly different sized projects using time is not strictly a valid comparison. However, it is very difficult to determine how a valid comparison might be made. To what extent the increase in model size makes the problem more difficult to optimise is very hard to evaluate. Nevertheless, it is easy to remove the problem of the objective functions taking different times to execute. This can be done by comparing the optimisations by the number of times the objective function has been executed (number of runs) rather than by the time taken. This makes the means for comparison of the optimisations more impartial. However, it will not consider the fact that optimisations become harder as project size increases. This fact should be taken into account in the analysis of results.

In order to achieve this the optimisations were performed over a fixed number of runs. As the optimisation progressed it kept track of the best solution it had found up until that point. The value of this solutions was logged at finite intervals. This would allow the progress of the optimisation to be observed. The value of 10000

runs was selected as the number of runs to execute because any number larger than this would represent a very large amount of computing for a very large project. This will be discussed further in section 6.5.

In order to be able to compare different projects, it was also necessary to evaluate how the value of the objective function for one problem might be compared to another. The project durations which represent the values of the objective function vary considerably over the 110 problems. The initial values found by the scheduler vary between 6 and 98 days. It is clear that in order to compare these it is necessary to identify a measure of fitness of a value of project duration which is independent of this duration. This was achieved by expressing the duration under consideration as a proportion of a benchmark duration for that project. This allowed projects to be compared in a manner which gave no bias to project size. Two benchmark values were considered: the initial duration of the project and the optimum solution for the case where no change in activity duration is permitted.

The initial duration is the duration found when all the activities' priority numbers are the same, the values of MEST are at their initial values and the values of primary resource use are the same as defined in the data set. Thus activities are prioritised in the order in which they appear in the data set. This leads to a duration which can, for some projects, be very long in comparison to the provided optimum for no change in activity duration. When this is the case, the schedule is an inefficient one which results in considerable underutilisation of the available resources. The provided optimum, however, will always be a better solution than the initial value. Its schedule represents the optimal solution for a particular local optimisation, and must therefore make good use of the available resources. Thus this value always represents a good solution of the project and can therefore be expected to be much more consistent from project to project. This consistency makes it a better value to use as the benchmark than the initial duration. Therefore all values of project duration are expressed as this proportion of this value, which will henceforth be referred to as the "target duration".

The 110 problems optimised using the computer program with various combinations of the project input variable types considered, 6 cases in all. These six cases involved 3 cases where only one project input variable was considered as variable – the individual optimisations – and three cases in which a combination of project input variables were considered as variable – the combined optimisations. The individual optimisations are defined in section 6.2, and the combined in 6.3.

6.2 Individual Optimisations

Firstly, three optimisations were performed each with only one of the project input variable types considered as variable, and the other two fixed at their initial values. The purpose of performing these optimisations was to allow the comparison of the global approach to these localised techniques. Furthermore, it would give some indication of how much scope for optimisation was available with each project input variable, and therefore how the global optimisation might be affected by inclusion of that project input variable within it.

After the optimisation had been performed for each of these three cases, the values of the best solution throughout the logged optimisations were converted to proportions of the target value. Once this had been done the average performance of all 110 problems could be calculated for each case at each log point. This permitted the average performance of each technique throughout the optimisation to be performed. This is shown in Figure 6.1.

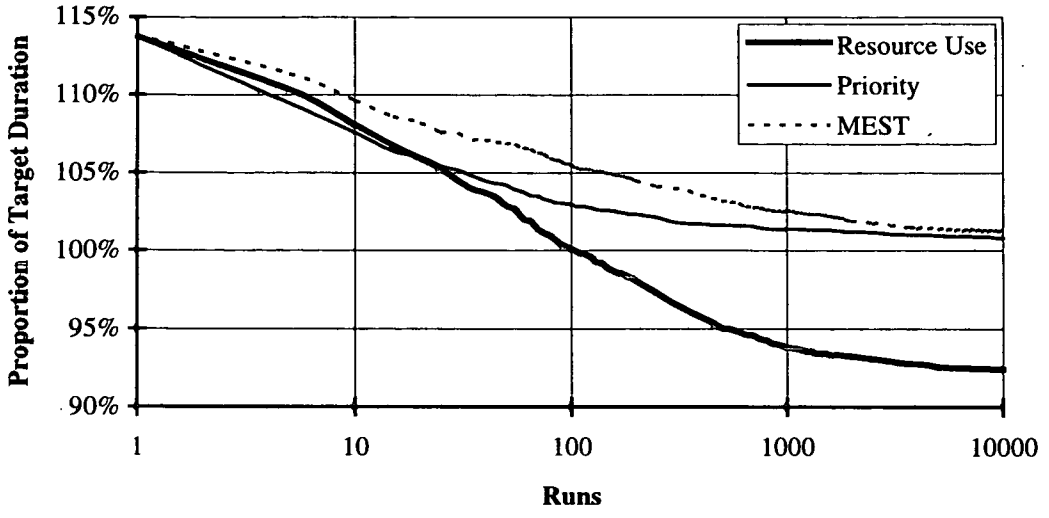


Figure 6.1. Average performance of individual optimisations

The averaged progress results of each of the three cases show that initially the optimisation of the priority case and the PRU case progress at a similar rate, both of them following a reasonably straight line on the log plot until around 20 runs, when the priority optimisation's progress drops off as the mean value approaches 100. The fact that it does not improve on the target duration is to be expected as the optimum value of the priority optimisation (as distinct from the global optimum) for any particular project cannot be lower than the target duration, although it can be greater. The PRU case's progress, on the other hand, doesn't begin to drop off until around 500 runs, suggesting that the optimisation is approaching its own, local optimum, which, from the graph, might be expected to lie at around 92% of the target duration.

The MEST optimisation, unlike the other two cases, appears to follow a much shallower straight line on the log plot than either of the other two, which might be beginning to level out after about 1000 runs, although it is difficult to judge.

In addition to the plot of progress the relative performance of the optimisations of the three cases can be compared using a number of other measures, which are shown in Table 6.1. These measures include the number of problems, as a proportion of all 110 optimisations, for which the optimisation's solution was not bettered by any other optimisation (*Best Solutions*) and for which it was the only optimisation to find

such a good solution (*Uniquely Best Solutions*), the median computational effort required to find the target value – in runs (*Median Runs To Target*) and how often it succeeded in finding that value (*Target Value Reached*). It also shows the mean, standard deviation and median of the final values of the optimisations after 10000 runs.

	Uniquely Best Solutions	Best Solutions	Median Runs To Target	Target Value Reached	Proportion of target at 10000 runs: Mean	Std. Dev.	Median
Resource Use	83%	95%	82.5	94%	92.4%	7.1%	93.3%
Priority	1%	15%	275	79%	100.8%	1.8%	100.0%
MEST	0%	13%	1900	63%	101.2%	1.7%	100.0%

Table 6.1. Comparison of individual optimisations.

As can be seen, the PRU optimisations are much better than the MEST and priority optimisations by all these measures of performance. (Note that standard deviation is not really a measure of performance in itself, merely a measure of variation in performance.)

These values imply that the PRU case can be expected to yield better results in most cases. However, it is possible that this apparent improvement has happened by chance, and that there is no real difference between the PRU case and the other two. This, statistically, is called the null hypothesis, and if it is assumed to be true then a *t*-test can be performed to evaluate the statistical likelihood that the differences in the results have occurred by chance.

In order to perform this test it is necessary to compare the differences between PRU & priority optimisation and the PRU optimisation for each problem. If the best value found by the PRU optimisation for any problem i is X_i, and the best value of the PRU & priority optimisation Y_i, then difference $W_i = (X_i - Y_i)$. The *t*-test is then performed using the mean ($\overline{W}$) and standard deviation S_W to obtain the test statistic T:

$$T = \frac{\overline{W}}{S_W / \sqrt{n}}$$ where n is the number of samples (110 in this case).

When the null hypothesis is true this value of T will follow a t distribution for n samples. Therefore it is possible to assert with what confidence the null hypothesis will be rejected, if it were to be rejected.

The values of $\overline{W}$, S_W and T for the t tests of both priority and MEST against PRU are shown in Table 6.2, along with the raw proportions of how many times the priority and MEST cases were better, the same as or worse than the PRU case.

	Better	Same	Worse	Mean W	Sw	T
Priority	5%	10%	85%	8.37%	6.90%	12.72
MEST	4%	10%	86%	8.77%	6.76%	13.60

Table 6.2. Comparison to PRU optimisation.

The values of T are very high. In order to reject the null hypothesis with more than 99.5% confidence for a sample size of 110 the value of T must be at least 2.63[3]. As the values of T obtained are much higher it is possible to say that the performance of the PRU cases is significantly better than both the priority and MEST cases.

This fact is also consistent with all the other measures of performance in Table 6.1, and was anticipated due to the large scope for optimisation introduced in the setting of the limits. However, the performance of the priority case also performs better than the MEST case by every measure. Indeed, the optimisation of the MEST case is very slow in comparison, and even though the optimisation of the priority case has slowed down considerably by 10000 runs, the optimisation of the MEST case has not caught up.

6.2.1 Performance Of MEST Compared To Priority

Given sufficient optimising effort, the optimisation of the MEST case should give the target duration for any project, while for the optimisation of the priority case this is not always so. This means that the best values in each feasible region for the MEST case are, on average, better than those in the feasible regions of the priority case. Despite this fact the mean performance of the MEST case only comes close to the

[3] From Hogg & Ledolter[Hogg, 1992], p449 by interpolation

priority case once the progress of the optimisation of the priority case has begun to drop off because it is approaching its minimum value, and even at 10000 runs the performance of the priority case is significantly better than the MEST. This can be seen in Table 6.3.

	Better	Same	Worse	*Mean W*	*Sw*	*T*
MEST at 1000 runs	7%	58%	35%	1.11%	2.84%	4.091
MEST at 10000 runs	11%	64%	25%	0.40%	2.10%	1.991

Table 6.3. Comparison of the MEST case to the priority case.

The minimum values of T required for rejection of the null hypotheses are, for n=110, 1.658, 2.36 and 2.62 for 95%, 99% and 99.5% confidence levels respectively[4]. The values of T for the MEST case are both greater than the 1.983 required for rejection with 95% confidence.

This can also be demonstrated by considering how frequently the optimisation was able to obtain the target value, and the median number of runs required to reach it, both of which can be found in Table 6.1. As can be seen, the MEST optimisation succeeds in finding the target value in only 63% of cases, which is less than the 79% achieved by the priority optimisation. The significance of this value can be evaluated by performing a test on the difference of proportions, as follows.

First the null hypothesis is accepted. Then, using the values of the observed the two proportions $\hat{p}_1$ and $\hat{p}_2$, it is possible to calculate the test statistic Z by dividing the difference between the two values by the true standard error of the problems. In order to do this the true mean of the population is assumed to be:

$\hat{p} = \frac{\hat{p}_1 + \hat{p}_2}{2}$ provided that sample sizes are the same.

[4] From Hogg & Ledolter[Hogg, 1992], p449 by interpolation.

Therefore the test statistic Z is defined by:

$$Z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{2\frac{\hat{p}(1-\hat{p})}{n}}}$$ provided the sample size n is common to both samples.

The value of $\hat{p}$ for this problem was 71.1%, and the corresponding value of Z 2.55. This means that the confidence with which the null hypothesis can be rejected is 99.5%[5] - from the normal distribution. This shows that it can be asserted with 99.5% confidence that the null hypothesis is false, and the difference between the performance of the two optimisations is significant.

A comparison of the median number of runs taken to reach the target (a mean value is impossible to obtain because the target value was not found at all in some cases) is also consistent with the hypothesis that the MEST optimisation is operating very slowly.

In order to assess why the performance of the MEST case is not as good as that of the priority case, it is necessary to consider how the MEST project input variable affects the project's duration. The purpose of the MEST project input variables is to delay certain activities and hence take them out of resource conflict with other activities. However, whether or not the activities delayed are necessarily in conflict with each other is not explicitly considered in the scheduling as it is with activity priority. Therefore it is perfectly feasible to delay an activity far more than is necessary, which in turn delays the activities which succeed it by precedence constraints, and hence delay the project. This is particularly the case when an activity is one executed early in the project.

As the number of activities increases, the expected number of activities whose MEST project input variables will be mutated as part of the reproduction stage of the genetic algorithm will increase. As the selection of the new value of any project input

---

[5] From Hogg & Ledolter[Hogg, 1992], p447.

variable selected for mutation is random, there is a reasonable probability that the new value of the MEST will be high. If the activity in question occurs early in the project's execution then the setting of this new, high, value will have a high probability of leading to an increase in the project execution time. This means that a variation in a single value of MEST project input variable can be highly detrimental to the project's execution time. This high probability of creating highly detrimental mutations will make it more difficult to find the target duration, or even good solutions, with the MEST case. As this probability of creating a highly detrimental solution exists for each MEST project input variable, this difficulty should also increase greatly with project size.

The tendency of MEST project input variables to generate poor solutions in the optimisation can be shown by considering the number of runs for which a feasible solution was obtained (one which did not violate the constraint on the project duration). This number of feasible runs can be expressed as a proportion of the total number of runs performed. This value, which will henceforth be referred to as the "feasibility rate", can be useful in determining how often the optimisation generated unfeasible solutions. The higher the value of the feasibility rate, the fewer unfeasible solutions were generated by the optimisation.

The mean feasibility rate associated with the priority case was high (96.3%), whereas the value for the MEST case was much lower (82.9%). This shows that use of MEST as a project input variable will usually lead to a large number of poor solutions, which is detrimental to the optimisation's progress.

The fact that this will increase in time can also be demonstrated by considering Figure 6.2, which plots all 110 problems' feasibility by number of activities, along with a trend line calculated by the method of least squares.

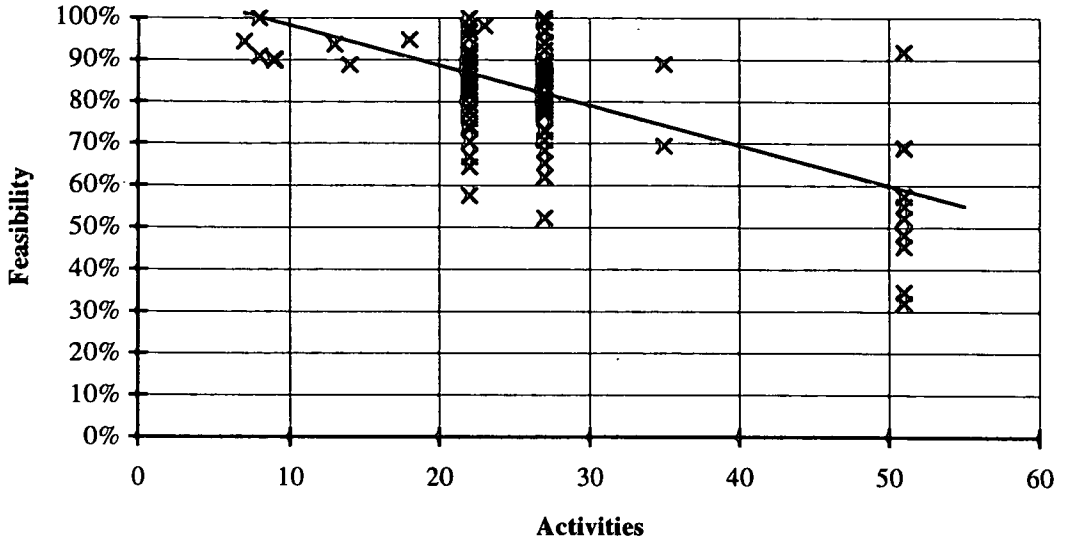


Figure 6.2. Feasibility against activities for MEST optimisations.

This graph has a lot of scatter, but nevertheless the trend is clear: as the problems become larger they suffer a significant decrease in feasibility. This corresponds to the increasing difficulty for the optimisation to find good solutions as the project size increases.

The deterioration of the optimisation of the MEST case's ability to find the target duration can also be shown more explicitly by grouping the activities by size. The projects in the data set can be divided into four clear groups. The first is projects 1-15, which are small activities of differing sizes with an average of 17.8 activities. The second, third and fourth groups, which consist of projects 16-57, 58-100 and 101-110 respectively, are all groups of project with the same number of activities: 22, 27 and 51 respectively. The number of times the target duration was found, expressed as a proportion of the whole group, is plotted in Figure 6.3.

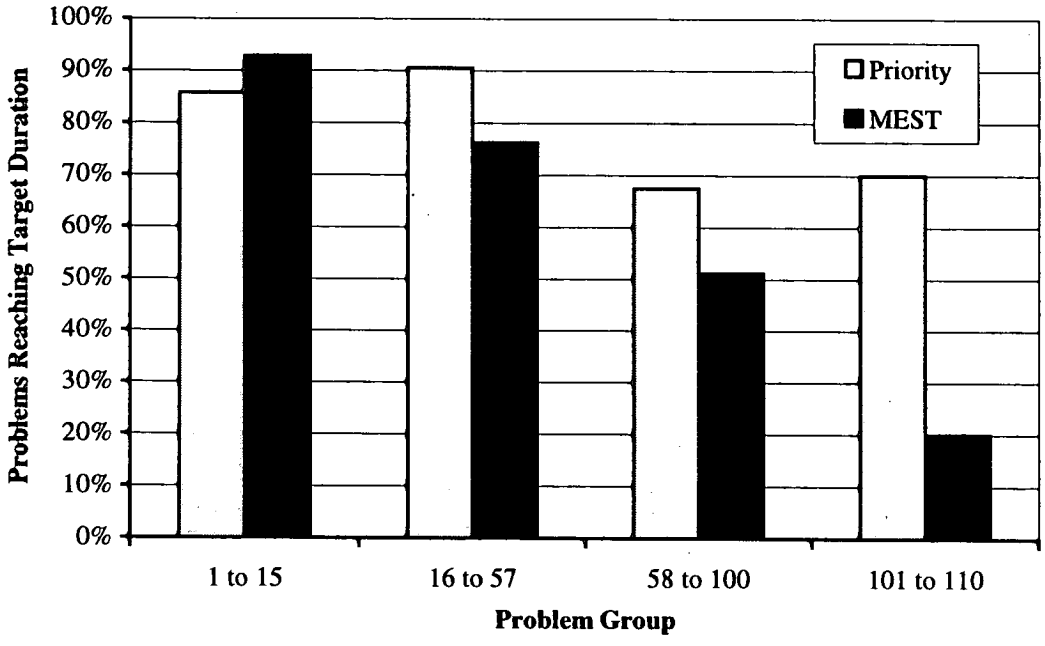


Figure 6.3. Target duration reached by group for MEST and priority.

As this chart shows, as the problems become more complex the optimisation of the MEST case will become very much less likely to find the target duration. This reduction is significantly higher than the corresponding reduction for the priority case, whose reduction is observable but not as marked.

So despite the fact that there exists, for every project, a set of MEST project input variables which would correspond to the target duration (the optimum for the case where there is no change in duration) it is still outperformed by the priority optimisation, for which no such set always exists.

This shows activity priority to be a much more efficient means of controlling the order in which activities are executed, although this does come at the price of occasionally excluding the target value from the scope for optimisation, as with problem 1 from the data set. In the optimisation of the problems the optimisation of the priority case was still successful in identifying the target duration in 79% of cases, and progressed much more quickly than the MEST case. In addition to this the performance did not diminish rapidly with project size, as did MEST.

This could represent a weakness in the global approach, where applied in its purest form. If a global approach uses MEST in preference to priority then it will be able to consider every possible configuration of activities, whereas if it uses activity priority then it cannot. Thus using activity priority could involve rejecting a small but possibly significant part of the scope available for optimisation of the problem. This rejection of some of the scope for optimisation goes against the philosophy of the global approach. Therefore the rigid application of the global approach involves implementing MEST project input variables, which is known to be detrimental to the optimisation's performance.

The importance of these facts will now be assessed by considering the application of the global approach to the problem.

6.3 Combined Optimisations

In total three further cases were optimised. Firstly, PRU and MEST were both considered as variable. This corresponds to the true global approach. The optimisations did not yield results which improved upon PRU, so priority was added to the optimisation and all three optimised together. While this did not constitute expanding the feasible region at all, it was hoped that this method might help the optimisation to find better solutions by providing the scheduling algorithm with priorities for the best method of scheduling a near optimal solution which still contained resource conflicts.

However, this also did not produce any significant improvement on the PRU case, so in the final case priority and PRU were considered as variable within the optimisation. The mean progress of the optimisation of all these three cases is shown in Figure 6.4 along with the PRU case.

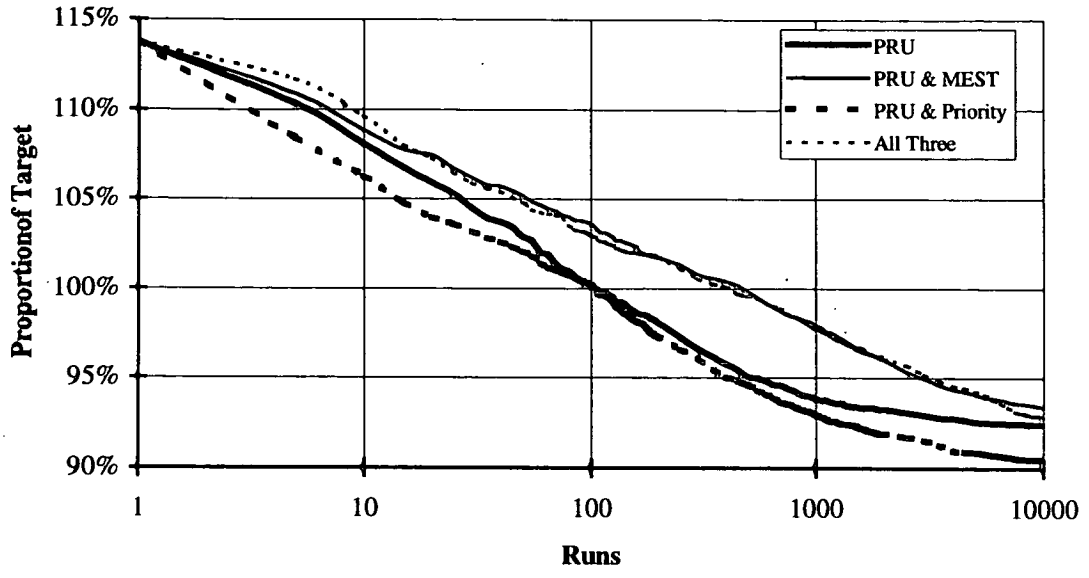


Figure 6.4. Combined optimisation performance.

This combined performance shows how, on average, the PRU and priority case achieves the best mean performance, even at lower values of computational effort. The PRU and PRU & priority cases follow an approximately straight line on the log plot, with PRU & priority being just slightly better until the PRU case begins to drop off at around 500 runs. The PRU & priority case also appears to begin to drop off at this point, although less markedly so than the PRU case.

The two optimisations involving the MEST project input variables do not perform as well as either the PRU or PRU & Priority cases. This was anticipated from the behaviour of the optimisation where MEST alone was variable.

Once these three cases had been optimised, it was also possible to compare all six cases. These are considered in terms of the measures of performance used to compare the individual optimisations, and are shown in Table 6.4.

	Uniquely Best Solutions	Best Solutions	Median Runs To Target	Target Value Reached	Proportion of target at 10000 runs		
					Mean	St. Dev.	Median
PRU	4%	43%	82.5	94%	92.4%	7.1%	93.3%
Priority	0%	6%	275	79%	100.8%	1.8%	100.0%
MEST	0%	7%	1900	63%	101.2%	1.7%	100.0%
PRU & MEST	0%	34%	300	94%	93.4%	6.6%	94.3%
PRU & Priority	33%	92%	72.5	100%	90.5%	6.4%	91.1%
Combined	1%	44%	325	96%	92.8%	6.5%	93.1%

Table 6.4. Comparison of all the cases.

As can be seen, the mean performance of the PRU & priority case appears to be significantly better than all the others. It achieves best solutions more often, reaches the target value more often and more quickly than the other solutions, and also performs better by its mean value over 10000 runs. However, it is necessary to demonstrate that these values are statistically significant, and not just down to the chance high performance of certain optimisations. Therefore the significance of the differences between these cases will be evaluated. This will be done firstly by comparing the best final durations and then by comparing their performance in realising the target value.

6.3.1 Comparison Of Optimisations By Best Final Duration

The first means of evaluating the significance of the apparent superiority of the PRU & priority case is by performing a *t*-test on the solutions against the other two combined optimisations and the best of the individual cases, the PRU. The values for the test are shown in Table 6.5, along with how often these optimisations were worse than, the same as or better than the PRU & priority case.

	Better	**Same**	**Worse**	**Mean W**	S_W	T
PRU	6%	39%	55%	1.98%	2.62%	7.93
PRU & MEST	3%	35%	62%	2.95%	3.64%	8.50
All Three	4%	43%	53%	2.34%	3.59%	6.83

Table 6.5. Comparison to PRU & priority case.

All the values of T are much more than the value of 2.62 required to reject the null hypothesis with 99.5% confidence. Therefore it is possible to assert that the PRU & priority case does perform significantly better than any of the other three.

This assertion also explains the fact that the comparisons to the PRU & priority case also show that for most projects the PRU & priority case will perform at least as well as another of the cases, and better than it with more than half the projects.

As well as showing that the PRU & priority is the best optimisation, it would also be useful to know if the inclusion of priority into the true global optimisation, the PRU & MEST case, would be expected to yield an improvement in the value of the best duration found. From Table 6.4 this would certainly appear to be the case. The significance of the mean values can be tested using a *t*-test. The important values for the comparison are shown in Table 6.6.

	Better	Same	Worse	Mean W	S_w	T
PRU & MEST	14%	52%	34%	0.61%	2.20%	2.91

Table 6.6. Comparison to the optimisation of all three variables.

The value of T is greater than 2.62. Therefore the null hypothesis can be rejected with greater than 99.5% confidence. Therefore there is a significant improvement of the average performance of the PRU & MEST optimisation by the addition of priority to it.

6.3.2 Comparisons By Realising Of The Target Value

The optimisations have so far been compared by their final values, which, as has been discussed at the start of this chapter, is not a completely fair means of comparison because the optimisation should require more runs as the model becomes more complex. Therefore, another means of comparing the projects might be on the basis of how quickly the optimisation found a certain good solution to the problem. Indeed this is probably of more value to the project planner, because they are not necessarily looking for an optimal solution, but a near optimal solutions which can be found using a reasonable amount of computational effort.

The only such good solution which has been defined for every project is the target duration. Therefore all the techniques can be measured in terms of how often they reach this target value, and the median number of runs required to find it. These values can be found in Table 6.4.

The best optimisation at reaching the target value within 10000 runs was the PRU & priority case, which managed to succeed in 100% of cases. Its closest rival was the optimisation using all three project input variables, which managed to find the target duration in 96% of cases. In order evaluate the significance of this it will be necessary to assume that there is not difference (the null hypothesis) and calculate the Z value for the difference of proportions.

The value of the mean proportion $\hat{p}$ for this problem is 0.98 (98%). Therefore the value of Z is 2.03. Using a normal distribution this permits rejection of the null hypothesis with of 97.9%[6] confidence. Therefore, it is possible to assert that there is a significant difference between the two results: There is a significant improvement in the number of times the PRU & priority case will find the target value over the combined case.

It can also be said that optimisations of the other problems reach the target value significantly less than the PRU & priority case.

It is also possible to perform this analysis for the two optimisations which used PRU & MEST project input variables. The PRU & MEST optimisation succeeded in finding the target value in 94% of cases, while the corresponding value for the optimisation of all three was 96%. This yields a value of $\hat{p}$ of 95%, and Z of 0.65. The minimum value of Z even for 80% confidence of the null hypothesis being false is 0.84. Therefore the null hypothesis can be accepted, which means that, unlike for the mean performance over 10000 runs, the observed difference between the number of times the target duration was found for the two cases which used PRU & MEST project input variables was not significant.

However, this particular use of the target duration is not wholly independent of runs. Just because the optimise failed to find the target duration over 10000 runs does not necessarily mean that the target value could not have been found with increased computational effort. Indeed, for the two optimisations which use MEST it is known

---

[6] From Hogg & Ledolter[Hogg, 1992], p447.

that the target duration must exist within the feasible region. Therefore these values are still dependent upon the number of runs over which the optimisation is performed.

For this reason a second measure of the performance of the optimisation against the target value is introduced: runs to target. This value measures how many runs were required to reach the target duration for each optimisation. The median of these values shown for all optimisations in Table 6.4.

The reason that the median is used for these values, rather than the mean is that the number of runs to the target value is not known for all the problems. When the optimisation fails to find the target durations it is either because the number of runs required to find the target duration is larger, or because the target duration does not exist within the feasible region. It is only possible to find the mean where all the values are known. In this case not all values are known, but more than half are known and it is also know that the unknown values are larger than 10000 runs (which includes infinity - for the case where the target duration does not exist within the feasible region). Therefore the median can be found.

This value of this median shows PRU & priority to have yielded the best performance, followed very closely by PRU. However, this median does not lend itself to any kind of significance testing, and so another measure of performance needs to be found.

The ideal test to perform would be a *t*-test. However, it is not possible to compare all problems in the sample by this means because the value is unknown for some problems. The closest it is possible to get to comparing all problems is to compare only those solutions which have a value of runs to target for both problems under comparison. Thus a mean, standard deviation and *T* value can be obtained from this slightly reduced sample. These results are shown in Table 6.7.

	Mean W	S_W	T
PRU - PRU & priority	13	378	0.351
PRU & MEST - PRU & priority	690	1405	4.981
All 3 - PRU & priority	832	1899	4.490
All 3 - PRU & MEST	3	1093	0.027

Table 6.7. Comparison with PRU & priority by runs to target.

As this shows quite clearly, the improvement of the PRU & priority case over the PRU case is small, and not significant, although its improvement over the other two combined cases is significant. Also, the improvement obtained by adding priority to the PRU & MEST case is very small and also not significant.

Although these results do not show any significant difference between the PRU and PRU & priority cases, it should be remembered that certain solutions were ignored because the value of the PRU's runs to target could not be calculated. Therefore the results show that for those projects for which the target value was found by the PRU case within 10000 runs there was not a significant difference between PRU case and the PRU & priority case's number of runs required to obtain the target value.

If the target duration exists within the feasible region for the PRU case for some or all of the projects for which the target duration was not found, then the number of runs required to reach it is an unknown value which is definitely larger than 10000. In this case, the exclusion of these values would invalidate the previous *t*-test, because the measure of the significance of the difference between the values has excluded the 4 greatest differences between the two cases. The result of no significance would not take into account the fact that in 4% of cases the difference between the two cases was so high it could not be taken into account. Therefore it is useful to see whether the difference between the two cases would be significant if these values were to be considered.

The effect of this can be evaluated by assuming that all the PRU optimisations would be able to find the target duration. If this were the case then the minimum number of runs to target for those projects for which the target value was not found is 10001. Therefore if a value of 10001 is used as the value of the runs to target of the PRU case for that optimum then the *t*-test can be repeated. This yields a value of T of

2.516. Therefore, the null hypothesis can be rejected, and there is a significant difference between how quickly the optimisations of the PRU case and the PRU and priority case find the target duration assuming the target duration exists within the feasible region for all the PRU cases.

The second analysis is based on an assumption whose verity is uncertain. However, it does compare the results of all the projects, taking into account the fact that the PRU case was not always able to find the target value within the allotted number of runs. This is not taken into account by the first *t*-test. Therefore the results of the second *t* test will be used, which state that there is a significant difference between how quickly the optimisations of the PRU and PRU & priority cases find the target duration.

The results of the comparison between the PRU & MEST optimisations with and without priority have a stronger case. While it is true that not all results are known, the difference between the mean values is very small, and the corresponding *T* value tiny. If the analysis is repeated with the assumption that non-values will be 10001 runs then the value of *T* is 0.0077. The assumption that the target duration exists is a valid one, because the target duration will always exist for any case which contains MEST. Nevertheless the values have been assumed to take the lowest possible value. If the values tended to be higher then the optimisation of the case which contains all three variables would not compare as well as it does.

6.4 Factors Affecting The Optimisations

Overall the comparisons of all the solutions have demonstrated that the PRU & priority optimisation yields significantly better results than the other cases for these 110 problems by several measures (though not by all evaluated). However, the purpose of these optimisations was to test the feasibility of the global approach to optimisation on small sample problems and to try to extrapolate from the results how global optimisation might perform on a real project. In order to do this it is necessary to begin to evaluate some of the characteristics of the sample projects which affected the performance of the optimisations.

Perhaps the most obvious difference between these projects and real projects is size. The largest problems in this data set were of 51 activities. A real project could be up to 100 times that size. Perhaps the easiest way of comparing the difference in performance of the optimisations over different project sizes is to divide the projects into the four groups defined in section 6.2.1. If the mean proportion of the optimum is evaluated over all these groups then it will yield a simple indicator of how the optimisations performed over differing sizes of project. This is shown for all the combined and PRU cases in Figure 6.5.

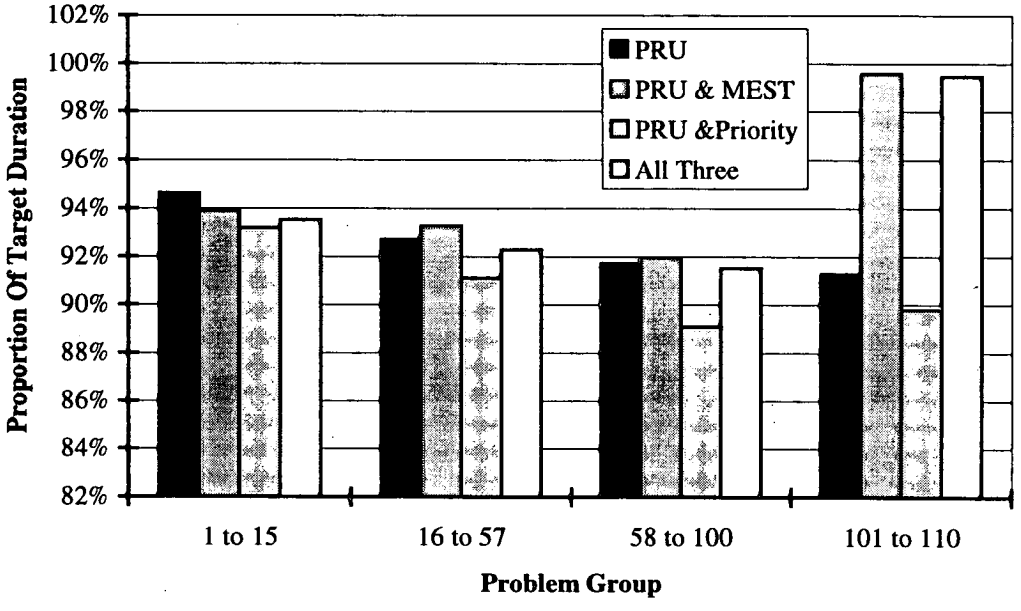


Figure 6.5. Performance by problem group

As an be seen from Figure 6.5 all four of the project input variable configurations improve as the complexity of the projects increase, with the exception of problems 101-110 which show a marked deterioration for the problems involving MEST. It appears from this that the scope for optimisation beyond the target duration actually expands as the project increases in size.

This apparent increase in the scope for optimisation would be explained by there being more good solutions around the optimum, or there being more optima (or both). If this were the case then the probability of the optimisation finding a near

optimal solution would be greatly increased. This means that more solutions would lie closer to the optimum, and therefore the standard deviation would be reduced. Figure 6.6 shows how the standard deviations vary with problem size for PRU and the three combined cases, and it can be seen from this figure that that the standard deviation is reduced with increasing size for all the cases.

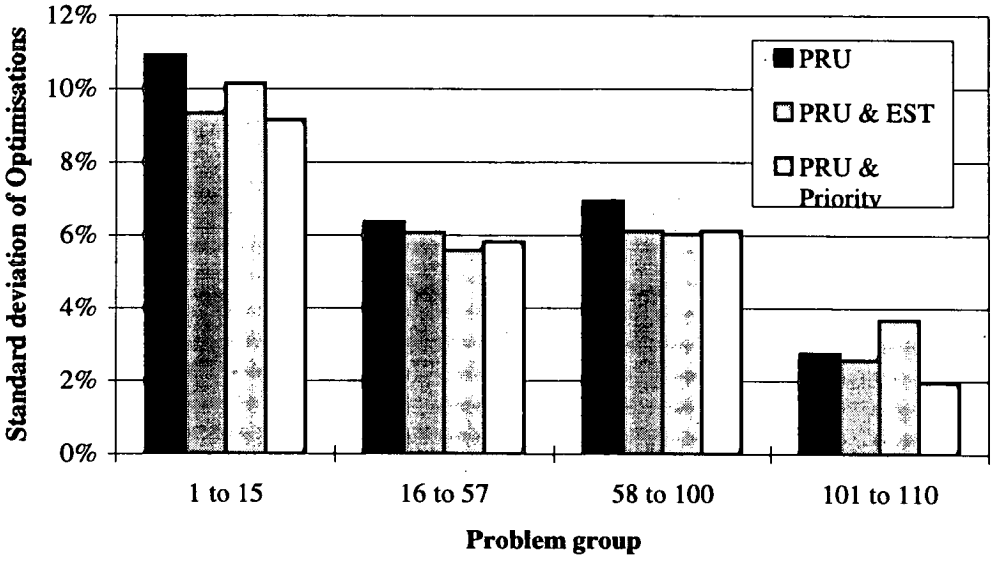


Figure 6.6. Standard deviations of combined results by group.

While it is true that if optimal or near optimal solutions are becoming easier to find then the standard deviation will be reduced, the converse is not necessarily true. There are other possible causes. It is possible that there is some kind of averaging effect emerging in the larger projects. This can be explained by considering the fact that within a project there are a number of local systems of activities which may be arranged in a number of configurations.

Each of these systems has a number of configurations which will usually be detrimental to the optimisation, and a number of configurations which will not be as detrimental. It would also be expected that these subsystems will vary greatly, and the number of detrimental configurations may be much higher in some particular subsystems than others. In a small project, with only a small number of these subsystems, optimisation usually has to find efficient solutions for all of these

subsystems in order to find a good solution. If an efficient solution to one of these subsystems is very difficult to find, or does not exist at all, then the value of the optimisation will be significantly affected. Similarly if very efficient solutions exist and are easy to find then the optimisation will also be significantly affected. However, in a large project the occurrence of one of either of these two types of subsystem will have a reduced effect on the optimisation whole. Also, each project would be expected to contain a more representative spread of subsystem behaviour, so the spread in performance of the optimisations will be reduced.

This alternative explanation is consistent with the reduction in standard deviation, which is illustrated by Figure 6.7, which shows the final performance of the PRU case against the number of activities. It also shows a linear trend line, fitted by the method of least squares.

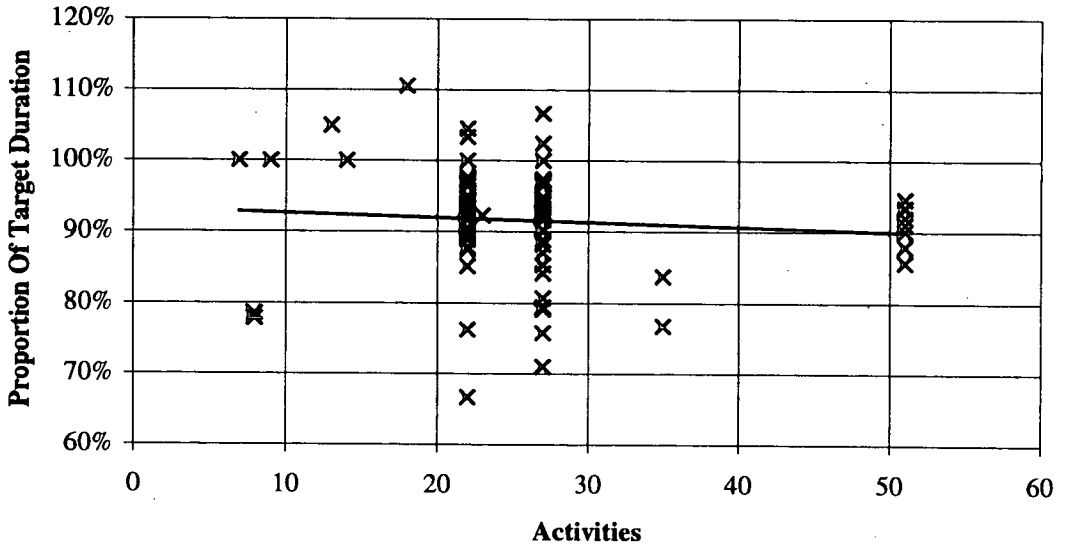


Figure 6.7. Performance of PRU case against activities.

What this figure illustrates slightly more clearly is that the reduction in standard deviation (which can be seen from the fact that the spread of values converges as project size increases) means that the number of very low results is also reduced as the number of activities increases. If the change in performance were caused only by an increase in the ease with which the optimal solution may be found then the

optimisations would all be expected to be at very low values, which is not the case. The problems are grouped around a higher value. Conversely, the averaging effect discussed, while it explains the decrease in spread, does not explain the slight downward trend of the results. However, the results can be explained by considering both effects as operating at the same time. It may even be possible that they are interconnected. This is now investigated in more detail by considering factors effecting the PRU case.

6.4.1 PRU Case

The observation of the consistent increase of the performance of the PRU case as the problems become increasingly complex, which can be made from Figure 6.5, is the reverse of what might be expected. An increase in complexity would usually be expected to lead to the optimum value becoming more difficult to find. There are two possible explanations for this phenomenon: either the optimum for PRU is becoming more easy to find, or the optimum for the PRU case is actually lower as a proportion of the target value, with larger projects.

For the optimum to become more easy to find, the increase in size in the model must somehow increase the number of good solutions around the optimum. Alternatively, as integer durations are being used, it is possible that an increased number of optimal solutions exist. This is possible if there is a global minimum duration with a number of possible combinations of values of PRU project input variable which yield this duration.

Both this possibility and the possible lowering of the optimal solution to the case could be caused by the unrealistic increase in the scope for optimisation arising from the broad limits set on the PRU project input variable. This unrealism has already been discussed briefly in section 5.2.4.3.1. The setting of the limits on the PRU project input variables may cause a far greater expansion of the scope for optimisation than might be expected for a real project. The setting of limits may be realistic for small projects where the availability of resources is not usually much more than the initial allocation, it is not necessarily so for larger projects. In larger

projects more activities which use the same resources tend to be executed in parallel, with a high availability in comparison to the allocations to the individual activities. Thus the limits set on the project input variables by the method used for the PRU project input variable could be very high. If this were so they allow activities to be compressed or stretched far more than may be expected for a real project. This could introduce a large number of possible combinations of activity resource allocations which perform well using either very compressed activities or very stretched activities.

The first way of evaluating if this is a real phenomenon, and if so what its impact might be, is by considering by how much the activities may be compressed or expanded for any particular project. The variability of activity durations has already been discussed in section 5.2.4.3.2. It showed that there was a general trend of the variability of the activities to increase with increasing project size, which was shown in Figure 5.1. While this value of variability is useful for evaluating the realism of the model, it can be analysed in further detail by being broken down into its two component parts. This could help to evaluate its impact the optimisations.

There possible effects of the PRU variable on the activity's duration are twofold. Firstly, an increase in the resource consumption results in the compression of the activity, which allows earlier completion of the activity and hence earlier commencement of its successors. However, it can also lead to the delaying of activities which would previously have been scheduled at the same time because there are insufficient resources. The second effect is the increase in duration by the reduction of the PRU. This makes it consume fewer resources during its execution, leaving more free for other activities to be scheduled at the same time, which could result in a reduced duration. However, this effect can also result in the delay of subsequent activities, and hence the project.

The variability of the activities may therefore be considered by two different methods, the mean compressibility and mean expandability of the activities. These values express by how much, on average, a given project's activities may be compressed or expanded from their initial durations. Compressibility and

expandability are shown in Figure 6.8 and Figure 6.9 respectively, plotted as a proportion of the original durations against the number of activities, with linear trend lines estimated by the method of least squares.

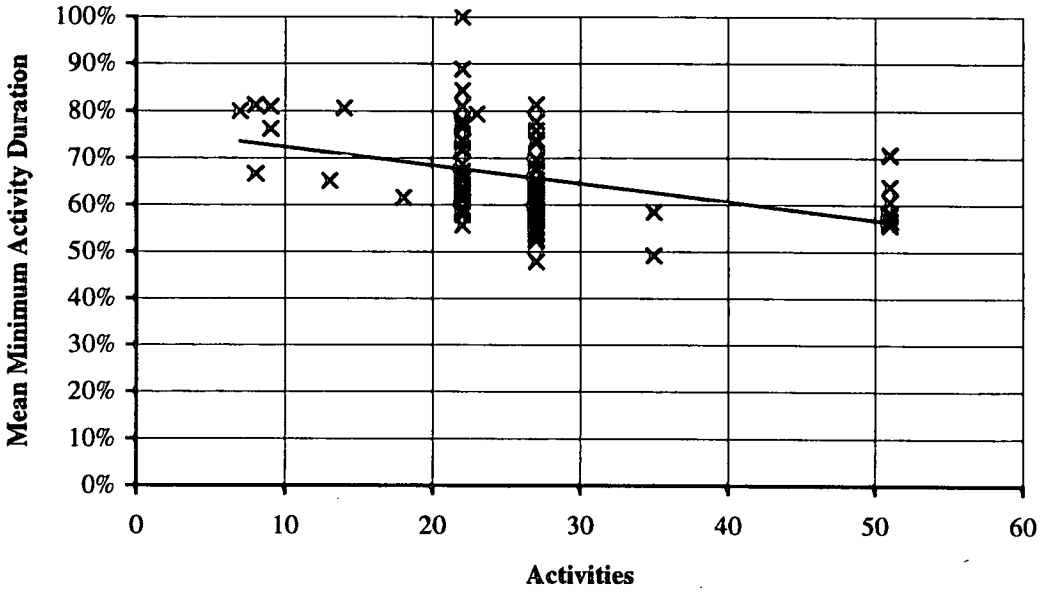


Figure 6.8. Mean compressibility of activities by project size.

As Figure 6.8 shows quite clearly, the mean minimum duration of activities tends to decrease as the project size increases. This shows that the amount by which an activity may be compressed tends to increase as the size of project increases. This is a direct result of the upper limit for the PRU project input variables being higher for larger projects. There are two factors that this might be attributed to: the increase in the size of the resource, and increase in the mean duration of the activities. The increase in the mean duration is a possible factor because an activity cannot be compressed to a smaller duration than 1. If this were to become the limiting factor then as activity duration increase it will be able to be to be compressed to a greater degree before reaching its absolute minimum value of 1.

Whether either of these factors would cause an increase in the compressibility with increasing project size with a large, real project is questionable. In a real medium sized or large project the setting of the upper limit would be based entirely on the

nature of the activity. Only in very small projects might the resource ceiling be the limiting factor on the maximum resource allocation. The resource ceiling will be increased as more activities are required to operate in parallel. Therefore increasing the resource allocation to an activity will be restricted by physical constraints on the activity long before it reaches the resource ceiling. Similarly changes in activity duration would also not be expected with variations in project size. An increase in the size of a project tends not to be characterised by an increase in the size of the activities, but by an increase in the number of activities. This is because projects tend to be modelled to similar levels of detail, regardless of their size. Therefore the nature of the activities within a project, and hence the compressibility of the activities, should not change with project size.

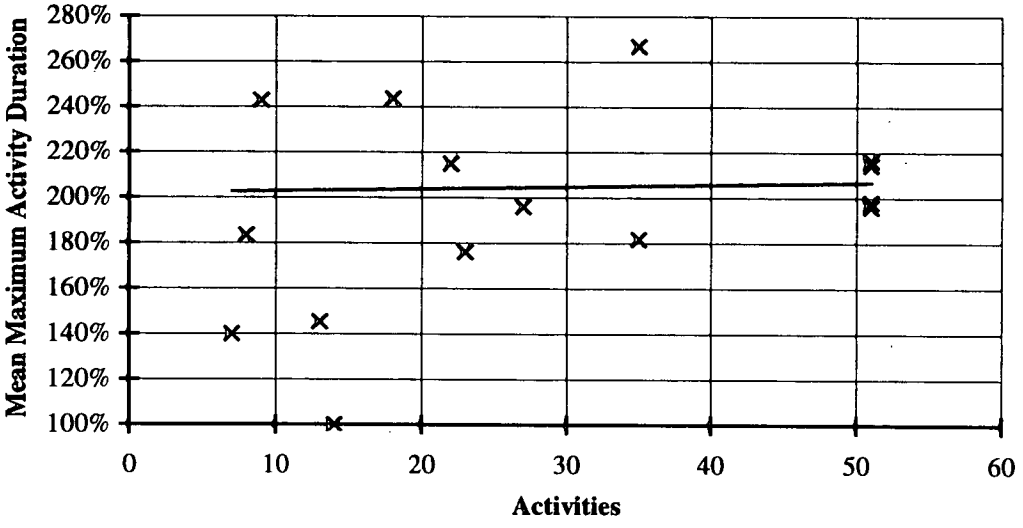


Figure 6.9. Mean expandability of activities by project size.

While the compressibility of the activities is seen to increase with an increase in size of the sample projects, the mean expandability, shown in Figure 6.9, does not really increase very much with project size. As this is related to the initial resource allocation, it shows that the average activities' resource allocations are remaining fairly constant. This is consistent with the expected trend for real projects, which is that there should be no real change in the average behaviour of activities.

This shows that the increase in the average variability by sample project size is caused by the increase of the compressibility of the activities, which would not be expected in a real project. If this phenomenon is the cause of the improvement of the performance of the PRU case, then it must be concluded that this improvement with increasing project size cannot be expected with a real project. However, in order to make this assertion it is necessary to evaluate whether this increase in activity compressibility is responsible for the increase in performance.

If this really is the cause, then it would be expected that the solutions provided by the optimisations would become more efficient at using resources as the project size increases. The aim of the optimisations is to reduce the duration of the project. As the total amount of each resource required is fixed for each activity, the total amount required for the whole project can be calculated. Therefore there is an absolute minimum value of project duration, which is equal to the total resource requirement divided by the resource ceiling, as any further reduction could not be achieved without violating the resource constraints. If the increasing performance is caused by the increase in project compression then this increase in performance must be achieved by being able to obtain more efficient solutions.

Where there are multiple resources an absolute minimum project duration, the total resource requirement divided by the ceiling, may be calculated for each resource. The minimum for the project is then the greatest of these values, because a lower value of duration cannot be obtained without violating the resource constraints. If this value, rather than the target duration, is used as the benchmark, then it is possible to see how efficiently the solutions found use resources. This is shown plotted against the number of activities in Figure 6.10, with a trend line fitted by the method of least squares.

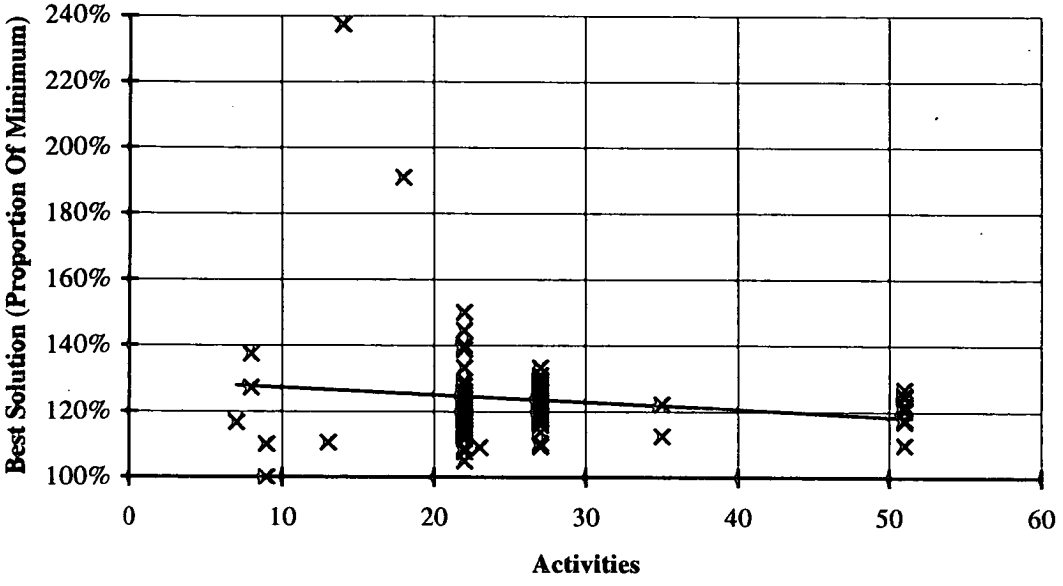


Figure 6.10. Adjusted performance of PRU case by project size.

As can be seen from this figure, there is a slight trend for the performance of the optimisation to move closer to the minimum value with increasing project size. This suggests that the optimisation of the PRU case is able to find more efficient solutions as the project size increases. This is as would be expected if the increased compressibility of the activities were to result in the optimisation being able to find more efficient solutions. However, there are two results which are very far from the rest of the results, and which would therefore have a considerable effect on the slope of the line. These are projects 1 and 9, whose values of best solution are 238% and 191% respectively.

The best solution is very high for project 1 because there are a large number of activities which have no resource requirement. The three resources are also not evenly distributed. Some are only used early in the project, and some only late in the project, and there are very few situations where the delaying of activities is required. In addition to this, the resource ceilings are very low and do not allow for much activity compression for those activities which do have a resource requirement. This leads to a very low value of minimum duration compared to what the lowest value in the optimisation's feasible region may be. As other projects do not display such

characteristics, it would be helpful if this problem were removed from the consideration of the adjusted performance.

The low performance of Project 9, on the other hand, cannot be explained in this way. Almost all activities are allocated resources, and the ceiling of 8 for the only resource is much higher than the average allocation of 2.3. There should therefore be considerable scope for optimisation. The reason the value or the PRU case is so high is that the project starts with quite an inefficient sequence of activities, which needs to be corrected by activity priority. This means that the PRU case is only able to obtain a duration of 21 days, 2 days higher than the target duration. However, the addition of the priority to the optimisation still only yields the target duration. What is perhaps most likely is that that the high value of this particular project's duration is simply due to the high scatter of performance associated with smaller project sizes, as defined earlier, in section 6.4. Therefore there is no reason for its not to be included in the analysis.

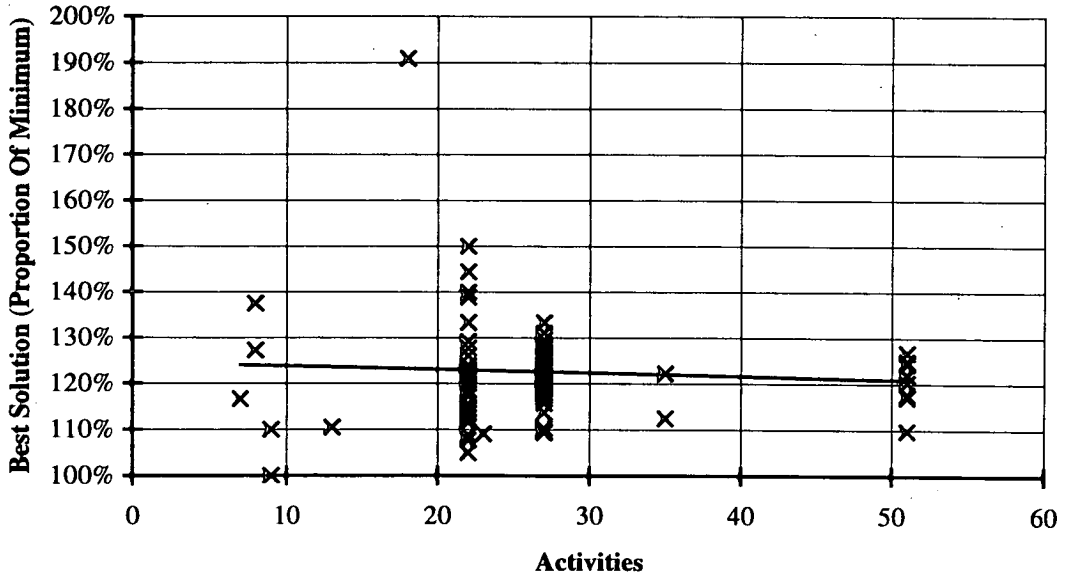


Figure 6.11. Adjusted performance of PRU case by project size, excluding problem 1.

So if project 1 is not included in the plot of adjusted performance against project size then the new trend can be seen in Figure 6.11. Project 1 will also be omitted from all

future plots which consider the minimum duration. The corrected trend is not as marked as the former trend, and it is difficult to say whether the trend is valid, as there is a lot of scatter in the plot. The correlation between the two values is certainly not a strong one.

While an analysis of how much better the optimisation performs against minimum duration by project size is significant, it would be better to compare the effect of changes in the compressibility directly to the minimum value. This is plotted in Figure 6.12.

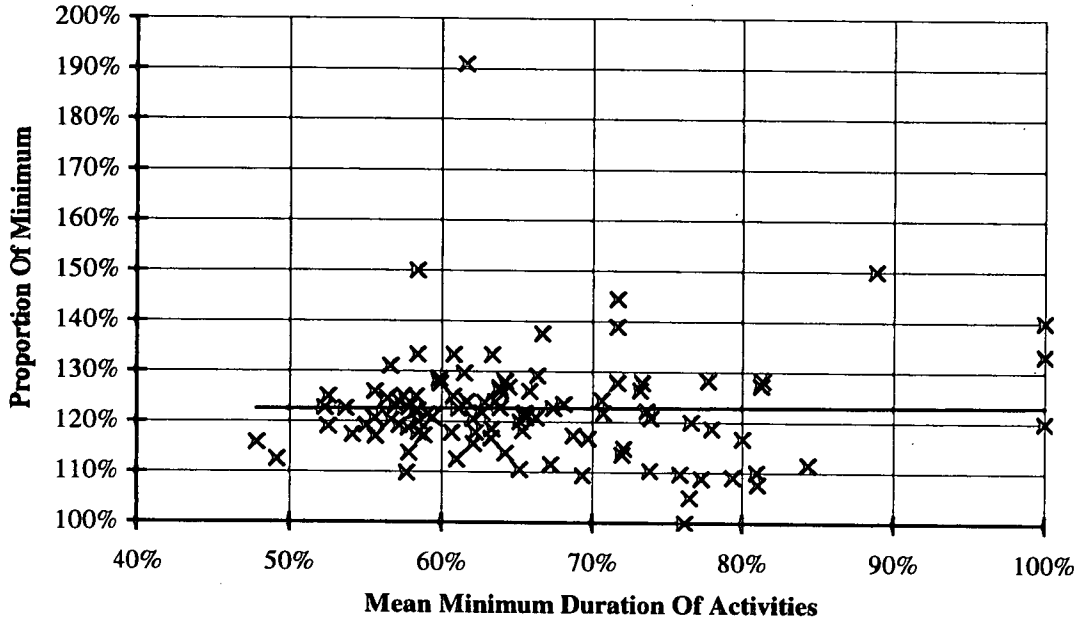


Figure 6.12. Efficiency of PRU case against minimum duration.

This plot has a lot of scatter, and no clear relationship between the mean minimum duration and the performance of the optimisation can be observed. This demonstrates that even if the unrealistic upper limits set on the resource requirements do cause the scope for optimisation to increase with project size, it cannot be responsible for the large increases observed with this data. Therefore it should be assumed that with real projects the scope for PRU optimisation will actually rise with increasing project size.

While this may be the case, it is possible that the increase in variables will make the optimisation much more difficult, and that as projects grow in size this factor might become the most significant of the two. As the project size increases the scope for optimisation will not decrease as much as it cannot go further than the absolute minimum value for the project. While the effect of the increase in variables may not have been large enough to reduce the values of the best solutions found by the optimisation, it is probable that, for larger problems, the increase in scope for optimisation is too small to outweigh the decrease in efficiency of the optimisation which must be expected with increasing project size. This means that with larger problems the best solutions found by the optimisation with the same computational effort will actually decrease.

6.4.2 PRU & MEST Cases

The two PRU & MEST cases (PRU & MEST and All Three) represent the true global approach, inasmuch as they make use of the full scope for optimisation. However, they do not perform on average as well as the PRU and PRU & priority cases. In addition to this, the average performance of these cases deteriorates with time. This deterioration can easily be explained by the large increase in the difficulty of finding efficient configurations using the MEST project input variables. This was identified in section 6.2.1, which considered the optimisation of MEST alone. In the same was as the individual MEST case, this difficulty can be illustrated by considering the feasibility rates of the optimisation. This is shown in Figure 6.13.

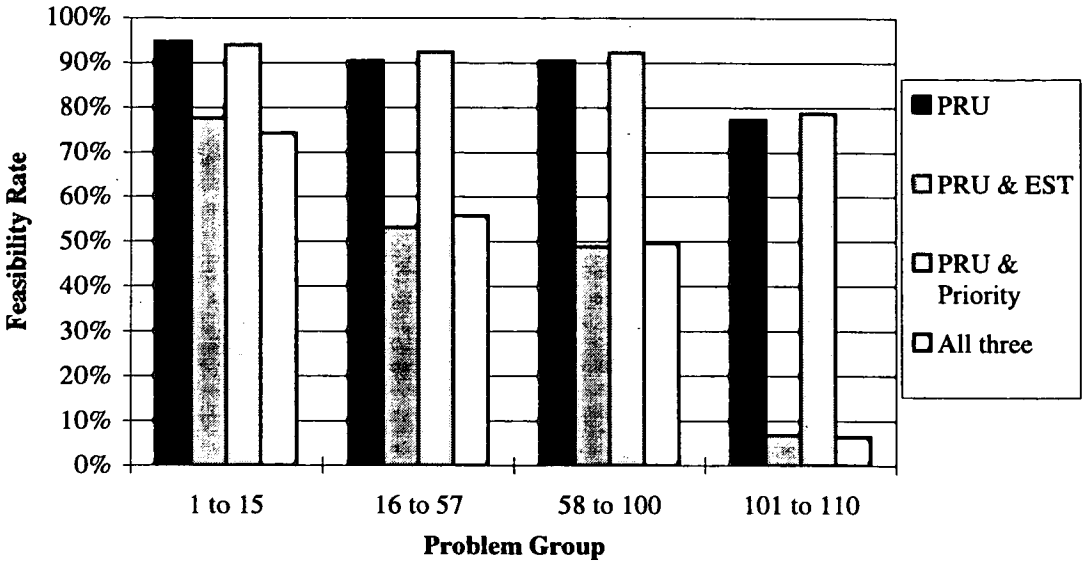


Figure 6.13. Feasibility of combined cases by problem group.

As with the comparison between the MEST and priority cases, the feasibility rates of the two cases which do not use MEST do fall slightly with increasing project size. However, the feasibility of the two that do use MEST show far more deterioration, particularly between the groups 58 to 100 and 101 to 110. This drop is even higher than for MEST alone, and explains the sudden drop in the performance of these cases on these problems as observed in Figure 6.5.

This increase in unfeasibility can be explained by considering the limits set on the MEST project input variables. When the limits on the MEST variables are set for the individual case, variation in duration is not considered. This is because there will be no variation in duration during the corresponding optimisation. However, for the combined cases variation in durations is considered for the setting of limits, which is performed with the durations set at their minimum values. This will lead to an increase in the ESTs and LSTs used to set the limits, an hence an increase in the size of the interval. This is illustrated in Figure 6.14.

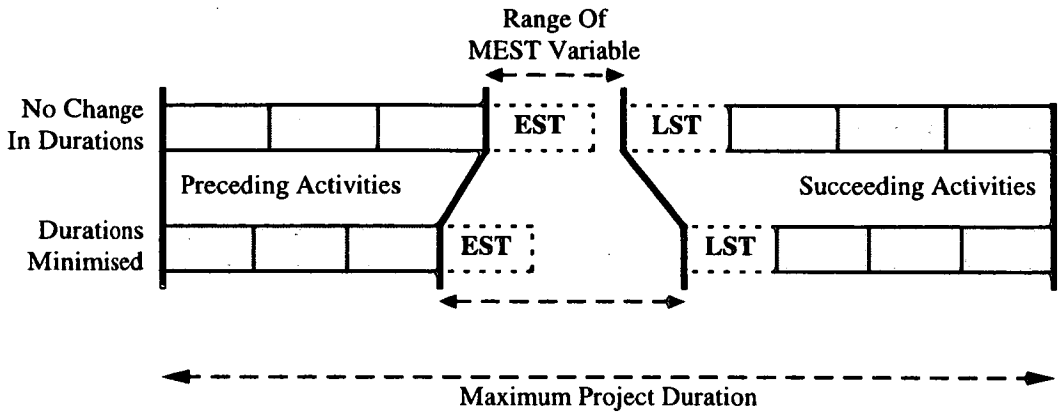


Figure 6.14. Range of MEST project input variables.

This increase in the distance between the limits will lead to an even greater chance of any mutation of a MEST project input variable generating a poor solution. This explains the fall in feasibility corresponding to the addition of PRU to the MEST case, as seen in Figure 6.13.

If an extrapolation of these results is performed then it can be seen that the use of the MEST variable in larger projects is likely to be very detrimental to the optimisation. Therefore this variable, which is critical to the true, global optimisation of these small problems, appears to be an unfeasible one to contain within the optimisation.

This could appear to invalidate the philosophy of global optimisation. Use of the priority project input variable instead of this variable causes a reduction in the scope for optimisation, and, as has been demonstrated for problem 1, can result in the exclusion of the true, global optimum. It might, of course, be possible to improve the way these project input variables are implemented. This might, for example, be achieved by increasing the probability of a low value being selected when a MEST project input variable is mutated, particularly for those early in the project.

However, without applying special techniques to improve specific optimisations, which is beyond the scope of this thesis, the priority method appears to be by far a better project input variable to use in activity selection. For the problems under analysis the priority cases found their optimal solutions (the target values) in 79% of

cases, so it can be asserted that the optimum is not excluded by the use of priority for at least 79% of the sample cases, and possibly more.

Indeed, it might be expected that for many of the 21% of problems for which the target value was not found the target value was not excluded from the feasible region, it was just difficult to find. However, speculation as to whether or not the target duration exists for these problems is not useful for the application of the global approach to real projects. It has already been established that the aim of an operational global optimisation is not to find the true global optimum, but to use the increased scope for optimisation introduced by the global approach to improve the optimum solutions found by the optimisation beyond what would have been found by considering the variables alone. As this is performed using finite computational resources, it is essential that the introduction of an additional variable is not detrimental to the optimisation. Indeed it is required that the addition of the variable improves the optimisation's performance. Therefore it must be recognised that while the proposed philosophy of global optimisation is to use the full scope of the optimisation available, there may be instances where to include a particular variable on these grounds is not feasible because its effects on the optimisation are too detrimental. Therefore the aim of global optimisation must be to make *best use* of the full scope for optimisation, rather than to make full use of it. This prevents a variable being included which makes the optimisation too difficult without rejecting the benefits of combining as many of the project variables as possible within the optimisation.

Therefore it is far more important to ask whether the priority method is a sufficiently good representation of the activity selection problem to perform well in large projects. This can be answered in part by considering again the performance of the priority cases by project group, as shown in Table 6.8.

	1 to 15	16 to 57	58 to 100	101 to 110
Mean Activities	17.8	22.0	27.0	51.0
Target Value Reached	86%	90%	67%	70%
Proportion Of Target	101.1%	100.4%	101.1%	100.6%
Median Runs To Target	1	133	1825	3963

Table 6.8. Performance of priority case.

This table shows that there is perhaps some reduction in the number of times the target value is found as the project size increases, although the trend is not as clear as the trend for the PRU case. Indeed, the largest and smallest values found are for the second and third groups, two very similar sizes of project. However, what is perhaps most important is the fact that the mean proportion of the target value does not really appear to change very much at all. If this could be extrapolated to large projects then it would suggest that priority would still be an efficient variable for use in global project optimisation.

6.4.3 PRU And Priority

While the use of activity priority rather than MEST does not mathematically represent global optimisation, it is certainly capable of finding the near optimal solutions required by a global optimisation. The PRU & priority case performs significantly better than the MEST & PRU cases over the 10000 run test period. For these 110 problems it represents the most efficient way of representing the two different types of variable available for optimisation: activity selection and resource allocation. While it doesn't necessarily include the global optimum for all projects, it does permit the formulation of efficient near optimal solutions for any project. When the efficiency of the alternative variable - MEST - is considered, it can be seen that to ensure the inclusion of the true mathematical optimum for all projects will be very costly to the progress of the optimisation. Therefore activity priority, rather than MEST represents the use of the global approach as it would be used on a real project.

The addition of the priority to the PRU case has significantly improved the performance. This shows how the use of the global approach enables better solutions to be found by the optimisation. However, while the use of the global approach has proved to be effective for this set of small problems, it is important to try to assess

whether the success of the global approach might be extrapolated to large projects. This can be done by comparing the PRU & priority case to its closest rival, the PRU case. If the improvement of the PRU & priority case over the PRU case is studied more closely it may be possible to identify whether the improvement increases or decreases with increasing project size. Therefore this improvement is plotted against the number of activities. This is shown in Figure 6.15, along with a trend line, plotted by the method of least squares.

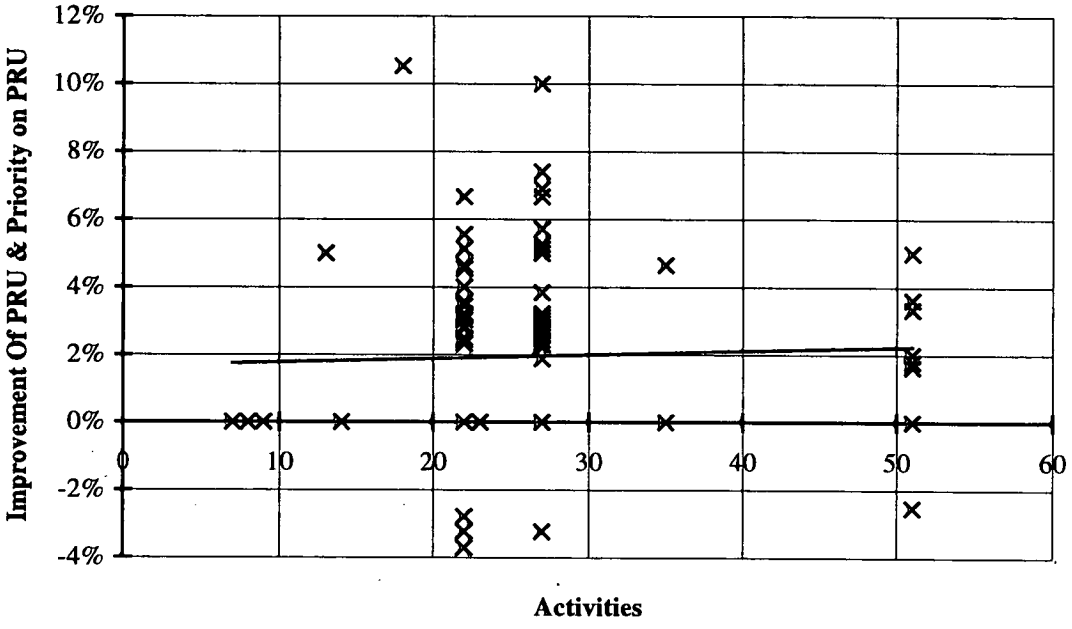


Figure 6.15. Improvement of the PRU & priority case by project size.

Figure 6.15 shows that there is no clear trend for variations in the project's size to correlate with improvement of the PRU & priority case over the PRU case. There is a lot of scatter in the results, and the trend line is almost flat. This, on its own, might suggest that the improvement of the PRU & priority case over the PRU case may be expected to remain approximately constant as project size increases. However, this trend line suggests that the average values for each of the four groups of project size are approximately the same. This is not the case, as is demonstrated by Figure 6.16.

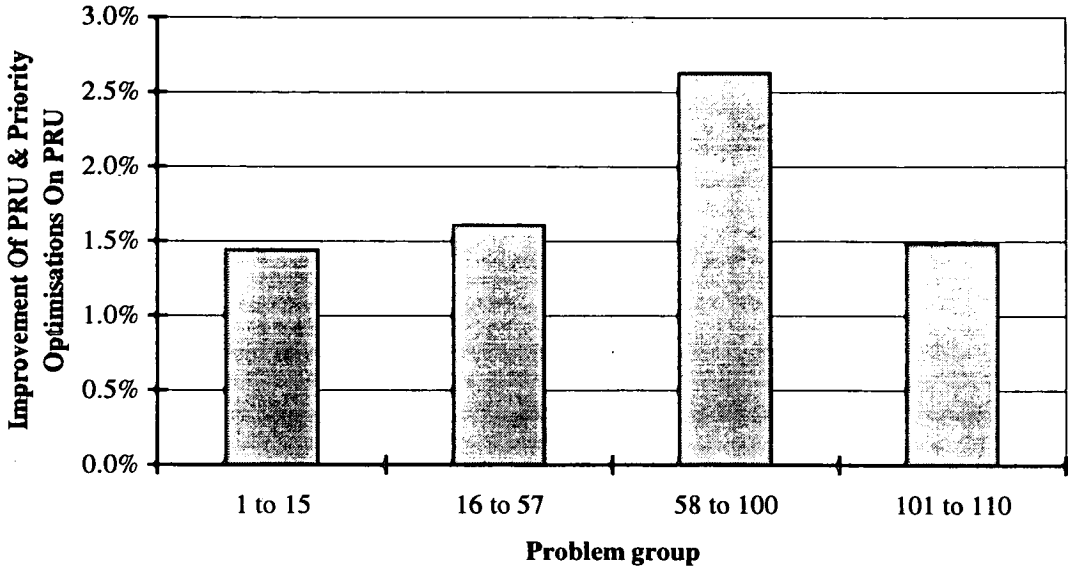


Figure 6.16. Mean improvement of the PRU & Priority case over the PRU.

It is clear from Figure 6.16 that there is a considerable decrease of the performance of the PRU & Priority case between the second largest and largest groups, although the difference has been rising until that point. There are two possibilities with this problem. Either these differing results have occurred by chance or they are the direct result of a real phenomenon.

If the null hypothesis - that there is no real difference between these values - is assumed then it is possible to perform an analysis of variances. The analysis of variances compares a weighted average of the sample variances to the variation of the sample averages around the mean for all the results. By dividing the latter by the former, a test statistic F is obtained. Provided the null hypothesis holds, this statistic follows an F distribution. The results of this analysis obtain a value of the test statistic F of 1.595. The maximum value of F before the null hypothesis should be rejected at the 95% confidence level is 2.7[7]. Therefore it is not possible to reject the null hypothesis with any confidence. It must be accepted that that this inconsistent variation in the results may have occurred by chance.

---

[7] From Hogg & Ledolter[Hogg, 1992], p450 by interpolation.

However, it is also possible that the results are representative of trends which are real, but not sufficiently large in comparison to the scatter of the results to cause the rejection of the null hypothesis. As this scatter is so large and the trend inconsistent it is impossible to assert from these results alone that there is any trend which might be extrapolated to larger projects. However, if a possible causes of this phenomenon were to be observed it might be possible to make a firmer judgement as to the performance of the PRU & priority case for larger, real, projects.

In the identification of possible influencing factors, it is necessary to address what might be expected from the results, and see what affect that has on different aspects of the optimisation.

The first thing which might be expected is that the optimisations of larger projects do not progress as much as the smaller optimisations. As the number of project input variables increases, the optimisation will become more difficult. Hence more effort would be required to obtain a near optimal solution.

Figure 6.4 shows how the values obtained by the PRU case are quite similar to the values obtained by the PRU & priority case at low run numbers. It is only later in the optimisation, when the optimisation of the PRU case's progress begins to diminish and the optimisation of the PRU & priority case continues to progress, that a large difference between the two is observed. The flattening out of the PRU case's progress could be explained by the optimisation coming close to its optimal solution. The continuation of the PRU & priority case beyond this point can also be explained by the fact that its optimal solution is lower, so progress continues until it comes closer to its own optimal solution.

If this is the case, then as the optimisations become more difficult both the PRU and PRU & priority cases would be expected not to be as near to their optimal solutions after the same number of runs. The fact that the optimisation is essentially in earlier stages would explain the decrease in the improvement of the PRU & priority case over the PRU case in the final group because the two cases tend to be closer together earlier in the optimisation. This would explain the decrease between the third group

and the final group. The increase observed between the first three groups can also be explained by asserting that the difference between the optimal solution for the PRU case and the optimal solution for the PRU & priority case is increasing with increasing project size.

Whether this might be true can be observed by considering the progress of the cases by group. The progress of the optimisations is shown for the four groups individually in Figure 6.17 for the PRU case, and Figure 6.18 for the PRU & priority case.

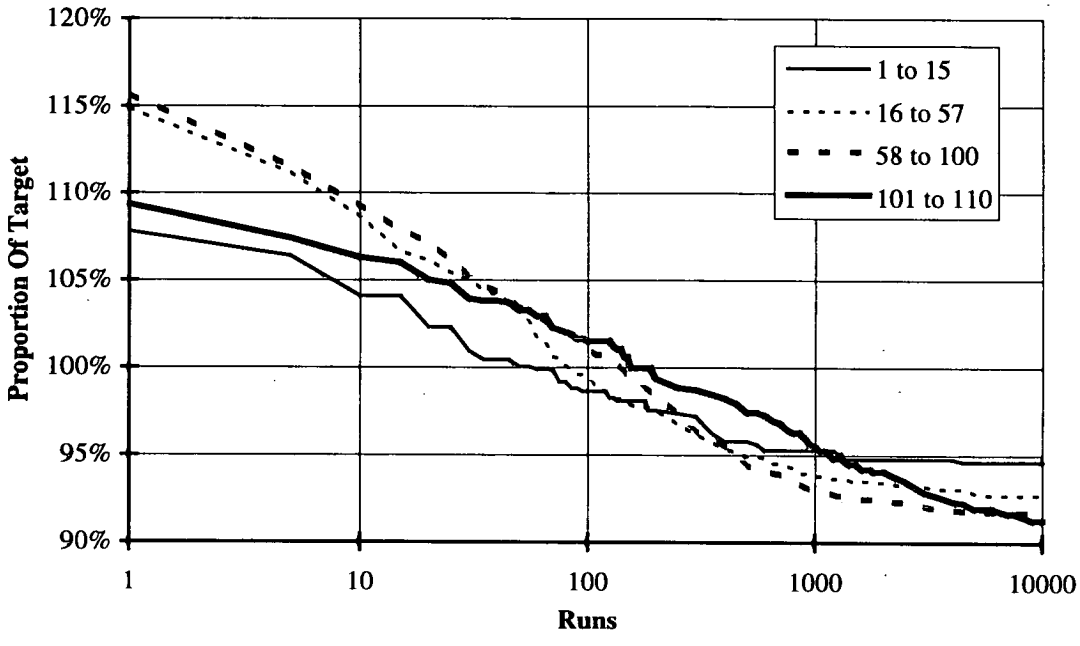


Figure 6.17. Progress of PRU case by group.

The PRU case, as expected, shows the smaller projects progress initially follows an approximately straight line on the log plot, before levelling out as the optimisation approaches the optimal solution. The smaller projects 1-15 level out first, at around 500 runs. The next two groups level out at around 1500 runs. However, the group of projects 101-110 does not appear to show this levelling out of optimisation at all. This is probably caused by the optimisation's not being sufficiently close to its optimal solution after 10000 runs, and is as predicted by the theory postulated to explain the variations in the improvements of the PRU & priority cases of the different groups.

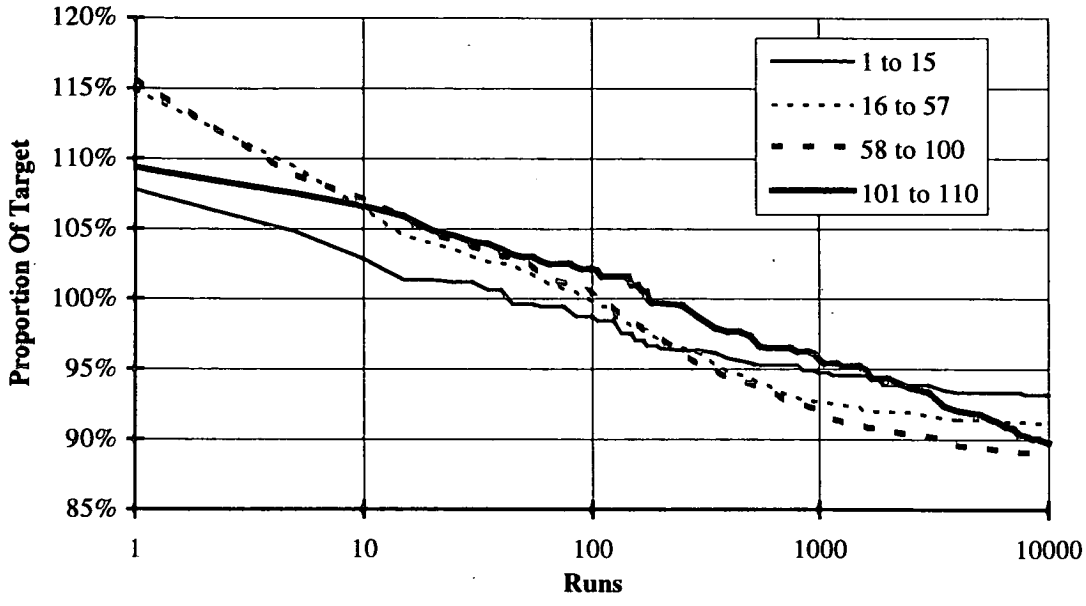


Figure 6.18. Progress of PRU & priority case by group.

The progress of the PRU & priority case behaves similarly to the PRU case. The first three optimisations level out, albeit a little later than the PRU cases, but the largest group appears not to. These results are also as predicted by the theory to explain the variations in the final values of the improvement of the PRU & priority optimisation over the PRU case. In addition to this, some useful observations an also be made from this plot about the difficulty of finding optimal solutions.

The fact that the PRU & priority case showed a later levelling out of the first three groups shows how increasing the number of variables makes finding a near optimal solution require more effort. The fact that the largest group does not level out for either of these optimisations demonstrates that the number of runs required to obtain even a near optimal solution should be expected to increase with increasing project size. Thus obtaining a near optimal solution is expected to increase with both project size and the number of variables considered. This will make obtaining a near optimal solution for a real project almost impossible. 10000 runs is insufficient computational effort to obtain a near optimal solution for projects with 51 activities and two different types of variable. Given this fact, finding near optimal solutions to

projects with several thousand activities and many more than two types of variable is not realistic.

This observation demonstrates the need for the global optimisation (which corresponds in these problems to the PRU & priority case) to perform better than the other cases at all stages of the optimisation. In larger projects the PRU case will not have levelled out, so the difference between the PRU case and the PRU & priority case earlier in the optimisations is important. It is also important for the theory used to explain the variations in Figure 6.16. This theory assumes that the reason for the decrease in the improved performance of the PRU & priority case between group 58-100 and group 101-110 is caused by the PRU case of the final group's not having levelled out. This is most certainly true, but it may not be the only factor involved. If the global optimisation is to be useful then the improvement in the largest group must be observable earlier in the optimisation. Therefore the improvement is analysed throughout the optimisation for each group, and is shown in Figure 6.19.

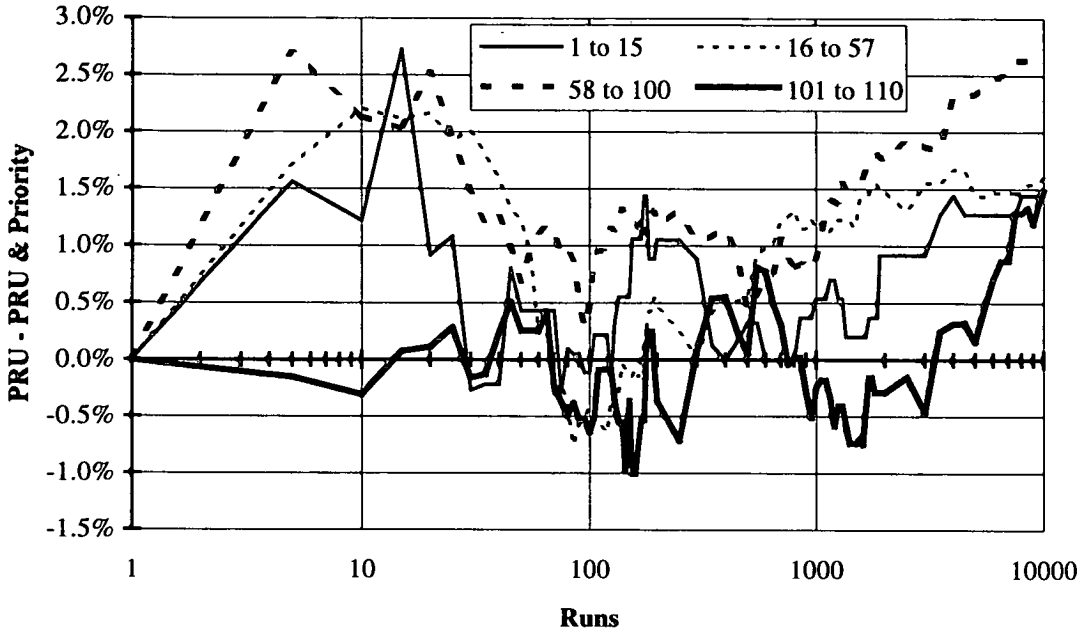


Figure 6.19. Improvement of PRU & priority on PRU as optimisation progresses.

Figure 6.19 has a lot of scatter, again a symptom of the fact that there is a lot of scatter in the results. However, this figure is still able to show that while the improvement after 10000 runs is significant, the apparent improvement of this group earlier in the optimisation is not so. The effects of the levelling out of the PRU case can be observed as the sharp rises in the values which appear later in the optimisation. This was expected. Nevertheless these rises in improved performance at the end are not as great as the rises which are observed at the very start of the optimisation for the first three groups. This initial rise was not anticipated. It appears that initially the PRU & priority case of the first three groups is significantly better than the PRU case, but the PRU case has "caught up" by about 50 runs. Then, as the PRU optimisation levels out at around 1000 runs the improvement becomes more marked again.

The hypothesis that the unexpected drop in the performance of the largest group is caused only by the fact that the PRU case has not really begun to level out is not supported by this plot. If the hypothesis were true then the improvement of the PRU & Priority case would be expected to be highest for group 101-110 early in the

optimisation. This is not the case. The improvement associated with projects 101 to 110 is substantially less than the others throughout the optimisation. This can be shown more clearly by considering the averaged improvement over different sections of the optimisation, as shown in Figure 6.20.

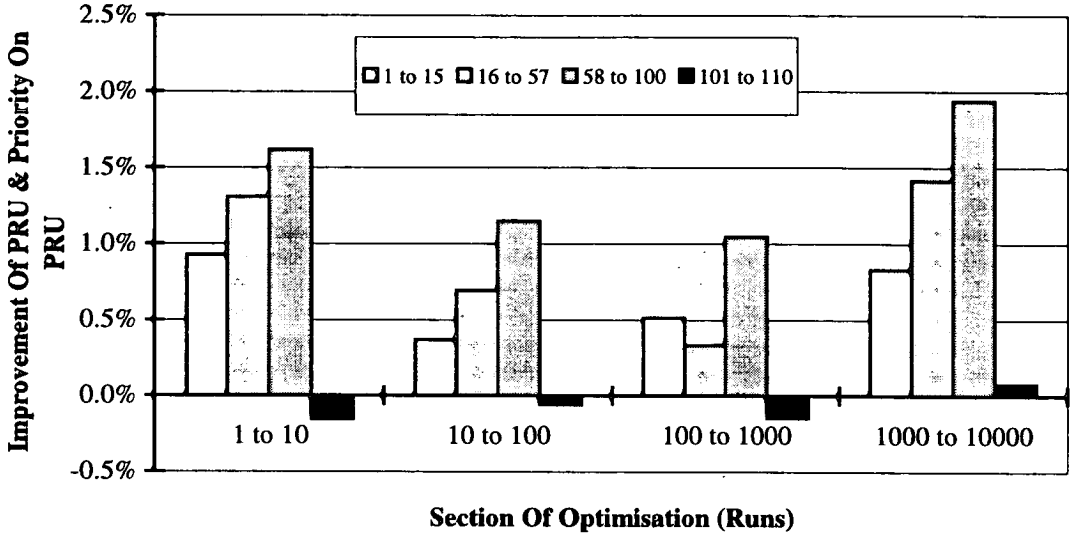


Figure 6.20. Averaged improvement of PRU & priority on PRU.

This figure shows that not only does the optimisation of group 101-110 not perform as well as the others earlier in the optimisation but it does not, on average, improve on the PRU case at all until quite late in the optimisation. If this sudden drop of performance can be extrapolated to larger projects then it suggests that the use of the global approach will not actually improve the values of optimal solutions found at all. Therefore it is necessary to try to identify the possible causes of this phenomenon.

The first possibility is that these results might have occurred by chance. If the null hypothesis is accepted, then it is possible to perform an analysis of variances on the different phases of the optimisation. As it has been shown from Figure 6.19 that the PRU & priority case's improvements on the PRU case has not begun to rise significantly until after 1000 runs, the improvements of each project can be compared before the PRU case has levelled out by considering the averaged values from 100 to 1000 runs. This can then be used in an analysis of variances. The value of the *F*

statistic obtained is 0.908, which is substantially less than the value of 2.7 required for the rejection of the null hypothesis at the 95% confidence level. Therefore it is possible that these results have been arrived at by chance. Nevertheless, it is also possible that the trend is real, but fails the statistical significance test because of the large amount of scatter in the results. Therefore possible causes of the observed poor improvement of the PRU & priority case over the PRU case will be investigated.

As well as using the final values of the optimisations and evaluating the relative performances at different points in the optimisation, another way of measuring optimisation performance is to consider how long it takes for the optimisations to reach the target value. If the optimisations are becoming harder as the project size increases then the average runs to target would be expected to increase with increasing project size. The PRU case and the PRU & priority case are compared in Figure 6.21. As not all the PRU cases achieved the target value, it was necessary to use median values for the comparison.

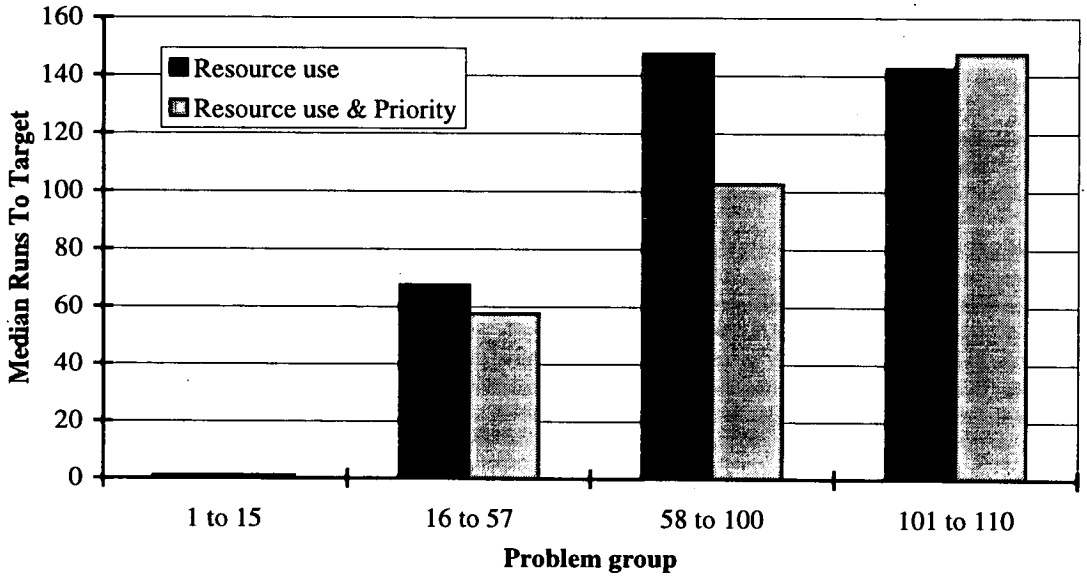


Figure 6.21. Time to target duration by group for PRU and PRU & priority.

As this figure shows, the PRU case initially takes longer to reach the target value than the PRU & priority case discounting the first group, for which over half of the problems begin start from the target duration anyway. This difference increases from

the second group to the third, and then the difference is reversed for the fourth group, for which the PRU & priority case takes longer than the PRU. This is exactly what would be expected as the PRU case tends to be better than the PRU & priority case until very late in the optimisation, by which point the target value has already been exceeded.

Another observation which may be made from Figure 6.21 is that the median time to target for the PRU & priority case rises with increasing project size. This should be expected as the problems are becoming increasingly difficult with increasing project size. The reason that the PRU case reaches the target duration earlier for group 101-110 is that it displays a decrease in time to target from the previous group. This conclusion is also consistent with the averaged results from Figure 6.17, which show that the PRU case of the two largest groups achieve 100% average proportion of target duration at approximately the same number of runs, although for these results – obtained by average rather than median – group 58-100 actually reaches the value first. For the PRU & priority case in Figure 6.18 group 58-100 achieves 100% in nearly half the number of runs, despite its average initial duration being further away from the target duration. This shows that while the PRU & priority case of the largest group is progressing as would be expected when the optimisations of the different groups are compared, the PRU case seems not to show the same order of diminished performance.

Although this analysis is useful for identifying the apparently unexpected deviation from the trend, it suggests that the average runs to target duration is increasing substantially. This is, however, not the case, as a plot of the number of runs to target by the project size in activities, in Figure 6.22, shows.

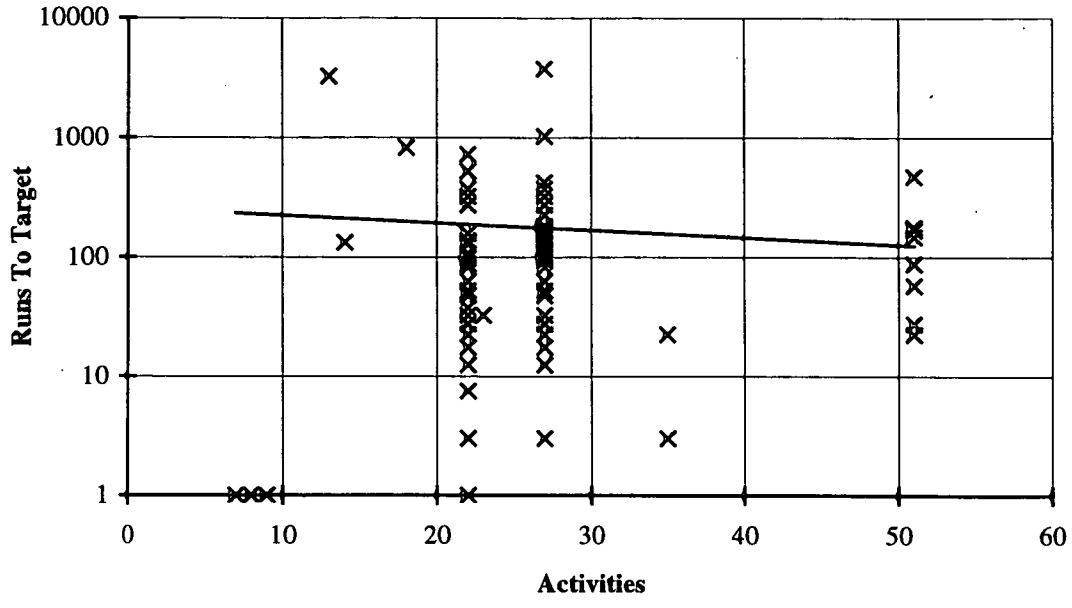


Figure 6.22. Runs to target by project size.

While there is a lot of scatter in these results, the overall trend is for the number of runs to target to be less for larger projects. This is the converse of what would be expected if the target value were to be a representative measure of how hard the optimisation is. However, it has already been demonstrated that for the PRU case the best solution found tends to be further away from the target value as the size of project increases. Thus for larger projects it may be easier to find the target value, despite the optimisation's becoming more difficult. This is because the target value is not as close to the optimum. This suggests that while, with larger projects, the optimisation may be further away from the optimum solution, the optimisation is still finding efficient solutions. This is expected to be true both for the PRU case and the PRU & priority case.

This result may be valuable in assessing how these two case's behaviour changes with increasing project size. Nevertheless, it does not help with the identification of the cause of the unexpected reduction in the improvement of the PRU & priority case over the PRU case alone. Therefore it is necessary to turn to the physical characteristics of the project themselves to see if there are any identifiable differences between them which might cause the result.

Perhaps the first and most obvious characteristic would be the variability of the activity durations within the projects. This has already been presented as a scatter plot in Figure 5.1 in section 5.2.4.3.1. The compressibility and expandability of the activities have also been presented in Figure 6.8 and Figure 6.9. However, in order to assess how these factors may vary from group to group, all three values are shown, averaged for each group, in Figure 6.23.

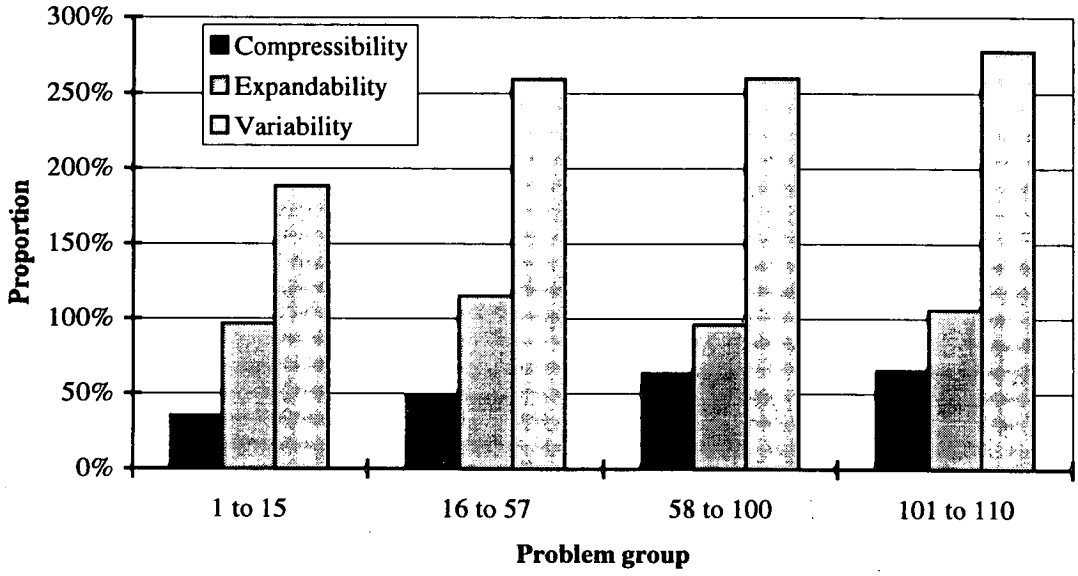


Figure 6.23. Average activity compressibility, expandability and variability by project group.

This figure shows a number of factors quite clearly. Firstly the mean expandability does tend to vary from group to group, but with no clear trend. As this value can more or less be related to the resource allocations to the activities, it shows that, in this respect at least, the nature of the activities is quite consistent from project to project. The compressibility does change however. It rises between the first three groups and then remains roughly constant between the third and the final groups. The result of this variation is that while the smallest group shows quite a low mean variability, the last three remain approximately constant. Therefore the change in the relative performance of the final group's optimisations and the previous optimisations can not be related to the whole variability of the activities.

However, it is possible that the compressibility of the activities is partly responsible for the phenomenon. The compressibility of the activities rises substantially for the first three groups, and then there is little difference between the third and final groups. It is possible that there is a trend for the PRU & priority case's performance to diminish relative to the PRU case's performance with increasing project size. It is also possible that this trend is offset by the increasing compressibility of the project. If increasing the compressibility of the project can be expected to improve the relative performance of the optimisations then the increases in compressibility observed over the first three groups would be expected to correspond to increasing performance over those same groups. Also, when there is no further increase in the average activity compressibility, the trend for the relative performance to diminish with increasing project size would be the dominant trend. Hence the diminished performance of the final group.

If this were true, then it would be expected that there be a trend for the PRU & priority's performance to increase relative to the PRU case with increasing activity compressibility. As this trend is required to offset the trend for the improvement of PRU & priority case over the PRU case to decrease substantially with increasing project size, it must be a strong one. The two measures are plotted against each other in Figure 6.24, and a trend line shown fitted by the method of least squares.

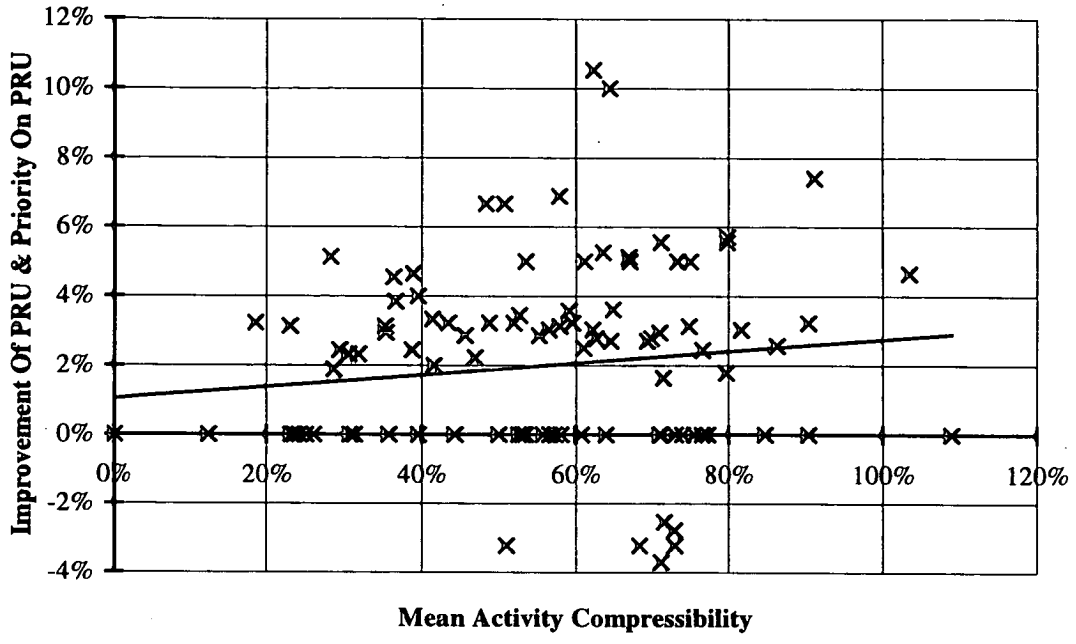


Figure 6.24. Improvement of PRU & priority on PRU against activity compressibility.

Figure 6.24 shows that the general trend for the projects is that as the mean activity compressibility increases, the improvement of the PRU & priority case over the PRU case also increases. The direction is as predicted by the theory. However, the trend is not a strong one, as was required. It can also be seen that there is a great deal of scatter on this plot, which might imply that there are other factors involved. Indeed, given that project size has also been identified as a likely cause in the variation of the improvement of the PRU & priority case over the PRU case, it is possible that a lot of scatter is introduced by the varying project sizes within the data. In order to remove this factor, it is necessary to see how the improvement in performance varies with activity compressibility when the project size is held constant. Therefore the trend lines are plotted for each of the four groups. This is shown, with a trend line for each group, in Figure 6.25.

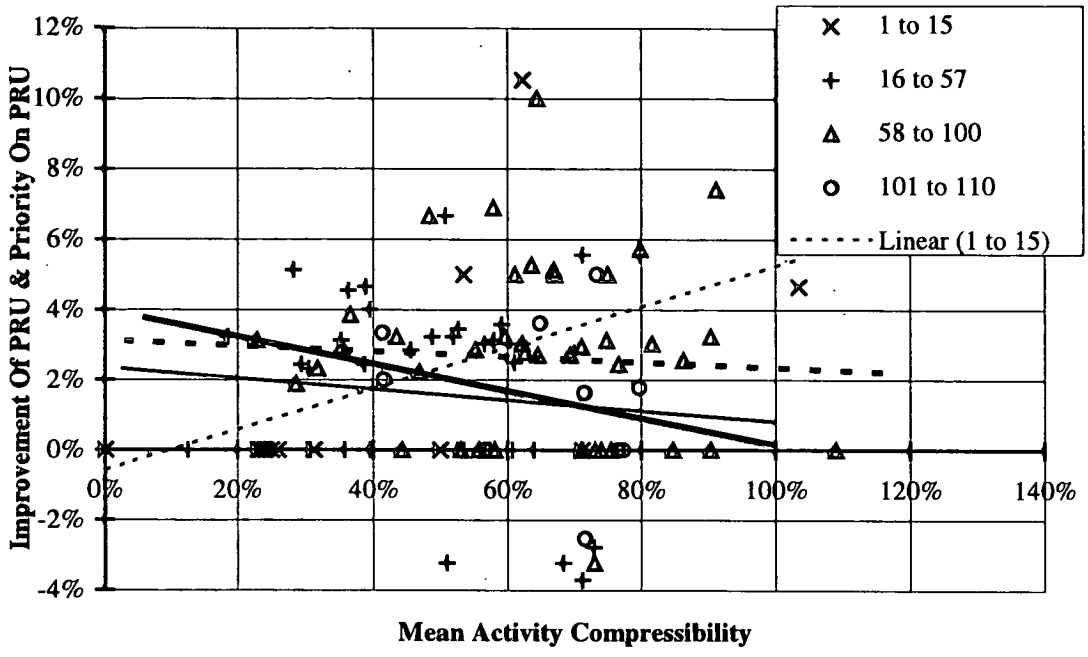


Figure 6.25. Improvement of PRU & priority on PRU against activity compressibility, by group.

This figure demonstrates that when the variability caused by changing project size is removed, the relationship between compressibility and the improvement of the PRU & priority case appears to be reversed. The first group, which is not of constant project size, displays a more marked trend to increase with increasing compressibility, while the trends of the other three, which are of constant size, are the reverse. As compressibility increases, the improvement of the PRU & priority case over the PRU case decreases. However, there is a lot of scatter in these results, which makes it difficult to assess their validity. In order to show this more clearly, the results of the regression analyses for the trend lines are shown in Table 6.9, including their coefficients of determination, R^2.

Group	Slope	Constant	R^2
All	0.0171	0.0104	0.0207
1 to 15	0.0582	-0.0057	0.3309
16 to 57	-0.0155	0.0236	0.0099
58 to 100	-0.0078	0.0313	0.0031
101 to 110	-0.0386	0.0401	0.0627

Table 6.9. Regression analyses of improvement of the PRU & priority case against activity compressibility.

The low values of R^2 found in this table are expected, due to the scatter of the results. Indeed, it could be argued that the slopes are arrived at by chance variations in the data. Thus these very low values make the validity of any comparisons of the slopes of the lines questionable. However, the high scatter does indicate that the relationship between activity compressibility and the improvement of the PRU & priority case over the PRU case is not a marked one, and that there are other influencing factors. Indeed, for the three groups for which activity size was constant the converse of the expected relationship was observed. This suggests that increasing the activity compressibility actually tends to correspond to a decrease in the difference in the improvement of the PRU & priority case over the PRU case. Therefore the theory which it was hoped this observation would justify must be rejected. Increasing activity compressibility does not increase the improvement of the PRU & priority case over the PRU case. A strong relationship was required to justify this, and the level of scatter and absence of the required trend show conclusively that the required trend does not exist. There is also no evidence to suggest that increasing project size decreases the improvement, as the one group for which variations in project size were retained actually showed an increase in the slope of the line for the improvement to increase with increasing compressibility, suggesting that it was caused by variations in project size.

Now that it is possible to reject the differences in performance being caused by the unrealistic limits set on the PRU project input variables, it is possible to address other possible causes. Another possible cause of the sudden decrease in the relative performance of the final group is the varying scope for further optimisation beyond that found by the PRU case. The performance of the PRU case was observed to come closer to the calculated minimum duration as the size of the optimisations increased. A neither optimisation can exceed this value it is possible that this reduced space between the absolute minimum and the minimum of the PRU case is reducing the scope for further optimisation available to the PRU & priority case. If this were true then there would be a relationship between the improvement and the scope for further optimisation beyond that achieved by the PRU case.

While it must be recognised that this minimum duration is not a measure of the true global optimum for the project, it is possible that it represents a consistent indication, from one project size to the next, of the likely value of the global optimum for each project. It can also provide some small indication of how difficult it is to obtain optimal solutions. The closer to the minimum the duration is, the more tightly packed into a small resource space the activities must be.

The PRU case has already been shown, in section 6.4.1, to tend to improve as a proportion of the minimum as the project size increases. This indicates that, as project size increases, either the PRU case is able to arrive at solutions which are closer to the global optimum, or arrive at solutions which are more difficult to improve upon. This means that the PRU & priority case either has less scope for further improvement available to it, or will find it more difficult to find a better solution than the PRU case. This effect can be demonstrated by considering the plot of the PRU's proportion of the minimum duration and comparing it to the improvement of the PRU & priority case upon it. This is shown in Figure 6.26.

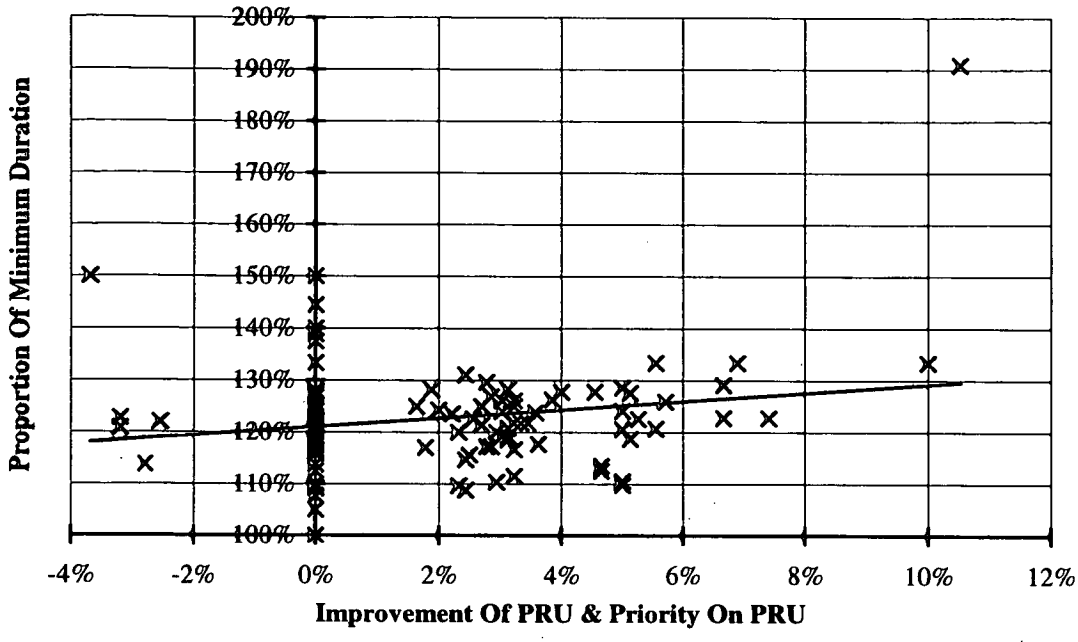


Figure 6.26. Relative performances against the PRU's performance.

Figure 6.26 shows that as the PRU case obtains solutions closer to the minimum the PRU & priority case tends to be less likely to improve upon it. This is a useful observation, as it shows that when the increase in scope for optimisation introduced by using the global approach is small, the global approach is less likely to be able to improve upon the solution. However, as with the comparison of the improvements to the activity compressibility, it may be possible that the observed trend is a secondary one. It could be caused by the variation of the improvement with project size and the variation of proportion of minimum obtained by the PRU case by project size. Therefore it is useful to observe whether the observed trend persists once variations in project size are removed. This is shown in Figure 6.27.

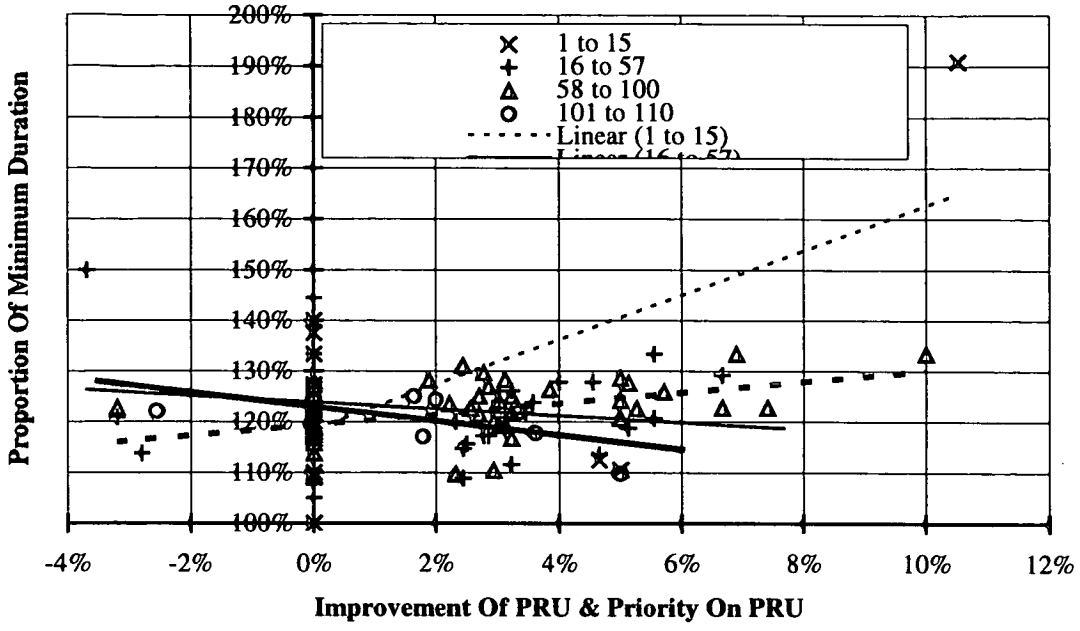


Figure 6.27. Relative performances against the PRU's performance by group.

As with the performance by compressibility, once the variation in project size is removed, as it is with the three larger groups, the trend is not clear. The slopes are very small, and only the slope of group 58-100 is actually in the same direction as was expected. This would suggest that how well the PRU case performs as a proportion of the minimum duration is not directly linked to by how much the PRU & priority case will tend to improve upon it.

However, when considering possible effects of the minimum duration, it is not only possible to compare the improvement to the final value of the optimisation as a proportion of the minimum, but also the initial value. If some optimisations start closer to their optimal solutions than others, it is possible that there could be some relationship between this value and the relative performance of the cases. There is a difference in how close to the optimum solutions the projects begin, as can be shown in Figure 6.28.

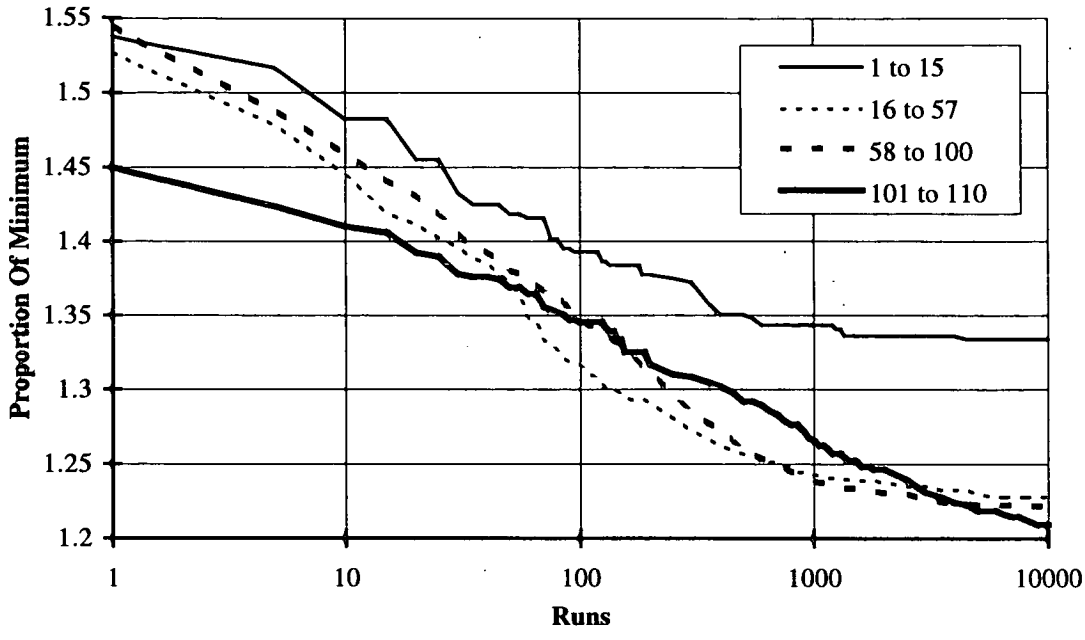


Figure 6.28. Progress of the PRU case as a proportion of the minimum.

This figure shows quite clearly that the optimisations average starting values, as a proportion of the minimum duration, are not constant. What is particularly interesting is that if the fact that the initial durations can be made to follow a similar pattern to the relative improvements of the three groups. Group 58-100 has the highest initial value. When it is considered that problem 1 in group 1-15 has an unrealistically high value of proportion of minimum, as was discussed earlier in the section, it can be asserted that if this problem were to be ignored then the next highest mean initial value is that of group 16-57, followed by group 1-15 and finally group 101-110. This is the same trend as was observed in the improvements. However, to show that there is a definite link it is necessary to compare how the initial value as a proportion of the minimum affects the improvement. As with previous analyses it is also necessary to try to ensure that the results are independent of project size. They are shown in Figure 6.29.

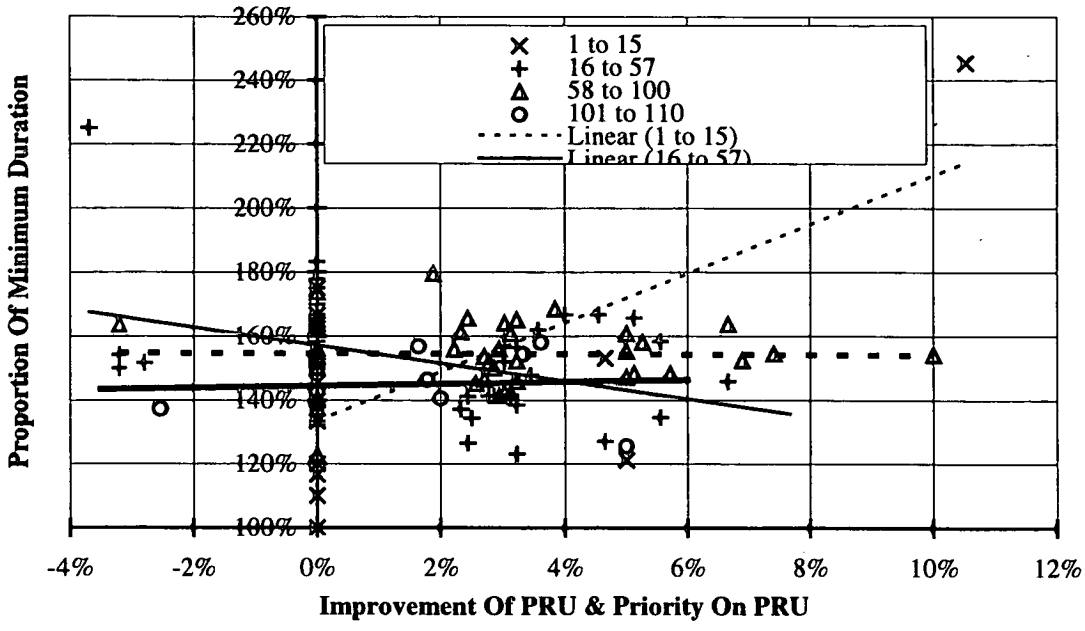


Figure 6.29. Improvement by initial duration.

As this figure shows, while the initial durations do follow the same pattern as the optimisations, there is no clear trend for the improvement of the PRU & priority case to relate to these initial values. Therefore the initial duration cannot explain the observed variation in the improvements by group.

While considering initial values, however, it is also possible to consider the initial value as a proportion of the target. While this does not provide any comparison of the initial value as a proportion of the overall optimum for the PRU & priority case, it could provide an indication of how efficiently the activities are arranged initially. If the activities are poorly arranged, then the PRU case must attempt to vary the activities in such a way as to overcome these inherent inefficiencies in the way the activities are arranged. This corresponds to the situation where the initial durations are far from the target. However, if the activities are well arranged then the optimisation only needs to perform adjustment of an efficient system towards an optimal solution. There are no patterns of inefficiency to overcome. Whether this factor might have any effect on the final values of the improvement of the PRU & priority case over the PRU case can be assessed by considering Figure 6.30, which

plots these two values by group, with a trend line fit by the method of least squares for each.

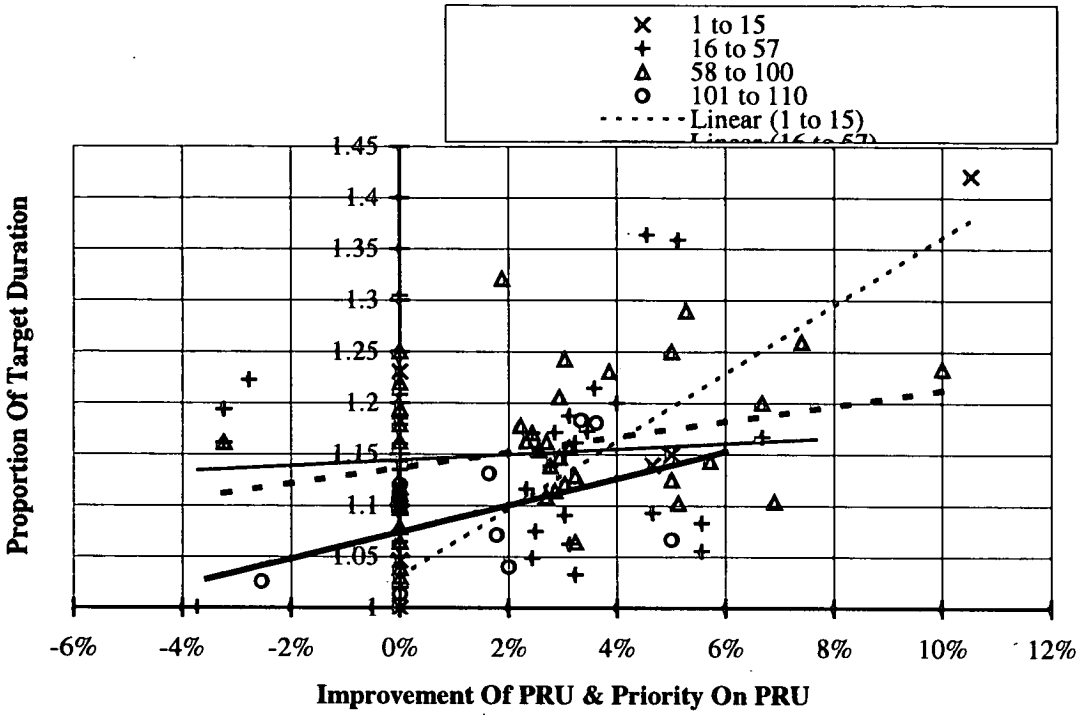


Figure 6.30. Improvement by proportion of target.

This figure shows a consistent trend for the improvement to rise as the initial duration rises as a proportion of the target value. This occurs for all groups, although the trend is stronger for some groups than for others. This difference in steepness of the trend lines can be explained by the high amount of scatter in the results. This scatter may also cast some doubt on the validity of the observed relationship. Nevertheless all the four groups show the trend to be in the same direction. If the slopes of the trend lines were simply down to scatter then it would not be expected that the slopes of all four group's trend lines would be positive. This trend can also be explained by considering the fact that at higher values of initial duration the PRU case will be harder. The inefficiencies inherent in the ordering of the activities cannot be overcome by the PRU case alone, because it does not have the ability to directly reorder the activities. Therefore the optimisation, in order to find a good solution, must try to arrange the resource allocations in such a way as to reduce the

impact of this inefficient system. In some cases it is possible the optimisation may be able to do this so well that there is no real difference between the optimal solutions to the PRU case than the PRU & priority case. The PRU & priority case, on the other hand, should always be able to overcome this inefficient ordering of activities, and its solution should therefore always be a very efficient one.

This trend can also explain the diminished improvement of the PRU & priority case over the PRU case. The initial duration varies by project group as shown in Figure 6.31.

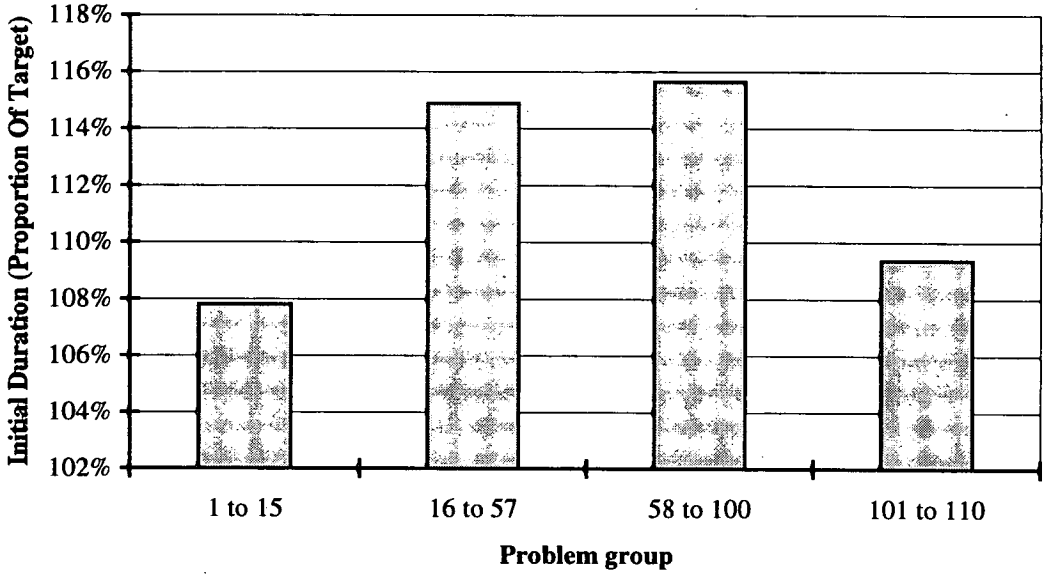


Figure 6.31. Mean initial duration by problem group.

This variation in initial duration follows a similar pattern as Figure 6.16. The mean initial duration rises as a proportion of the target value, before dropping for the final group. As it is known that there is a positive relationship between the average improvement of the PRU & priority case over the PRU case, the initial duration can be shown to be an influencing factor in the final value of this improvement.

6.4.3.1 Implications For Implementing The Global Approach To A Real Project

The relationship between initial duration and the improvement of the PRU & priority case over the PRU case, when interpreted in terms of general global optimisation, can

help identify certain factors which may affect the benefits of employing the global approach. The observed improvement is obtained by the addition of the priority project input variable to the existing PRU case. Adding this variable introduced additional scope for optimisation. It was found that when the scope for optimisation introduced by the addition of this variable was higher (as it would be, on average, when the initial duration was higher than the target), the improvement of the global approach would be higher. Therefore the benefits of the implementation of the global approach are expected to be greater when the increase in scope for optimisation introduced by its implementation is greater. Where the increase in scope is large, the benefits are expected to be large. Conversely, where the increase in scope is small, the benefits of implementing the global approach is likely to be smaller.

The second conclusion which may be drawn from the observations from Figure 6.31 is the importance of considering the interrelationships between the project input variables. When the optimisation of the PRU variables alone is performed it fails to consider the fact that the ordering of the activities may be having a detrimental effect on the optimisation's performance. However, when the priority variable is introduced to this optimisation, it can take this into account. This allows the PRU & priority case to improve on the PRU case. The values of the priority project input variables can be detrimental to the PRU case, and are not considered with in this local optimisation. However, when the global approach is adopted, the effects of such detrimental configurations of priority project input variable are considered as part of the optimisation, allowing better solutions to be found. This shows that where two variables are inherently interrelated it is important to consider the effects of both variables within the optimisation.

While this observation regarding the optimisations may be true for the comparison of the PRU case and the PRU and priority case, it could be argued that the value of the PRU optimisation might be improved by performing a short local optimisation using priority alone first. This would remove any large inefficiencies from the ordering of the activities and provide a project with which the PRU case's performance might be

more comparable to the PRU & priority case. While this may be true, this kind of sequential optimisation may not be appropriate for true global optimisation. It must be remembered that while the PRU and priority project input variables are interrelated, they are both employed, within this optimisation at least, with the same target: to create a schedule of activities which makes efficient use of resources. It has been shown in the consideration of Figure 6.30 that to improve the performance in terms of the priority project input variables will increase the possible performance in terms of the PRU project input variable. Therefore sequential optimisation may work in this isolated case because local systems represented by the project input variables operate in a mutually beneficial manner. What is of benefit to one optimisation will usually also be of benefit to the other.

However, a real global optimisation would contain a much larger number of project input variables. Each of these project input variables represent small local subsystems of the project. Localised optimisation of these subsystems has been criticised because optimising one is usually detrimental to many others. Therefore it must be concluded that for the project input variables to behave in a mutually beneficial manner will not be the norm for projects. Therefore such sequential optimisation is unlikely to be successful.

While this shows how the general trends in the data fit together, and how the benefits vary from project to project on average, it must be remembered that all the trends identified showed a lot of scatter. Therefore while identifying whether such factors it was necessary to assert that the global approach had improved upon local approaches *on average.* This implies that it cannot be asserted for any particular project considered that the global approach would necessarily yield the best performance. It was only possible to express a probability that it would.

However, it was also observed that for many of these values which showed a great deal of scatter, the scatter was reduced as the project size increased. This was discussed in terms of the final performance at the beginning section 6.4. If this reduction in scatter were to continue, and the increased average performance of the global approach over localised techniques to persist as the size and complexity of

projects increases towards a real project model, then the application of the global approach could be expected to provide better optimisation to projects.

6.5 Time Taken By The Optimisations

The discussion of the implications of the results of the project are important for evaluating the likely success of applying the global approach to optimisation on larger projects. This analysis has been done on the assumption that there will be constraints on the computational resources available to the project. However, the previous analysis has not been able to evaluate how constrained the projects might be, as time was not considered. While it was important to eliminate time from the analysis in order to provide a better comparison between projects of different sizes, it is also important to consider the time taken by the different optimisations in order to evaluate how much time a real global optimisation may take up.

6.5.1 Comparison Of The 6 Different Optimisations

In choosing to compare the optimisations by runs two assumptions were made. Firstly it was assumed that running the objective function would take approximately the same amount of time with any of the 6 optimisations performed. Secondly it was assumed that the overhead associated with the genetic algorithm was small in comparison to the computational effort required to evaluate the objective function.

The validity of both of these assumptions can be evaluated by considering Table 6.10, which shows the mean duration for each of the 6 optimisations performed. It also shows by how much, on average, the run time on any project differed on from the average run time for that project.

	Run Time (μs)		Deviation From Average	
	Average	Std. Dev.	Average	Std. Dev.
PRU	430	171	-4.54	42.23
Priority	433	180	-1.81	39.83
MEST	420	175	-14.54	41.01
PRU & MEST	416	156	-18.18	45.96
PRU & Priority	455	194	20.00	46.85
All three	454	167	19.09	37.49

Table 6.10. Times of optimisations.

As the table shows, the greatest difference between the mean run times of the optimisations is approximately 38μs, between the PRU & MEST optimisation and the PRU & priority optimisation. The significance of these values could be determined by performing an analysis of variances on the results. However, an ordinary analysis of variances cannot take into account the fact that the average times of the optimisations vary considerably from one project to the next. This is reflected in the fact that the standard deviations from the mean execution time are much less than the standard deviations of the execution times. Therefore, rather than simply using the raw execution times of each case for each project, the deviation of each case's execution time from the mean of all the cases for that project will be used. If these variables are assumed to be normally distributed then they can be compared using an analysis of variances.

This analysis yields an F value of 16.3. The required value of F for rejection of the null hypothesis with 99% confidence is 3.02. Therefore it is possible to reject the null hypothesis, and assert that there is a significant difference between the mean execution times for the different cases.

While there is a significant difference between the cases, the average difference between them is less than 10% of the whole average run time of 435μs. This means that the difference in run times and the variation in overhead created by the genetic algorithm between the optimisations accounts for less than 10% of the overall run time. It is also possible to identify from Table 6.10 that the differences between the optimisations are caused by both a change in genetic algorithm overhead and variations in the objective function execution time.

The assertion that some of the variations in run time are caused by changes in the genetic algorithm overhead can be justified by considering the averages of the three sizes of feasible region. As the number of variables in the optimisation increases, the overhead required for the genetic algorithm would be expected to increase. This is simply because it must process more variables. The averages are $428\mu s$ for the single variable optimisations, $435\mu s$ for the optimisations with two variables and $454\mu s$ for the optimisation of all three variables. This shows a general trend for the run time to increase with increasing numbers of variables in the feasible region, as would be expected.

However, while this general trend may be observed, it should be remembered that the largest and smallest run times were observed for two optimisations with exactly the same number of project input variables, so this difference cannot be attributed to a variation in genetic algorithm overhead. Rather, it must be due to some variations in objective function execution time.

While the variations in run time caused by these two phenomena do not make up a substantial proportion of the overall run time, it must be recognised that variations in run time can be expected between different combinations of project input variables. It should also be recognised that the addition of project input variables to the optimisation will increase the overhead associated with the genetic algorithm, and hence increase the time taken by an optimisation of a finite number of runs. However, the potential benefits of increasing the scope for optimisation should outweigh the associated small increase in run time.

6.5.2 Variations Within The Optimisations

While the differences between the run times of the optimisations is small, there was a great difference between the execution times of different sizes of project. This is of great concern, as it would be expected that real, large projects would take much longer for one execution of the objective function. If this figure rises polynomially or even exponentially with project size then it is could render the global approach unusable, because the objective function execution time would be so large. In order

to assess how the run time increases with project size, the average execution times are plotted against the number of activities in the project. This is shown in Figure 6.32, with a power regression line obtained by the method of least squares.

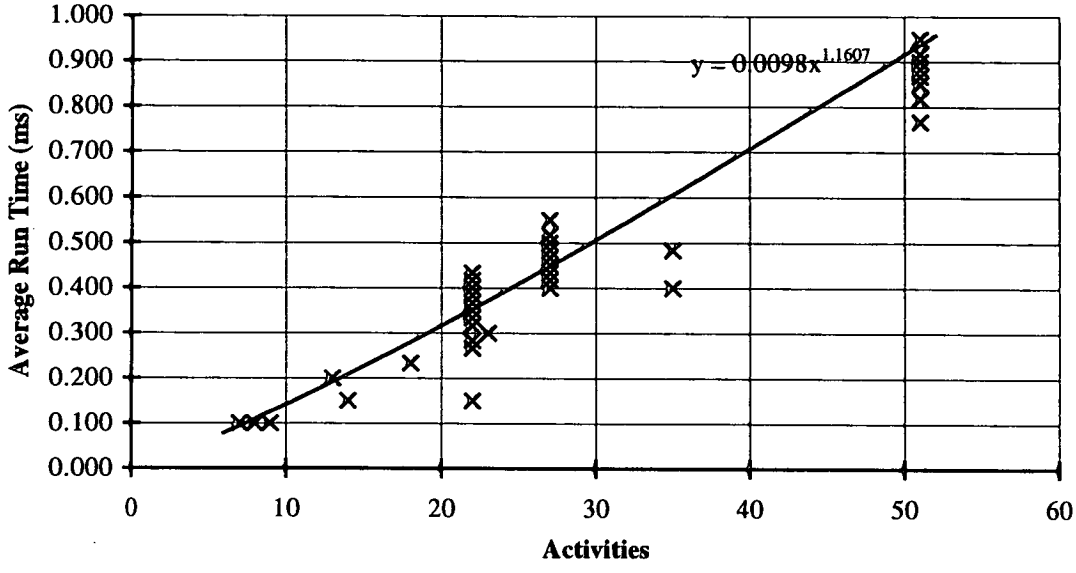


Figure 6.32. Mean run time by activities.

The equation of the best fit line is $y = 0.0098x^{1.1607}$, which is fairly close to linear. It can also be observed that for the largest set of projects most of the values are less than the value predicted by this equation. This suggests that the gradient may, in reality, be closer to linear, and the fact that the line of best fit is not is only down to scatter. This can be demonstrated by trying to fit a straight line to the data. If a straight line is fitted to the data using the method of least squares, the equation is $y = 0.01765x - 0.02459$, and the R^2 value - the proportion of the squares of the variations of the data points that can be explained to the total variation, is 0.929. This value of R^2 for this straight line actually indicates a slightly better fit than the assumed power equation, whose value of R^2 was 0.922.

However, even if the power trend were accurate then it would suggest the execution times for different sizes of project shown in Table 6.11. As this figure shows, the execution times are still realistic for a project of 10000 activities.

Activities	Run Time (ms)
100	2.05
1000	29.2
10000	416.0

Table 6.11. Predicted run times for larger projects.

This demonstrates that the increase in run time associated with the increase in project size, in activities, will not make the use of global optimisation unfeasible.

While the consideration of the number of activities is vital to assess how changes in the size of the project will affect the run time, there is a second variable which may also influence run time: the duration of the project. Unlike the number of activities, however, it is not possible to identify a single value which represents all runs of the optimisation. The variation of the project's duration is an inherent part of the optimisation. Therefore a representative duration is needed for each project. The target duration was selected for this value. While it is recognised that this will not represent that absolute average of the durations of the projects over the optimisation, it does provide a good relative measure of duration which will be consistent from project to project. The variation of the run time by target duration is shown in Figure 6.33, along with a power line fitted by the method of least squares.

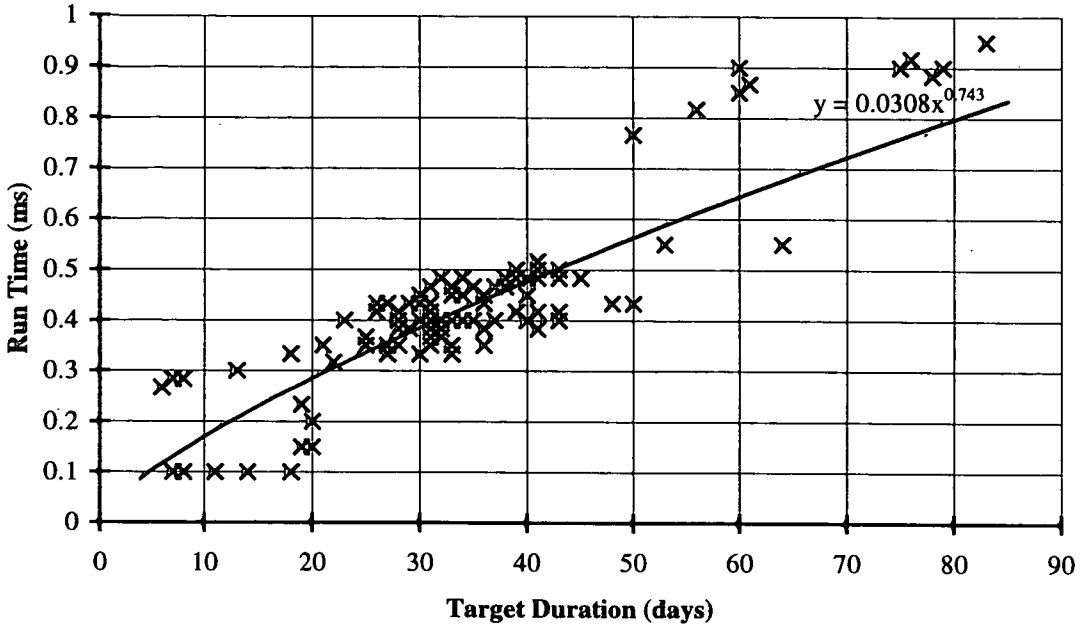


Figure 6.33. Mean run time by target durations.

The power fit from Figure 6.33 suggests that as project duration increases the run time will also increase, although the increase will not be as marked for longer durations. However, as with Figure 6.32, the line does not appear to be consistent with the largest values, suggesting that the true trend is somewhat closer to linear. This is again assessed by performing a linear regression analysis, and the equation found is $y = 0.01045x + 0.07013$, with an R^2 value of 0.806, compared to an R^2 value of 0.668 for the power fit.

This attempt to relate duration to run time has not produced as good a fit as relating run time to activities. This can be seen by the reduced R^2 values obtained with duration regression analysis. However, as larger durations tend to be found in projects with more activities, it could be argued that the observed trend for the run time to increase is simply due to the increased run time associated with an increase in activities. This can be evaluated by considering the trends of the durations for each of the four different sized groups. This is shown in Figure 6.34.

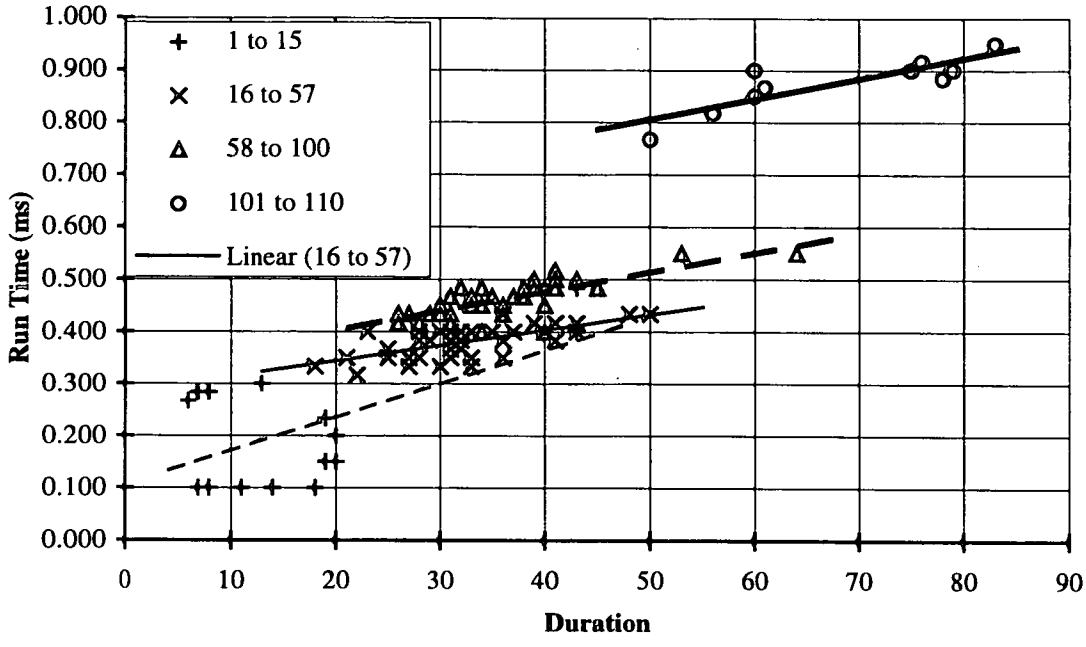


Figure 6.34. Grouped mean run time by target durations.

The first observation which may be made from this figure is that there is a definite trend for the run time to increase with increasing target duration for each group. As the last three groups constitute projects which have the same number of activities, the trend observed for these groups is independent of the number of activities. The influence of the number of activities can also be observed by the trend line appearing higher for larger groups of activities. It can also be seen in the slope of the first group, which contains projects of different sizes. In this group the number of activities is also influencing the run time, and the slope is greater.

Thus it can be seen that the run time is dependent on both the number of activities and the duration of the project. This can be quantified by performing a multiple linear regression analysis on the data. Where z is the run time in *ms*, x the number of activities and y the duration in days, the equation found by the regression is:

$$z = 0.01242x + 0.004035y - 0.02916$$

This equation has an R^2 value of 0.956, so it can be seen that this is a good fit. It not only shows that the run time is related both to the number of activities and the project

duration, but also gives some idea of how much of an effect the two factors have on the final value of the run time. For a 10000 run project which takes five years, for example, the predicted run time would be $131ms$. This shows that even for very large projects the expected run time is not prohibitive to the optimisation.

The model has been shown to cause approximately linear increases in run time with both increasing duration and increasing project size, this makes the equation from the regression analysis useful for predicting the order of magnitude of the run times for larger projects. It is not, however suitable to predict the real run times for a full scale global optimisation performed on a real project. This is because the model used is only a basic one. It does not consider any of the more complex interactions which must be considered in the modelling of a real project. Much more modelling would undoubtedly be required for a real project, to take these additional aspects into account. These increases in the complexity of the model must be expected to increase the run time of the objective function.

While the actual amount of computational effort required must be expected to be increased it should also remember that computer technology is moving forward all the time. Improvements could be made in both the speed of the program. It could also be run on a much faster machine. As the genetic algorithm requires a large number of independent executions of the objective function at one time the problem could also be parallelised.

Despite both the detrimental and advantageous factors mentioned, the run time of a real model could still be expected to be of approximately the same order of magnitude as predicted here. As the run times for the basic model are feasible for the implementation of the global model, those of the larger model run times may also be expected to be realistic.

6.6 Conclusions

Optimisation of all 110 projects was performed using the program. The duration of the optimisation was measured in executions of the objective function, or runs. Runs

were used because they would provide a fairer comparison of the performance of the optimisations between projects of differing sizes. The limiting number of runs on the optimisation was 10000. The duration of the projects at any point was expressed as a proportion of the target duration, in order to be able to equitably compare projects of differing sizes.

Six cases were analysed in all, each representing a different combinations of the three project input variables available to the optimising algorithm. In each of the first three cases only one project input variable was considered as variable, and represented local optimisations. The fourth case, which used PRU and MEST project input variables, represented the global approach. The fifth case added the priority project input variable to the fourth case, to see if any improvement could be gained by implementing a second activity selection technique.

The optimisations of all six cases produced reductions in duration for all projects except a number of smaller projects for which no reduction in duration was possible. The variation in the optimal durations found was shown to decrease as project size increased, suggesting that some kind of averaging effect was in operation.

6.6.1 Primary Resource Use

The PRU case was the most effective of the three cases with only one project input variable. It was initially supposed that the validity of the limits imposed on the PRU variables was questionable, because their values were set by project level characteristics whose limits were too broad, particularly for larger projects. This suggested that the PRU case's effectiveness was not one which could be extrapolated to large projects, because it was caused by these unrealistically high limits.

Primary resource use was found to be more effective in larger problems. The results indicated that better solutions were being obtained for larger problems, suggesting that there was a link between the unrealistic limits and the improved performance. However, further analysis suggested that there was no correlation between broad limits and increased efficiency. Therefore the hypothesis that the increased

performance over the other two individual cases was caused by the broad limits was rejected.

Nevertheless, the fact that the optimisation was able to find more efficient solutions for larger projects was of interest. It demonstrated that highly efficient solutions could be obtained by the use of PRU project input variables alone. While this effect permitted the performance of the optimisation to increase with increasing project size, the trend was expected to be reversed in larger projects due to the increase in the number of variables.

6.6.2 MEST & Priority

Two variables were proposed as activity selection techniques: priority and minimum early start time (MEST). The MEST case, whose feasible region always includes the optimum, was found to be far less efficient at finding the optimum than the priority case. This was shown to be due to the fact that it was very easy for a simple change in one of the MEST project input variable's values to cause a considerable increase in project duration. This inefficiency was confirmed through comparing the average feasibilities of both the priority and MEST optimisations. It was also shown how this effect increases dramatically with project size, making MEST a highly unsuitable project input variable for use in a real, large project.

6.6.3 PRU & MEST Optimisation

The problems with MEST project input variable increased substantially when it was optimised along with PRU. This was because the introduction of the PRU to the problem actually increased the space between the upper and lower limit on the MEST project input variables. This in turn decreased the feasibility rate. The problem also increased even more with increasing project size.

This appeared to show the philosophy of global optimisation not to work, because adding a variable which allowed inclusion of the full scope for optimisation would not allow the optimisation to perform better than the individual optimisation. However, by changing the strict philosophy proposed to one which aims to make *best use* of the full scope for optimisation, rather than making full use of it, it was shown

how the global approach could still be valid. For this simple problem making best use of the scope for optimisation appeared to be by implementing the priority project input variable in place of the MEST.

6.6.4 PRU & Priority Optimisation

The PRU & Priority optimisation was shown to have obtained significantly better values of the objective function over 10000 runs. However, this improvement over other cases is not consistent from group to group. The improvement over the PRU case was found to diminish with the group of the largest projects. It was also found that the value of the optimisation of this group was worse than the PRU case earlier in the optimisation.

This was addressed first by considering the times to target of the optimisation. However, these were found only to reflect the average progress results. Nevertheless, it did show that the time to target was reduced with increasing project size. This demonstrated that the optimal solutions tended to be further away from the target value, as with PRU

As with the PRU optimisation, it was first shown that this phenomenon was not linked to the high limits applied to the PRU project input variables. Then it was attempted to link the performance to the scope for further optimisation, and it was found that when the scope for further optimisation introduced by adding the priority variable was small the improvement of the PRU & priority optimisation was correspondingly smaller. This explained the diminished improvement of the last largest group, although it failed to allow any firm predictions on what this improvement would be for very large projects. Nevertheless it showed that the results of this simple application were positive. If the increase in the scope for optimisation of real projects obtained by implementing the global approach is large, as it is expected to be, then the global optimisation should be able to yield a performance which improves upon the local techniques currently employed.

It was also shown from the results that optimisation will tend to require more effort to obtain the optimal solution as project size and number of variables increases. As

this is unrealistic for real projects the need for a method which makes the best use of the scope for optimisation with limited computational resources was demonstrated.

6.6.5 Times Of Optimisations

The times of the 6 different optimisations were found to vary by less than 10% between the fastest and slowest optimisations, which were the PRU & MEST optimisation and the PRU & priority optimisation respectively. It was shown that the variations were caused partly by variations in the overhead of the genetic algorithm, and also variations in the execution time of the objective function.

The run times were shown to vary both with project size and project duration, with an increase in either causing an increase in run time. Furthermore, a multiple regression analysis was carried out on this data, and an equation relating project duration, the number of activities to the run time was determined. It was found to fit the data quite well. This equation, by predicting the times of the basic model, could help predict the order of magnitude of the run time required for larger projects. However, it was also recognised that a full blown optimisation would almost certainly require a more complex model, which would require more computational effort to execute than the basic model implemented in this chapter. It was also recognised that improvements of the program, computer technology and the fact that the code could easily be parallelised would also affect the time taken by the optimisation.

However, the order of magnitude of the times required for such an optimisation were determined and shown to be reasonable. It has already been shown by the results of the optimisations themselves that better results can be expected by using the global approach. Therefore it is possible to assert that this simple application has demonstrated the advantages of applying the global approach.

Chapter 7 Application Of The Global Model To A Real Project

The global approach to optimisation has now been developed. A general model has been described, and from that a specific model for the solution of the resource constrained project scheduling problem has been developed. The model has been implemented on 110 small sample projects to show that the global approach can provide better solutions. This has demonstrated that using the global approach expands the scope for optimisation, allowing many possible solutions to be considered which were not considered by local techniques. It has also demonstrated that this increase in scope, permits optimisation using the global approach to obtain better solutions than local optimisations.

Therefore, in this chapter the global model is applied to the construction phase of a real project.

7.1 Car Park At Johannesburg International Airport

The project to be studied is the construction of the new car park at Johannesburg International Airport. In recent years the number of passengers travelling through the airport has increased greatly, and is expected to continue rising over the next twenty to thirty years. Partly in anticipation of this rise, and partly in response to already congested road and parking system the Airports Company of South Africa (ACSA) decided to fully upgrade the terminal, access and parking at the airport.

Part of this plan involved the upgrade of existing parking facilities to a completely new 1,144 bay car park. Schneid Isrealite and Partners (SIP), a South African project management firm, were contracted to manage the project. The main contract commenced in March 1998 and is expected to be completed in August 1999.

A 4th Year student at the department, Harry Fleetwood-Bird, worked on the project in the Easter Holidays of 1998, and used his findings for the compilation of his final year thesis, which was submitted in June of 1998. As well as this he obtained

permission for the project data to be taken away and used for this research, which was performed by myself.

The results of this work are presented in this chapter.

7.2 Extensions To The Project Model

The extensions to the model which were carried out in the previous chapter are retained. These include activity priority and MEST activity selection techniques, as well as the implementation of primary resources. However, in order to model this project it is necessary to extend the model further.

7.2.1 Project Output Criteria

There were two project output criteria which were important to the project. The first of these was cost. Reducing the overall cost of the project could have saved both SIP and the contractor money. Therefore cost was of sufficient importance to become a project output criterion.

The second criterion was time. It was required that the project be completed by the 12th of August 1999. Failure to do so would result in the loss of earnings from the car park, as well as disruption to public services.

7.2.2 Project Input Data

The project input data was not available in one single tidy file, unlike the data for the sample projects in the previous chapter. Some of the data was supplied with a project network, which included some resource information. This was the schedule used by the project managers to monitor project progress and predict resource consumption. The rest of the information was in reports of various kinds. Harry Fleetwood-Bird compiled much of this information into his final year thesis[Fleetwood-Bird, 1998].

In addition to this the schedule data supplied did not cover the entire construction phase of the project. Although it went as far as the handover, the schedule data provided only began after the piling had been completed. This was because the

schedule was never intended to be a schedule of the whole project, but was created to model the construction of the structure once the piling was complete in order to facilitate the planning of that phase of the work. Therefore, the activities from earlier in the project were not available.

Because of the constraints on the model it was also important to be able to observe the consumption of concrete, and determine to what types of activity it was going. Therefore concrete consumption was also included in the model. Other resources, however, were not included. Only the activities included in this schedule are to be optimised. Additional project data must be calculated beforehand using these figures with data from the quantity surveyor's report[McKintosh Latilla Carrier & Laing, 1997].

7.2.2.1 Activities and Relationships

The activities in the data set begin from the trimming of the pile caps, and finish with the commissioning and handover of the completed car park. All activities were linked to other activities by precedence relationships. The durations were all expressed as a whole number of days, and were therefore assumed to be integers.

The relationships were set up for a precedence networking model, and not a resource constrained scheduling model. Therefore while the network calculation algorithm which had been incorporated into the model was capable of obtaining a project duration which corresponded with that determined by the project planner, the execution of the scheduling algorithm with no resource constraints yielded a much greater duration. After analysing this unexpected result, it was discovered that the delays which caused this increase in duration, while arising in different types of activity throughout the project, were all caused by the same phenomenon. They all arose in activities which were, according to the network logic, intended to start before activities upon which they were dependent. These activities usually had long durations, and their predecessors short ones. The relationship was always a finish-finish relationship. Thus, when the scheduling logic was used, it arrived at the day on which the activity was due to be scheduled. As one of its predecessors was yet to be scheduled, it could not be added to the queue.

If the activity had been scheduled then the predecessor would still have been scheduled in time for the activity to finish after it. However, because a time based scheduling algorithm was employed, it was not permitted to look ahead in this way to see that its predecessor would be scheduled in time. Therefore the activity was only added to the queue and scheduled after the preceding activity had been scheduled, meaning that the activity was scheduled at a later date than it would otherwise have been. This led to both itself and the project being delayed.

The fact that the algorithm was not permitted to look ahead to discover that it would have been able to schedule this dependent activity may be considered as a weakness in the scheduling algorithm as it has been implemented. However, in order to establish that the predecessor will be scheduled later, the algorithm must not only observe the network logic, but also ensure that sufficient resources will be available to schedule the predecessor when it is added to the queue. This is a complex problem which could be quite computationally demanding. It also represents a move from time based scheduling towards activity based scheduling, the drawbacks of which have already been discussed in detail in chapter 4.

Attempting to make any such changes to the scheduling algorithm was therefore highly undesirable. In order to attempt to find another solution to the problem, the relationships which caused these delays were scrutinised carefully. It was discovered that they had all arisen out of a network structure which had not been set up to be representative of the actual relationships which existed in the real project, but to maintain a structure that the project planner envisaged it should have. Therefore alternative relationships and activity breakdowns were implemented which solved the problem while maintaining the accuracy of the relationships.

The delays and solutions were as follows:

- The casting of the perimeter footing of the building was required to finish at the same time as the trimming of the pilecaps in order for subsequent activities to run properly. In particular it was vital that the footing was started before either the base of the flower boxes or the perimeter wall could be started, and completed

before they were completed. It was also essential that it was complete before the construction of the base of the service tunnel could begin. As all the activities were directly or indirectly preceded by the trimming of the pilecaps, a finish-finish relationship was applied such that the trimming could not be completed until the footing was completed. This, in the original model, indirectly ensured that none of the activities which were dependent upon the completion of the footing could start until the footing was complete. However, as the footing took 4 days and the trimming 7, this resulted in a delay to the block of 3 days when the scheduling algorithm was employed.

The solution to this problem was twofold. Firstly, the finish to finish relationships with the trimming of the pilecaps were removed. Then a number of relationships were added, with the appropriate lags, between those activities which were dependent upon the footing activity, and the footing activity itself. This allowed the fact that certain activities were dependent on the construction of the footing to be modelled more explicitly.

- The casting of the concrete surface beds could not be completed until one day after the backfilling of the service tunnel was complete. This was because the service tunnel ran underneath two of the four beds. The duration of the backfilling operation was only two days, while the casting of the four surface beds took four. Thus it was possible to cast the first two beds (A & B) without the backfilling having been completed, then to cast the third slab (C) on the second day of backfilling, after the area under the third slab had been backfilled, and to cast slab D after the backfilling had been completed. In this scheme the casting of the beds was due to start a day earlier than the backfilling of the tunnel. However, the scheduling algorithm would only schedule the casting on the day when the backfilling began at the earliest.

The solution was to split the casting of the four surface beds into two activities, the first comprising the beds A and B, and the second C and D. Thus the finish to finish relationship became two relationships: a start to start and finish to finish relationship, both with a lag of one, from the backfilling to beds C and D. This was

more explicit than the original finish to finish relationship to the activity representing all four beds.

- The floor slabs in the upper floors were divided into three equally sized sections, A, B and C, as shown in Figure 7.1. A and B were cast first, and then post-tensioned before slab C could be cast. The placing of the electrical ducts, reinforcement bars and post-tension cables in preparation for the pouring of the third section C had a precedence relationship such that this activity could not be completed until the post-tensioning of slabs A and B was completed. However, the post-tensioning only took 1 day, whereas the placing of the ducts, bars and cables took 3. The finish to finish relationship determined that they were to be completed at the same time, and this lead to a delay of 2 days, because the placing of the ducts, bars and cables could only be commenced once its predecessor, the post-tensioning, had been scheduled.

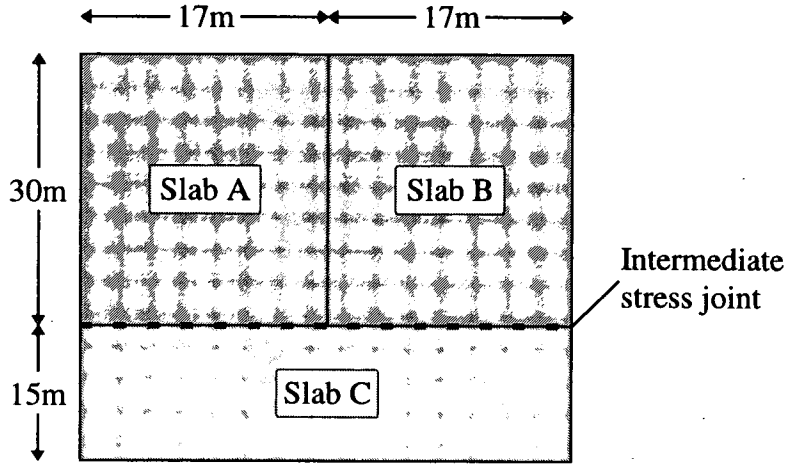


Figure 7.1. Casting of the upper floor slabs.

The solution of this problem was more complicated than the previous ones. The relationship is required because the placing of the ducts, bars and cables cannot be finished until the post-tensioning is complete. However, unlike the casting of the slabs there is no tidy way to break the activity down. The placing of the post tensioning cables and anchoring them to the cables in slabs A and B will only be performed in the last day of the activity. However, representing only the placing of the cables as a separate activity is also not valid, as work on the bars and ducts will

also take place on this day, and will be performed by the same crew. Therefore, it would not be possible to accurately assign the resources to the activities. This meant that it was necessary to break the activity into two activities as follows. The first activity involves most of the placement of the bars and ducts, and lasts two days. The second involves the completion of this work on the duct and bars as well as the placing and anchoring of the cables. The implementation of this solution did constitute a slight increase in the realism of the model, because it showed how it was only the final phase of the placing of the ducts, bars and cables which was dependent upon the stressing of the slab.

Once all the corrections to the data for the relationships had been made there were 1252 activities in the data set, and 1797 relationships.

7.2.2.2 Resources

The resource information within the project data set was found to be quite restricted both in availability and scope. Firstly not all the resource types actually used by the project were included in the data. This was because the only resource requirements which had been added were those which the project planner felt would be constrained in some way, and were therefore required in the project model in order to monitor their consumption as the project progressed. These resources were those relating to the placing of the shuttering and to the pouring of concrete. No other resources were supplied as they had not been considered by the planner, and, therefore, could not be modelled within the project. Nevertheless, these two resources were the most constrained in the project and hence the ones most likely to generate conflicts.

In order to use this data it must therefore be assumed that the other resources would not generate any significant conflict within the project, nor would the cost of these resources vary significantly under the kind of differing conditions created during the optimisation. This, of course, may be a very unrealistic assumption. Nevertheless, it will still permit the effectiveness of the global approach to be demonstrated on the data available.

The second problem with the resources was that they were found not to be truly representative of the resources they were intended to represent in the project. For any given activity the resources had been amalgamated into one resource type. For example, with activities which involved the pouring of concrete for the concrete beds, there was a single resource "CS", which both represented and cost the manpower, concrete materials, batch plant and concrete pump within one resource, and was measured in terms of how much concrete was required. This cost per m^3 of concrete pour had been calculated by the quantity surveyor, and so when the schedule was put together, this single resource was used.

The report from the quantity surveyor did provide a breakdown of some of these resources although it did not determine how they were divided up between different types of activity. Some of these resources could obviously be applied to one particular type of activity. Reinforcement costs, for example, will only be applicable to the activities which involve the placing of the reinforcement. Therefore the total reinforcement could be distributed among the activities. The amount of reinforcement applied to an activity would be proportional to the amount of concrete which would subsequently be poured to the which that activity operated in. However, it was not possible to break up all resources in this way. In addition to this the costs of each resource were simply the cost of procuring and installing that resource, which meant that the most important resources to the project model – manpower resources – were not available.

While this approach permits the costing of the activities to be done quite easily, it is not a realistic model of what actually takes place when the building is constructed. If the bill of quantities for that activity represents the amount of reinforcement, then the amount to be prepared and placed per day reduces to the total amount required divided by the duration. This value could easily correspond to a non-integer manpower requirement, which is impossible. When this occurs in reality it is usual for the contractor to round this figure up, in order that the number of men on the activity will be sufficient to complete that activity within the required duration. This means that more men are assigned to the resource than the number specified by this

calculation. This would result in the activity costing more. If, conversely, it is rounded down then there will be insufficient manpower to complete the activity in the required time and the activity will be delayed. This could easily result in the delaying of the project. Therefore such an amalgamation, while useful for early project cost estimation and accounting purposes, is not useful for accurate modelling of the project.

Despite the inappropriateness of this resource costing and representation, it was the only information available within the project documentation provided which would allow the breakdown of the resources into their component manpower and materials. However, Harry Fleetwood-Bird had also been able to obtain the manpower team structures for two activity types: the placement of the support work and decking for the floor slabs and the pouring of the slabs. Therefore it was possible to assign manpower and materials to those activity types based on the amount of concrete to be cast.

These team structures are shown in Table 7.1, along with their daily capacities.

Activity	Max. work per day	Team Structure	Cost per team per hour	Working hours per day
Concrete pour	200m^3	Pipework and pump: 5 labourers Batch plant: 2 operators Pour: 4 labourers, 1 operator	R129.45	10
Support work and decking	300m^2	1 foreman, 14 shutter hands	R298.44	8

Table 7.1. Team strucures and capacities.

While these rates are useful for calculating the cost per hour of the teams and determining the capacity of those teams, they do not actually show how much work a team is typically able to perform per day. The support work and decking rate shown is the rate which the contractor felt would be the highest attainable. Therefore it should follow that the amount of shuttering, in m^2, which can be placed by each shutter hand each day should be:

$$\frac{300}{14} = 21.43\,\text{m}^2$$

The concrete pour had a definite physical limit of $200m^3$ per day. This was restricted by both the amount of concrete which could be produced by the batch plant in a day and the amount of concrete which could realistically be handled by that size of team. However, the contractor maintained that although the capacity of the batch plant was $200m^3$ per day, the most concrete pour that his men were able to achieve was $120m^3$ per day. This was due to the fact that interruptions in the pour were required because pipework must be connected and cleaned etc., meaning that the batch plant could not be used as efficiently.

The implications of the imposed limits on the validity of the model will be discussed more fully in section 7.2.2.4.

7.2.2.3 Costs

The rates shown in Table 7.1 enabled the manpower cost per day to be calculated. As the cost of the concrete and shuttering including the manpower can be calculated from the bill of quantities, it is possible to break the resource down into its component costs, including manpower.

The simplest way to obtain these costs was to divide the cost per team per day by its capacity, thereby obtaining a cost per m^3 of concrete, or per m^2 of support work and decking. However, this method assumes that the team would be performing at capacity every day. This, as explained in 7.2.2.1 is not the case. Most of the time the team would not be required to operate at capacity. However, this does not necessarily mean that the team would either work fewer hours or be paid less. Therefore any reduction in the work to be completed in a day could not be expected to yield a corresponding reduction in the number of hours worked.

On these grounds it was simply assumed that the team would always be working a full day, irrespective of how much work was actually required. This assumption would normally be valid provided the amount of work required each day did not drop too low. How low this value would be and what subsequent reductions in duration might be achieved was, however, difficult to discern.

The cost rates applied by the quantity surveyor are based on historical projects, and would therefore inherently take into account such fluctuations in work rate, as this phenomenon will not be exclusive to this project. This means that the quantity surveyor's figures can be used to calculate the total cost of both the shuttering and the slab pour. Therefore, in order to obtain the cost of both the concrete and the support work and decking excluding the manpower costs, the total manpower cost, as calculated by the scheduling algorithm, should be deducted from the total cost of the resource. If this new resource cost is then divided by the total amount of the resource used, the cost per unit of the materials is obtained. These figures are shown in Table 7.2.

	Total cost	Per Unit	Units
Concrete Slabs (total)	**R 7,965,760**	**R 310.00**	**25696**
Labourers	R1,056,084	R124.70	8469
Operators	R280,252	R182.10	1539
Concrete	R6,629,424	R257.99	25696
Support Work and Decking (total)	**R 5,583,955**	**R 65.00**	**85907**
Foremen	R 125,171	R 168.24	744
Shutter hands	R 1,517,403	R 145.68	10416
Materials	R 3,941,382	R 45.88	85907

Table 7.2. Corrected resource costs and unit rates for concrete slabs and support work.

7.2.2.4 Constraints

Pouring of concrete was limited to $20m^3/h$ by the nature of the batch plant used. As the plant was to be used in a 10 hour day, this corresponds to a daily rate of $200m^3$. This means that the maximum amount of concrete available to all activities in any one day was $200m^3$.

In addition to this was the problem that the contractor did not feel that a rate of over $120m^3$ could be maintained by the team doing the concrete pour. The fact that the contractor felt that this was the most they could maintain, rather than the most they could achieve, made a hard limit on the capacity of the team slightly harder to define. If a limit of $120m^3$ per day were applied, then it would mean that some activities would not be scheduled. 84 of the 191 activities which used the concrete pour teams required over $120m^3$ of concrete to be poured per day. It is the average consumption

of the project which is constrained to $120m^3$, and which the project as supplied comes under, with an average consumption of $111m^3$. It is therefore only necessary to consider this ceiling within the model if the daily consumption is permitted to change. This will be discussed further in section 7.2.3.1.

Space was also highly constrained. The site area is 27, $415m^2$ and of that the free area remains at about $3{,}000m^2$ throughout the project. This very low amount of free space would need to be very carefully managed in order to ensure that there would always be sufficient space for activities and resources vital to the smooth running of the project. While this was not considered within the project schedule in any way, nor by any other analytical method, it had been identified as a potential problem by SIP.

7.2.2.5 Data Not Available

Although the data provided represented a large amount of information, much data which might have been included was absent. This was not because the data had been lost or withheld, but simply because no attempt had been made to determine it. The absence of a proper resource breakdown for any resources not relating to concrete pour or support work and decking has already been covered in section 7.2.2.2. However, some other important data was also absent.

The absence of any data concerning the space constraints has already been mentioned in section 7.2.2.4. In addition to this, the schedule data was found not to represent the whole construction stage. None of the activities concerning the early stages of the construction were supplied. The project network began from the trimming of the pile caps. As with the absence of most of the resource data, the reason for its absence was that it was not viewed as being very constrained. Only the phases which contained activities relating to the pouring of the concrete were considered, other phases were not felt by SIP to be of great importance, and were therefore not required to be modelled in any detail.

7.2.3 Analytical Techniques

The analytical techniques from the RCPS problems were retained. However, further analytical techniques were required in order to model the project more accurately. These relate to the way resources are handled within activities, and the subsequent costs of those resources.

7.2.3.1 Resources

The resources used in the previous chapter were all renewable. However, the resources in this project were not all renewable. The materials for the two resources available for consideration – concrete and support work materials – are non-renewable. Indeed, concrete is doubly constrained, being non-renewable and also having a maximum consumption per day. However, there are no data which enable the special requirements of a non-renewable resource to be taken into account. There are no delivery schedules, no inventories, and no ultimate limits on their consumption over the duration of the project. Therefore, while these resources are non-renewable, it will make no difference to the accuracy of the model if these resources are considered as renewable. In either case modelling the resource will only amount to counting how many units of each resource are used each day and keeping a running total, and, for the concrete pour, ensuring that the consumption of the resource per day does not rise above the maximum.

In addition to these two resources, it is also important to consider how the different manpower resources will be implemented. There are four types of manpower in use in this project: foremen, shutter hands, operators, and labourers.

In the RCPS problems, the resources had a total requirement which had to be fulfilled. Therefore, the minimum allocation of manpower per day which would achieve that requirement was used. However, the foremen cannot be considered in this way. They are simply present to direct the labour in order to ensure that the work is done efficiently and to the required standard. While the number of foremen required to supervise a large group of labourers may be greater than that required for a smaller group, the number of foremen is not directly related to the duration of the

activity in the way the resources in the RCPS problem were. Indeed, only one foreman is required for any of the activities to be considered in this project. The foremen must therefore be assigned at a constant fixed rate which is independent of the activity's duration.

In a similar way, there is only one operator on the concrete pour, who is simply required to perform the actual pouring of the concrete from the pipework. Only one operator is required to do this, irrespective of the amount of pouring required. Therefore the concrete pour operator must be assigned to the activity at a constant rate of 1, exactly as the foreman was.

The foremen, together with the shutter hands, form a team which performs the support work and decking activities. Because they form a team of constant size which moves from activity to activity, the number of operators are also supplied to the project at a constant rate, irrespective of whether the activity might be completed in the same time with less men.

It might be argued that this is an inefficient way to progress, and that it would be better to vary the team size depending on the activity. However, this is not necessarily the case, as changing the team size can have implications for the efficiency of that team. In addition to this, in order to implement such team size changes, a fixed rate at which the team may operate must be established. The algorithm which determines the team size would then be able to adjust the team size so that it would consistently operate as close to that level as possible. This level would be very difficult to find. If the maximum rate were used then the teams would be required to operate at their maximum rate all the time. However, the men cannot be expected to operate at maximum efficiency for any length of time, as has already been discussed in section 7.2.2.2. Therefore a lower rate must be used, in which case the fact that the team is capable of operating at a higher rate will never be taken into account. This, in itself, could also carry cost penalties because as the team will never work at maximum efficiency.

In an ideal situation, it would be possible to ask the contractor for how long the team might be capable of working at maximum efficiency. It would also be possible for him to identify under what conditions of prolonged low activity might the team be reduced in size in order to reduce its cost. Indeed, he might also be able to suggest how any analytical techniques required both to identify that situation and to make appropriate changes to team size might be set up, in order that the implementation of those techniques might avoid carrying cost penalties of its own. However, it was not possible to do this. Therefore, as was discussed in section 7.2.2.2, the maximum must be used, and the optimal solution monitored to ensure that the optimisation has not created a situation where the teams are required to work too hard for too long.

The labourers on the concrete pump and pour are very similar in behaviour to the shutter hands in this respect, as was mentioned in section 7.2.2.2, and hence will be assigned to activities at maximum, whatever the amount of work required by the activity.

The batch plant operators are similar to the shutter hands in that they will always be required to operate the batch plant, irrespective of whether the batch plant is operating at capacity. Indeed, the argument that it would be better for there to be fewer operators at lower levels of activity is invalid because the batch plant cannot physically be operated by less than two operators. However, it is not possible to assign this resource to one activity at the maximum level, as the concrete that the batch plant produces might go to several activities in the same day. If this resource were assigned to an activity at the maximum rate then it would be impossible for the scheduling logic to schedule this activity alongside another which operates at the same time. The scheduling logic must therefore consider this resource as operating at a rate proportional to the amount of concrete required by the activity.

While this permits the scheduling logic to operate properly, it must be remembered that there will always be 2 operators on the batch plant whenever the batch plant is required, and post-scheduling algorithms must take this into account.

7.2.3.2 Cost

In order to evaluate cost it is necessary to determine how the model will take the use of the resources and obtain a final cost for that resource. As most of the costs associated with the project are not available to the model, only the resources for which data exists may be cost. While this will not constitute the total cost of the project, it will nevertheless provide a cost of the resources under consideration, which will provide an opportunity to show the effectiveness of the global approach on this restricted problem. Therefore some means of calculating the cost of each resource is required. This cost can then be added to the costs of the other resources to obtain the total cost of the resources modelled.

All of the evaluations of total resource cost are the sum of the resource cost for each day. However, obtaining the cost of the resource each day may not be as simple as simply multiplying use of the resource by the consumption. Indeed, four different models of the cost of a day's resources have been identified: as used, at ceiling as used, at ceiling for project and at ceiling from first use to last use.

Costing the resource as it is used is the simplest of these algorithms. The resource cost is obtained by multiplying the number of units of the resource used by the cost per unit. This straightforward calculation is suitable for non-renewable resources and most renewable resources. Almost all the resources may be cost in this way: concrete; shuttering materials; foremen; shutter hands; labourers and the concrete pour operator. However, the batch plant operators cannot be cost in this way, as the values provided by the scheduling algorithm will usually be less than the two operators required, as was explained in section 7.2.3.1. Therefore they must be cost at ceiling for any day in which the batch plant is used.

Costing at ceiling as used is implemented only under certain circumstances. These circumstances exist where, in order to make use of a renewable resource, a certain amount of that resource must be brought on to site. When this is the case the amount of the resource which will be paid for on any day which the resource is used is the ceiling amount, regardless of whether or not the resource use is at ceiling. The batch plant operators are a good example of a resource suitable for this kind of costing.

When they are required there must be two of them operating the plant, and when they are not they need not be paid for.

Another common situation is where a resource must be cost at the ceiling for the project duration. This is required when a resource must be paid for in full to be made available for the entire duration of the project, irrespective of whether or not it is used in full, or even at all, every time period. For example, if extra storage space is required for plant and materials on a highly constrained site then it may be possible to rent part of a nearby plot of vacant land for the entire duration of the project. Therefore, the rental of this space, which would act as a constrained renewable resource, must be cost for the entire project. The algorithm would therefore cost the resource as being at ceiling at every day of the project.

If it is not absolutely necessary that a resource be available for the entire duration of the project, but must be available the first time it is used, and remain available until the last time it is used then it can be cost from first use to last use. For the example used for the costing at ceiling for the duration of the project, if the excess space were only required from midway in the project to the end, then it may be possible not to rent it until it is required. Then the resource could be cost from first use to last use, which would be cheaper than renting it for the entire project. Therefore, for this costing method, any day between first and last days on which the resource is used should be cost at the ceiling.

While these last two examples are useful definitions, only costing as used and costing at ceiling as used will be implemented in this model.

7.2.4 Project Input Variables

The number of project input variables available was quite limited. There are two reasons for this. The first is that not all the data which might have been included in the model of the project was available for inclusion. If considerations of the other resources, including the amount of space available on the site, had been included in the data set then it might have been possible to identify additional project input

variables relating to some or all of these additional resources. The absence of such data prevented this.

However, the fact that not all aspects of the project were modelled was not the primary reason for the limited scope in variables. The selection and constraining of the project input variables requires an intimate knowledge of the project. Whether any input to the model is variable or fixed cannot usually be implied directly from the data. Neither can the limits on that value if it is variable. Their identification requires an understanding of the nature of the real activity or process that a variable or combination of variables seeks to represent, including how it is affected by the operation of the project as a whole and the organisations working on it. While in some cases there may be obvious physical limits on these values which are easily derived from the physical nature of what the variable represents, in other cases such obvious technological limits might not exist and the limits may have to be determined by professional judgement.

There were two project input variables implemented on the RCPS problems in the previous chapter which were shown to be suitable for use in the optimisation of RCPS projects: PRU and priority. Both of these are applicable to this project. However, the way they were applied to the simple RCPS problems in the previous chapter may not be sufficient. Therefore the two will be examined in detail.

7.2.4.1 Activity Priority

Priority was shown to be suitable for use in a global optimisation model, and is therefore retained in the same form as it was implemented in the previous chapter. In this project, the scheduling algorithm will still be required to order the queue in descending order of priority in the same way as it did for the RCPS problems. Therefore, it is retained in the same form.

7.2.4.2 Primary Resource Use

The selection of the primary resource and the limits on that resource were made using rules designed to represent reality as closely as possible with projects for which it was not possible to identify real limits. However, in this project it is possible to

select the primary resource and its limits in a way that models the real behaviour of the activity. Some of the activities will have a primary resource that dictates the duration of that activity, although it may not be practical in reality to allow this primary resource to be varied. In these cases, the primary resource will not be a project input variable. This differs from the sample projects used in the previous chapter where all activities were assigned a variable primary resource. Similarly the setting of limits on the primary resource use project input variables must be based on the real upper and lower limits of those resources for that activity. Therefore determining the project input variables for primary resource use must consider first which of the primary resources identified for different activities are variable and which are not. Subsequently limits must be defined for those that are.

In this project there is a large set of activities which all have a primary resource. However, only two types of primary resource have been identified: the support work and decking, and the concrete pour. In order to identify which primary resources are variable it is necessary to consider what the varying of the primary resource means in reality. Varying the primary resource, which is manpower for both of these types of activity, means varying the size of the team. For reasons outlined in section 7.2.3.1 this varying cannot be done on an activity by activity basis. The team size should remain constant throughout the project. Therefore, the problem of defining the primary resource project input variables reduces to one of determining whether or not the team size is variable, and if so what the limits are on its variation.

The labourer team size for the concrete pour defines how much concrete can realistically be poured in a day. However, the size of each day's pour is set and has already been incorporated into the design, as all joins in the concrete must be designed for. Therefore to change the size of the pour is not possible.

There are two possible effects of changing the team size. Firstly, it can result in a change in the duration of the activity. This flexibility of activity duration can be used to schedule the project more efficiently in terms of time, cost, or some other project output criterion. It is clear that because the pour sizes cannot be changed, the activity's duration cannot be changed either. The team is required to perform one

and only one of the pours involved in the activity per day. Of course, had this been considered in the design phase then it might have been possible to provide a number of possible layouts which gave the activities different durations. However, the project is no longer at the design phase and therefore cannot be changed in this way.

The second advantage in changing the team size is that it might be possible to reduce the number of men on the team, while keeping the activity duration constant and therefore reducing the manpower cost. However, the smallest possible reduction in the number of labourers is from 5 to 4. If the maximum average pour size for 5 labourers is $120m^3$ per day, and a constant rate of team efficiency is assumed (i.e. the capacity of 4 men will be 4/5 the capacity of 5 men), then the maximum average pour size for 4 labourers should be $96m^3$ per day. This is much less than the $111m^3$ required to execute the activities in their current form. Therefore the number of labourers cannot be varied in this case either. Therefore it must be concluded that the labourer team size is not variable. Hence there are no project input variables for the concrete pour activities' primary resource use.

The other type of activity to which a primary resource project input variable might be assigned is the support work and decking activities. The team size for this activity is not completely fixed. Unlike the concrete pour there are no physical constraints on how much work may be done each day. Changing the team size will simply change the amount of support work which can be erected in a day.

However, while it is quite straightforward to identify the primary resource use as variable, it is difficult to see what limits might be applied to the team size. The actual size and operational capacity of the team was decided by the judgement of SIP. This was based on experience of typical team sizes and their capacities from previous projects. There is no indication from this data what limits, if any, might be applied to the team size.

While this is true, it is important to consider why these levels have become the accepted norm. It is possible that the figure has just been "the way it is done", and has never really been questioned. However, it is more likely that engineers have

accepted that team size because they have observed it working efficiently. If this explanation is accepted then it is clear that moving too far away from this team size is likely to result in loss of efficiency in terms of man days per m^2 of support work erected. If the drop in efficiency is large, then it is unlikely that the optimal solution will lie here, as this part of the project is not operating very efficiently, and subsystems are all expected to operate efficiently at the optimal solution.

Therefore it is necessary to set limits which ensure that the team size does not vary too much from the value of 14 set by SIP. This can be expected to be particularly so when increasing the team size. Because projects are almost always constrained by time, projects will usually have a number of men which enables most activities to be done quickly. If this is the case then it is possible that the team size is such that to increase it by much more will result in a considerable drop in efficiency because there are simply too many people working on the activity. It is certainly true that this project is highly time constrained, and that SIP will have considered this when planning the project. Therefore it is possible that the team size is very close to the maximum value attainable before a considerable loss of efficiency arises.

If reductions in the team size are considered in this light, then it would seem that there would be more scope for reduction in the team size. The primary reason for loss of efficiency is overcrowding[Mosheli, 1993(2)]. The reason a minimum crew size exists is usually because it is not physically possible to operate the activity with fewer than a certain number of men. Therefore it is likely that the minimum team size is substantially less than the size set by SIP.

Of course without being able to contact the contractor and ask what he thought the maximum and minimum team sizes would be it is impossible to say for certain that what has been argued about the limits on the team sizes is correct. Nevertheless, it is not possible to do this, so it will have to be assumed that the arguments are correct.

However, this acceptance still does not yield actual values for the upper and lower limits. It is necessary to simply choose two limits bearing in mind the arguments raised. Therefore the lower and upper limits on team size were set at 7 and 16 men

respectively. The upper limit represents only a 14% increase in the team size, and is therefore does not go too far above the team size set. The lower limit is much smaller than the initial team size, but still has 7 men on the activity, which, it is assumed, will still be sufficient to carry out the activity.

7.2.5 Optimisation Of The Model

Before the optimisation could commence, the project was set up with the limits and team sizes of the initial conditions and an initial schedule obtained. It was found that the schedule with all the resource constraints applied went from 352 days (the calculated duration) to 360 days. This was expected as the initial schedule took no account of any possible resource conflicts. Indeed, the resource histograms supplied by SIP from the network model clearly indicated days when the resource consumption did exceed the maximum. However, when the primary resources were assigned to the support work activities, the project duration went down to 349 days. The reason for this was that the assumption that the capacity of the support work teams remained constant at $300m^2$ per day was not consistent with the project data. Many of the activities had durations specified in the data which should have been considerably lower had the assumption held. The most extreme case was the support work for the lift and services shafts, whose activities had a duration of 5 days, despite requiring only $78m^2$ of support work. There are two possible explanations. Either the estimator's assumptions for the selection of activity durations were flawed, or the amount of support work which the team could place in a day varied between different types of activity. The first possibility could have occurred if the estimator had set the durations based on previous experience without considering the size of team for this project. The second could have been the result of the fact that the rate of work of the teams was based on their performance on the main floor slabs.

The first possibility is consistent with the fact that the durations were determined by the estimator, while the maximum efficiency was supplied by the contractor. It is possible that the estimator failed to consider the team sizes when determining the activity duration. However, while this may explain the slight inconsistencies in the slab pour durations among the main slabs, it does not explain the extremely low

values of team efficiency on the lift and service shafts, which, at under 26m^2 per day for the whole crew, are less than a tenth of the assumed efficiency. Therefore it is also likely that there are substantial differences in the crew work rates for different types of activity. The support work for the slabs were for a simple, large rectangular slab. However, not all the support work would be quite so simple. The concrete ramps involved slopes and curved surfaces, and the service shaft concrete was vertical, as well as having to provide shear resistance for wind loading. Therefore it must be expected that the support work for these slabs would take longer to place. This suggests that while the criticisms surrounding the selection of activity durations might be valid, it is not the only contributing factor to the discrepancy between the rates suggested by the schedule and those expected.

In order to address this, it was decided to optimise three different cases.

The first was the case already discussed, where the support work team efficiency was considered to be constant across all the activities.

The second was where the support work team efficiencies for all the different types of unit, eight in all, were considered separately. This included the support work for four different types of slab pour A & B for the main sections, the slab pour C, the slab pours for the area which could not be accessed until the main sections of the car park were completed (section E), the service and lift shafts, and the concrete ramps. This assumed that changes in efficiency were due to solely to differences between the major groups of activities. The maximum team size was determined for each group by considering the maximum team efficiency achieved by any activity in that group as being the maximum efficiency possible for the whole group.

For the third case, a combination of the two possibilities was considered. The support work for the main section slab pours were assumed to be of constant efficiency, the 300m^2 suggested by the contractor, while that for the concrete ramps, smaller slab pours in section E, and lift and services shafts were considered separately, giving four sections in all. The team efficiencies for the last three groups were the same as those obtained in the second case.

The optimisation of the model was carried out using the genetic algorithm. The number of evaluations of the objective function was limited to 10,000.

7.3 Results Of The Optimisations

The optimisation was performed on the three cases, both in terms of time and of cost. Further optimisation was done to observe how changing the constraints on the project would affect the optimisation.

7.3.1 Case 1

Four optimisations were performed initially, two of which sought to minimise the cost and two which sought to minimise the project execution time. The results are shown in Table 7.3, and include both the time and cost achieved at the optimal solution of each of the four optimisations.

		Initial	Resource only	Full
Minimising Cost	Cost	R14,189,692.29	R14,066,630.65	R14,055,765.95
	Duration	356	353	346
	Team Size	14	12	12
Minimising Time	Duration	356	347	334
	Cost	R14,189,692.29	R14,129,333.47	R14,156,565.52
	Team Size	14	16	16

Table 7.3. Results of the case 1 optimisations.

In contemporary practice, one common way of improving the efficiency of a project schedule, in terms of both cost and time, is to perform a line of balance analysis. This involves varying resource allocations (usually team sizes) to obtain the best combination of allocations. The first two optimisations in terms of time and cost (those marked *resource only* in the table) correspond approximately to such a technique.

It can be seen that once activity priority is added to the optimisation, its performance is enhanced, confirming the advantage of a global optimisation over a line of balance in this project. However, the differences are not similar for the optimisation of the two different project output criteria. The effect of adding activity priority is much more marked in the duration optimisation than the cost optimisation. The cost

reduction achieved by adding the priority is only R10,864, compared to the improvement over the initial value of R123,061 achieved by varying the team size alone, which represents only a 9% increase in the cost reduction. The reduction in duration achieved by adding the priority, on the other hand, was much more marked. The reduction of 13 days achieved is corresponds to a 144% increase on the initial reduction of 9 days achieved by the optimisation involving the resource only. This suggests that activity priority plays a greater role in the minimisation of the duration than the cost. Similarly the team size has a more significant role in the minimisation of cost than of duration.

A further useful observation is that the minimisation of cost also yields a reduction in the duration of the project from the initial value. In order to minimise the cost the resources must be used as efficiently as possible. This efficiency could have taken the form of reducing the team sizes and slowing the project down. However, in this case, it does not. The increased efficiency is obtained partly by a reduction in the team size, as can be seen, but it also corresponds to a decrease in duration. Compressing the schedule of activities which use those resources which must be supplied at their maximum rate is likely to lead to reduced costs because the total amount of time any resource will be on site is reduced. However, this effect is not marked enough that the duration corresponding to the optimal cost solution and the optimal time are the same value. The optimal time is considerably less than the optimal cost, and the cost of this solution is much greater than the minimum cost. This suggests that there is some trade-off between the time and the cost.

7.3.1.1 Changing The Constraints

So far two solutions have been presented for this project: a cost optimal solutions and a time optimal solution. But what if the project planner feels that the duration of the optimal cost solution is too long, while the cost of the optimal time solution is too short? Is it possible to make any compromise? It has already been suggested that there is a trade-off between time and cost within the optimal solutions. Is there any way this can be evaluated?

One common way of representing a project is in terms of cost-time curves. These show clearly the trade-off between cost and time. Selecting the best solution in terms of cost and time therefore becomes a matter of selecting the most appropriate point on the curve.

One means of achieving this with optimal solutions is to perform further optimisations while varying the constraints on the project applied by the project output criteria. Initially there were no constraints on the project output criteria, and the optimisations were free to progress to their optimal solutions without restricting themselves to a certain maximum cost or time. However, if the optimisation were to begin from the cost-optimal solution, and be optimised in terms of time, it would be possible to place a constraint on the cost of the project. Thus the optimisation is no longer trying to find the absolute minimum duration, but the minimum duration whose cost is less than a fixed value: the cost constraint. If this were performed with the constraint on cost being only slightly higher than the optimal solution, and then repeated, with the cost constraint being increased by a small amount each time, then it is possible to assess the trade-off between cost and time. This process could be continued until the optimisation succeeded in finding the minimum duration. Thus it would be possible to obtain cost-time curves of the optimal solution.

This process was attempted on the project data. Firstly, duration optimisations were performed, beginning from the cost optimal solution, with cost ceilings beginning from R14,060,000 and rising in increments of R10,000. Before this process was commenced, an unconstrained optimisation of time was performed from the cost-optimal solution. This was made to determine whether or not a better solution, in terms of either cost or time, might be achieved. However, while the duration of the solution was the same as the optimisation which started from the initial values, the cost of this second solution was R14,163,479, R6,914 more than the first. This showed that starting from an efficient cost-optimal solution was not necessarily an advantage in obtaining a very good time optimal solution.

There were two possible limits on the number of optimisations performed. The first was the cost ceiling being greater than the cost achieved by the unconstrained

optimisation. In this example there were two such unconstrained optimisations: the optimisation which started from the initial values and the optimisation which started from the cost-optimal solution. Therefore the lowest of these would be selected. This assumes that there is nothing to be gained from increasing the cost ceiling, as this will not constitute the addition of a better solution to the feasible region.

The second limit on the optimisations is when the minimum duration found by the unconstrained optimisation is obtained. Again it is assumed that progressing can yield no better solution because the optimisation should not be expected to find a better solution than the initial, unconstrained solution.

It was this second limit which was achieved first. This optimal duration, 334 days, was obtained when the cost ceiling reached R14,130,000. Further cost optimisations were not performed.

This process was repeated by minimising cost with a time constraint. Cost optimisations were performed from the duration optimal solution, with the constraint on the duration varying from 334 to 345. Optimisation stopped at 345, as the duration of the cost-optimal solution was 346, so no better solutions were to be expected by attempting further optimisations at higher values of duration constraint.

The results of both of these sets of optimisations are shown in Figure 7.2.

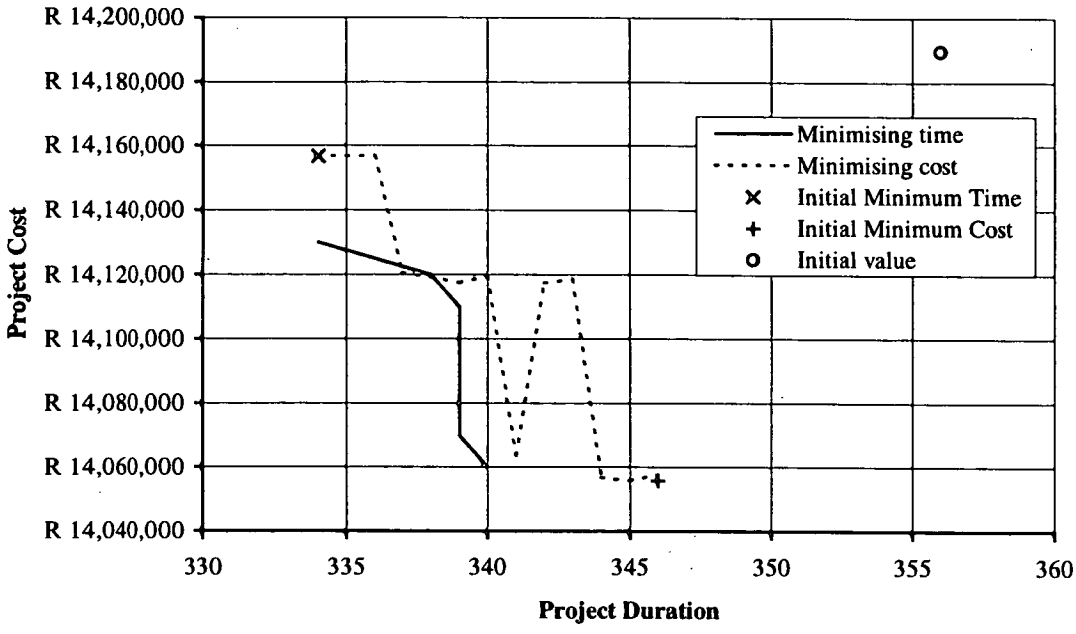


Figure 7.2. Cost time curves for the optimal solutions.

This figure shows how the optimal solutions found vary with the constraints applied. The minimising time optimisations tended yield better solutions than the minimising cost optimisations. It got down to a value of R14,060,000 at 340 days, whereas the cost optimisations were not able to achieve a solution of less than this value until 344 days. Additionally, the minimising time optimisations were actually able to improve on the cost of the minimum solution. However, the two sets of optimisations yielded much more similar results between 337 and 339.

There is, however, an anomaly in the graph, which is the cost of the minimising cost optimisation constrained at 341 days. The reduction in cost from 340 to 341 is very large. In addition to this the subsequent optimisations constrained at 342 and 343 days show very similar values to the optimisation constrained at 340 days. Both of these optimisations failed to find a solution which is known to exist within their feasible regions.

This can only be explained by considering the nature of the optimisations themselves. It was observed in the previous chapter that the objective of the optimisations could never be to obtain the true globally optimal solution, as this would require far more

computational resources than would be available. However, this does not mean that it is impossible to find the globally optimal solution, merely that it is highly improbable. Therefore it is likely that the optimisation constrained at 341 found either the optimal solution or another good solution which was very difficult to find. The other optimisations did not find it simply because the probability of their finding it was very low.

One of the factors which affected the ability of these optimisations to find good solutions was the fact that the feasibility rates of the optimisations were very low. At their most constrained (334) the number of feasible runs obtained by the cost optimisations was only 3 out of 10,000. At the constraint at 342 this value had risen to 1084, but it still only constituted just over 10% of the runs attempted. The fact that most of the runs were infeasible was caused by the fact that they had very small feasible regions. Subsequently these optimisations were not performing as efficiently as they might have been had they not been so strongly constrained.

This explains the higher performance of the time optimisations than the cost. The feasibility rates were found to be far higher for the time optimisations. This can be seen in Figure 7.3, which shows the number of feasible runs for each optimisation over the full set of constraint values. The optimisations started from their most constrained, and all optimisations were complete once the upper limit on the constraint was reached.

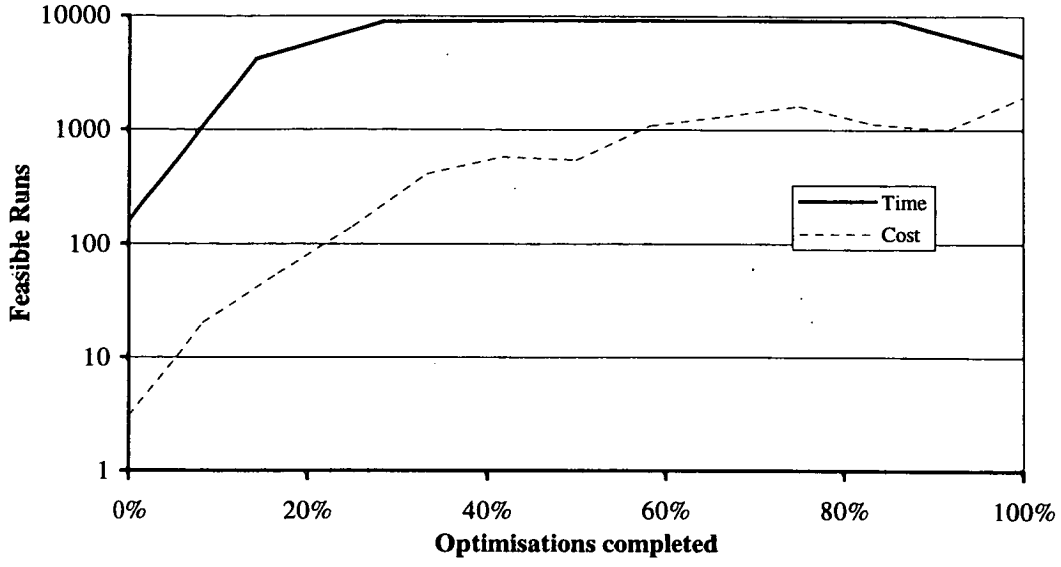


Figure 7.3. Feasibility rates of case 1 cost-time optimisations.

This figure shows that the minimising time optimisations tended to have several times as many feasible runs as the cost optimisation, with the difference being as many as 50 times for the more constrained optimisations. If the optimisation finds it difficult to even find feasible solutions then it suggests that it will also have difficulty finding good or near optimal solutions. This is borne out by the fact that the time optimisations, which had higher feasibility rates, tended to obtain better values.

One possible reason for the higher feasibility rates obtained with the time optimisation is that there was much more scope within the project for varying time than cost. Therefore the GA would be more likely to generate solutions with significant variation in duration, than significant variation in cost. This can be demonstrated by considering the actual values of the optimal solutions obtained by the two sets of optimisations. In the cost-time curves presented the value of the constraint on the output criterion was used rather than the actual value of the final solution. In reality the value of the constrained criterion tended to be less than the constraint. Therefore if the actual values of the criteria are considered, a scatter plot of the solutions may be obtained. These values are shown in Figure 7.4.

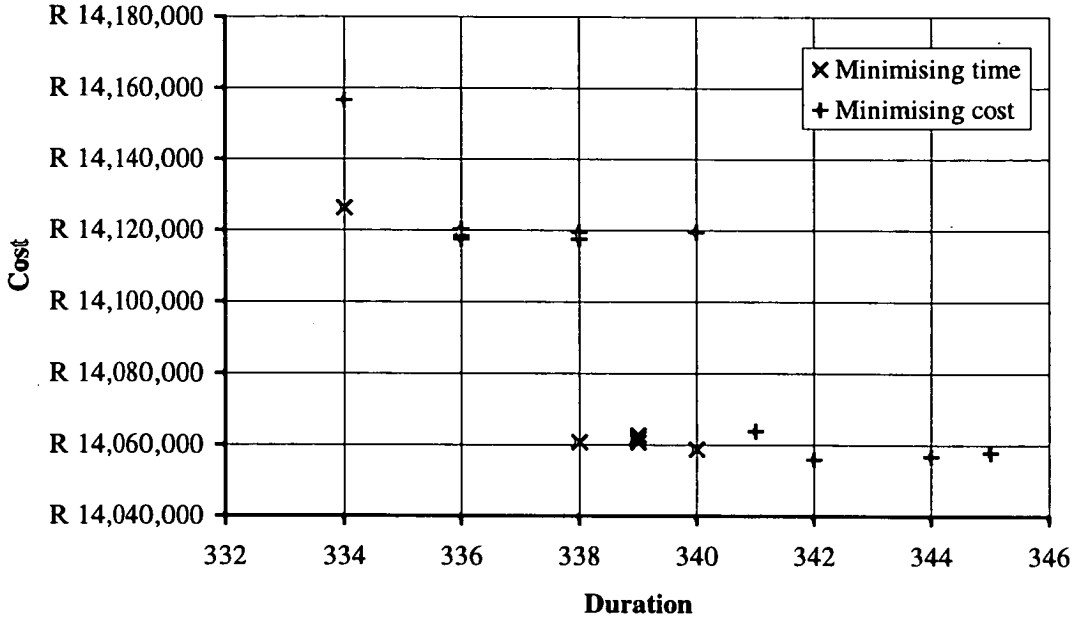


Figure 7.4. Actual results of cost/time optimisations.

There are two clear groups of results in this graph. Each of these groups display significant variation in duration, without much variation in cost. In the first group the cost is scattered at around R14,040,000, with a lot of variation in duration, and the second one is scattered around R14,120,000 and also displays some variation in duration, although not as much as the first group. The fact that there is so little change in cost within these groups in comparison to the duration suggests that it is easier to change duration without changing the cost significantly than changing cost without affecting time. Therefore if an optimisation is constrained in terms of time then it is more limited than if it were constrained by cost. It is this fact which leads to the higher feasibility rates among the time optimisations. The existence of these groups could also explain the improved performance of the time optimisations. Any optimisation would tend to be restricted by its constraints to one of the two groups. This is because any change in the team size would probably cause a significant change in time as well as cost, because the highly efficient schedules in the population rely on a set of activity durations, and changing these durations could significantly disrupt this schedule.

In the optimisations of cost, there was no means of making the optimisations find lower values of duration within the group in which it was constrained. The duration optimisation, on the other hand, would necessarily seek out better solutions within that group. While initially this may have been difficult due to a very low cost constraint, once the cost constraint had risen a little it would be comparatively easy..

The existence of these groups, and their significance for the optimisations can be explained further by considering Figure 7.5, which groups these bands by the support work team size.

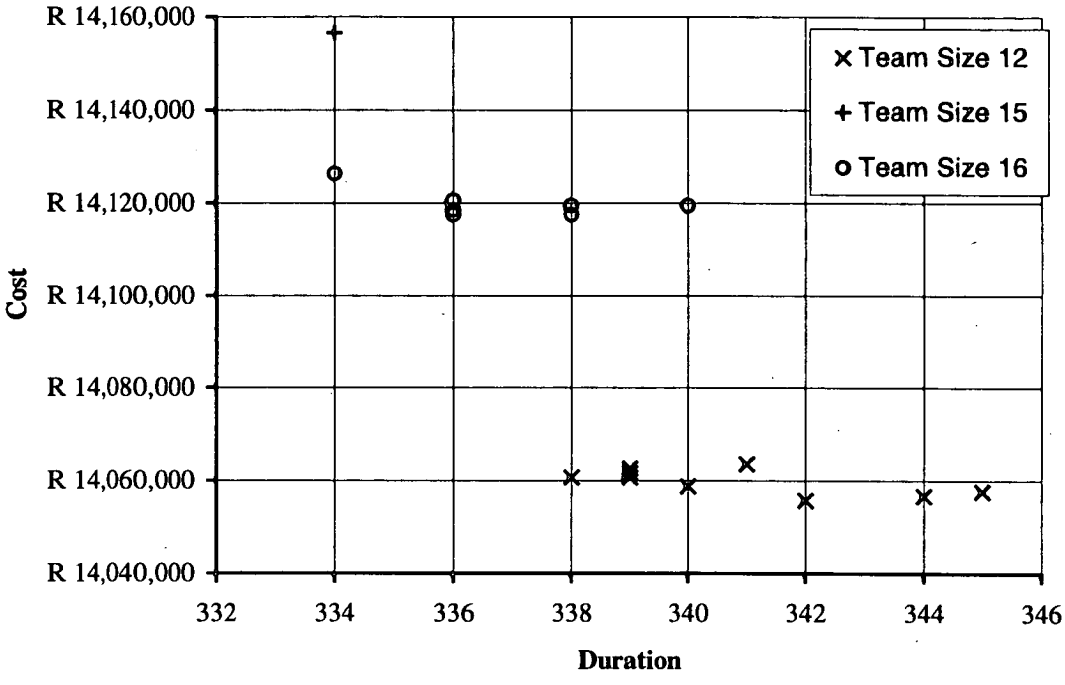


Figure 7.5. Actual results of cost/time optimisations grouped by support work team size.

This Figure shows how the groups correspond to two different team sizes for which efficient solutions could be found: 12 and 15. It also identifies a third group for which only one solution was found: team size 16.

This shows how significant the selection of the support work team size is in determining the cost of the optimal solution. While the duration of the project may change, the cost can remain approximately the same. In addition to this, it can be

seen how shorter durations are harder to find, if they exist at all, for the smaller team, size 12, than for size 15.

7.3.2 Case 2

In case 2 the assumptions for the working rates of the support work teams were highly conservative. The initial schedule under these assumptions was 469 days, compared to 356 days for case 1. The cost of the project was also much higher: R15,020,062 – nearly R1,000,000 more than case 1. These high values were both caused by the fact that many of support work activities will have a greatly increased manpower requirement over those in case 1.

As with case 1, the optimisation was attempted with first only the team size as variable, and then both team size and priority. The results of these optimisations are presented in Table 7.4.

		Initial	Resource Only	Full
Minimising Cost	Cost	R 15,020,062	R 15,020,062	R 14,966,831
	Duration	469	469	417
	Team Size	14	14	14
Minimising Time	Duration	469	469	394
	Cost	R 15,020,062	R 15,020,062	R 15,136,686
	Team Size	14	14	16

Table 7.4. Results of case 2 optimisations.

Unlike case 1, no improvement on the initial schedule can be obtained by only allowing the team size to vary the optimisation. For the cost optimisation, this can be explained by considering what will happen if the team size is either increased or decreased. For each of the groups of support work activity, most of the activities entailed placing a very similar quantity of support work. The ceiling is fixed as being the highest of these values. Therefore the activities, according to their applied resource ceilings, are already all operating at near maximum efficiency. Changing the team size is highly unlikely to produce any greater efficiency than this.

The failure of the time optimisation using team size only to improve upon the initial value can be explained by the fact that it may not be possible to perform any reductions in duration of activities. The highest possible increase in team size from

the initial value was by two men, giving a team size of 16. This is a 14% increase in manpower. The highest duration of any support work activity was 6 days. In order to achieve a reduction in duration from 6 days to 5, a 20% increase in manpower would be required, assuming that the resource was at its maximum. While the resource was not always at maximum, it was usually very close, and this fact would prevent any reduction in activity duration in most cases. If any reductions in duration were achieved in the optimisation using team size only, they were never in activities which were contributing to the delays to the schedule, hence no improvement in the project duration could be achieved by varying the team size alone.

The optimisations that included priority did achieve substantial reductions in time. Although the resulting duration was still much more than even the initial duration for case 1, the minimising time optimisation achieved a reduction in duration of 75 days. This showed that insufficient consideration had been made in the preparation of the schedule of the possible resource conflicts which might exist within the project. Only by performing an optimisation which considers the prioritising of conflicting activities can a good schedule be obtained.

The cost optimisation, on the other hand, did not achieve such a large reduction. The cost reduction was only R53,231, less than half that achieved in case 1. However, this low level of reduction was to be expected, there was no benefit to be derived from changing the team size, a fact which can be shown by considering the fact that the optimal team size was the same as the initial team size. In the case 1 optimisations most of the cost reduction came from varying the team size, and the addition of priority to the optimisation only obtained a further R11,000 saving, much less than in case 2. It is clear that the initial schedule for case 1 is more efficient. This can be confirmed by considering the fact that the reduction in duration by the time optimisation in case 1 only yields a 22 day improvement, and the cost optimal solution is only 9 days shorter than the initial, whereas the difference in the case 2 optimisations is much greater.

The fact that the teams are at near maximum efficiency also explains why the cost of the solution obtained by the duration optimisation was so much higher than the initial

cost: in order to achieve this duration, a higher and more costly team size was employed. However, as it has already been argued that variation in team sizes will cause at most very few reductions in duration, the fact that the duration optimisation chose such a value might seem surprising. Indeed, as the duration optimisation does not consider cost, it is possible that this high team size, and hence high cost, appears only by chance.

In order to address this is it necessary to consider the cost-time curves for the optimisation of the project. These are shown in Figure 7.6.

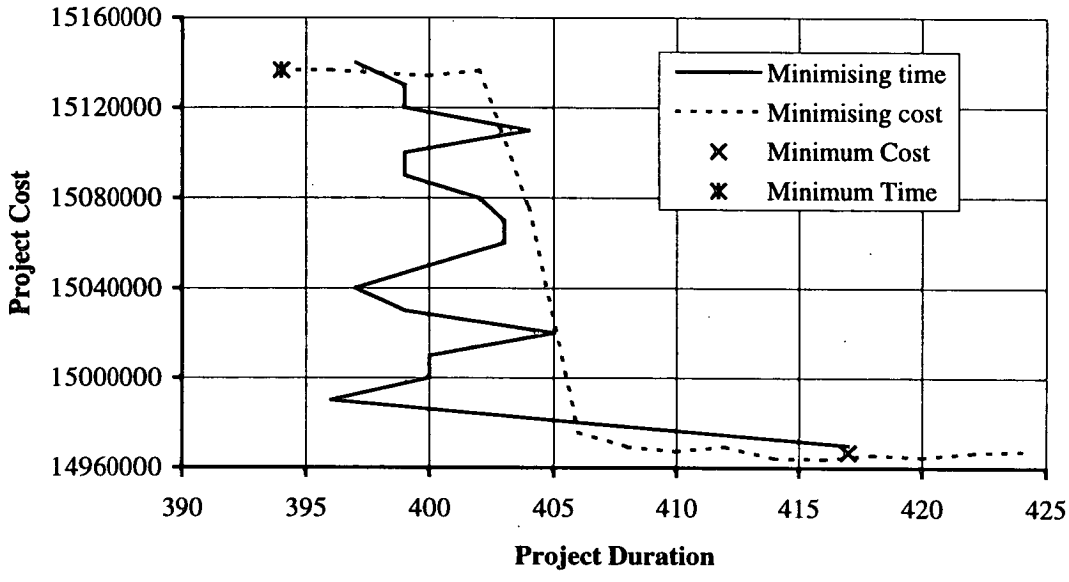


Figure 7.6. Cost-time curves for optimal solutions, case 2.

The cost-time curves show a similar pattern to those in case 1. The minimising time optimisation performs better than the minimising cost optimisation. It should be pointed out that while the performance of the two optimisations between 406 and 417 appear to be very similar, this part of the graph only contains one data point for minimising time, at 417. Again it can be seen how the minimising cost optimisation has difficulty finding low values of cost because of the constraints on time.

The fact that the minimising time optimisations achieve a solution which is very near the time optimal solution at a much lower cost does suggest that the minimising time

optimisation might not need to increase the team size in order to achieve its lowest solution. This may be confirmed by considering the actual results by team size. These are shown in Figure 7.7.

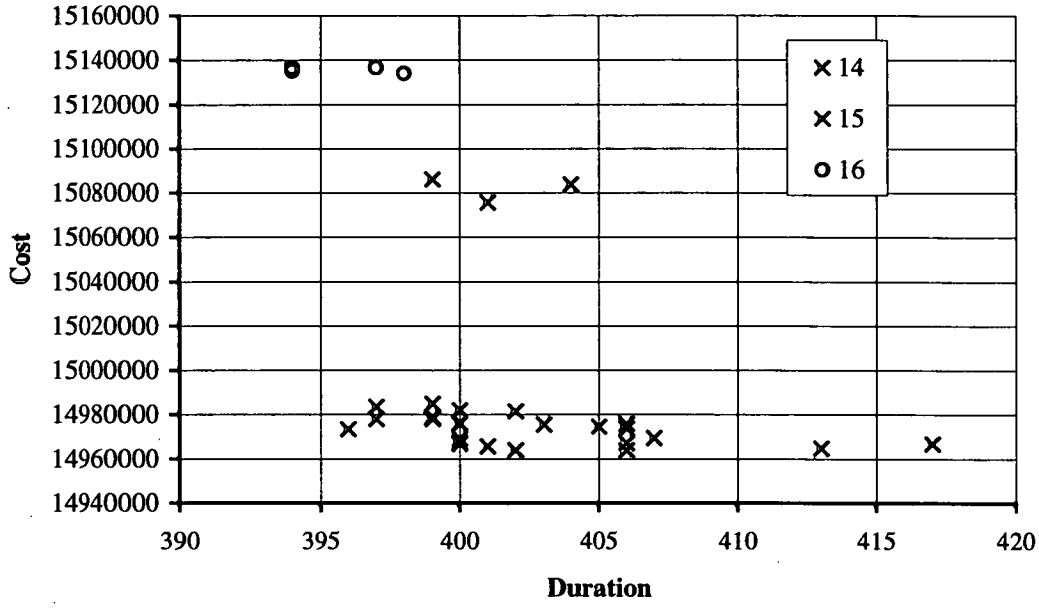


Figure 7.7. Solutions from cost-time optimisations case 2.

This plot shows how the cost of the solutions vary with different team sizes. The shortest solution obtained with the team size at 14 is only two days longer than that obtained by team size 16. This suggests that while the team size does not affect the duration of the project significantly, it is possible to make a very small time saving by increasing the team size. However, there is a significant cost penalty involved in this.

7.3.3 Case 3

Case 3 is essentially a compromise between the liberal assumptions made in case 1, and the highly conservative assumptions from case 2. As such, however, it is probably the most accurate of the three cases. The initial duration and cost of this

project was, as might be expected, between the values from case 1 and hose from case 2. These, as well as the results of the optimisations, may be found in Table 7.5.

		Initial	Resource Only	Full
Minimising Cost	Cost	R 14,796,973	R 14,770,129	R 14,734,572
	Duration	429	391	355
	Team Size	14	16	16
Minimising Time	Duration	429	391	347
	Cost	R 14,796,973	R 14,770,129	R 14,742,882
	Team Size	14	16	16

Table 7.5. Results of case 3 optimisations.

As in case 1, the optimisation which corresponds to a line of balance optimisation does obtain a better solution than the initial schedule. However, better solutions can be obtained by considering priority as well. Indeed, the results of this optimisation are similar to the previous cases. There appears to be more scope for optimisation of time than optimisation of cost. As with case 1, the efficient solutions found by the optimisation which minimises time also have a cost which is better than the initial value.

However, there is one important difference between the case 1 team sizes and the team sizes in this case. In case 1, the optimal cost was obtained by reducing the team size. In this case, all the optimal solutions found had a team size of 16, which means that the optimisation of cost actually required an increase in the team size, the opposite of case 1. It is often assumed that to increase the allocation of resources, sometimes referred to as crashing activities, will necessarily involve a cost increase. However, this is not the case for this particular project. These results suggest that, as was argued by Yau et al.[Yau, 1990], increasing the resource allocation to an activity will not necessarily lead to an increase in cost. It is possible that in certain situations such a strategy can actually lead to a reduction in cost.

The fact that the team sizes are the same for both time and cost optimal solutions means that the cost time curves may not necessarily have a similar structure to those of cases 1 and 2. The structure of both of these curves was influenced by the fact that different groups of results were to be expected at different values of team size. However, all the solutions in the cost time curves for this solution have the same value of team size: 16. The curves are shown in Figure 7.8.

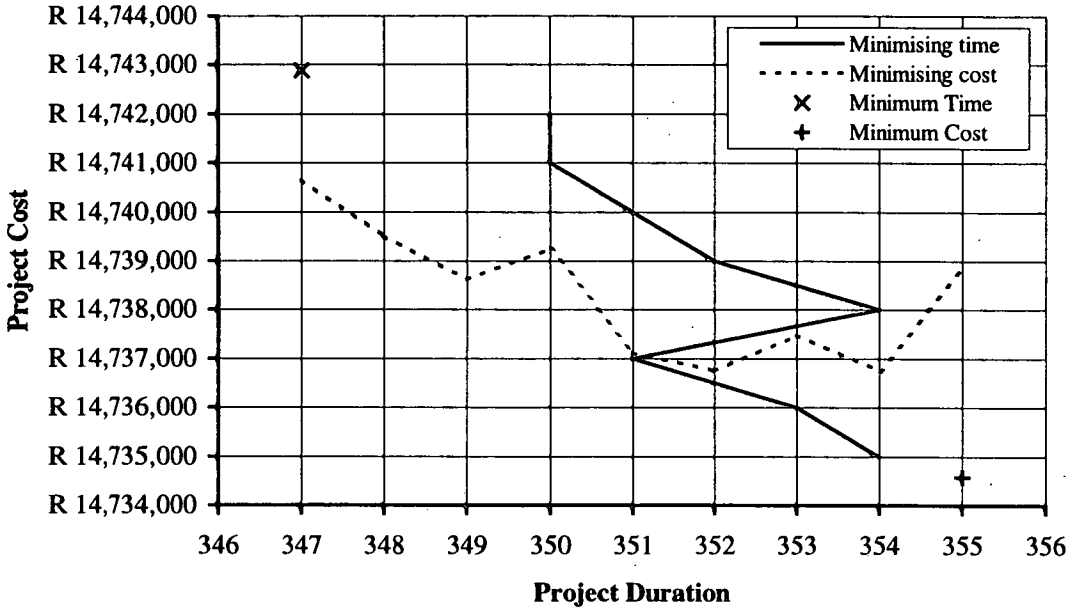


Figure 7.8.Cost-time curves for case 3

In the previous 2 cases, the minimising time optimisation obtained the best solutions. Initially the minimising time optimisation does manage this for this case also. The optimisation starts from the unconstrained cost optimal solution, which has a duration of 355 days. This starting solution has a much lower cost than that of the minimising cost optimisation with its constraint at 355 days. Therefore the small decreases in duration obtained in the first few optimisations yield solutions with a much lower cost than solutions obtained by the minimising cost optimisations at similar values of duration. However, as the cost ceiling is raised, the optimisation is simply unable to find better solutions than 350 days. This is 3 days less than the duration of 347 found by the unconstrained time optimisation.

The behaviour of the minimising cost optimisation is quite similar. It starts from a solution whose duration is much less than that which could be found by the minimising time optimisations for a similar cost. Some initial reduction in cost was obtained as the ceiling on the duration was increased, but little improvement was obtained as the constraint on duration was increased beyond 351 days.

This differs from the first two cases. In these the optimising cost optimisation would, at least, obtain a value very close to the unconstrained optimal cost. Similarly the minimising time optimisation would approach the unconstrained minimum time. However, this is not achieved by either of the two cost-time curves in this case. It would appear that finding such a solution is too difficult.

In previous cases, one reason for an optimisation failing to perform is that the constraints cause a low feasibility rate, which reduces the number of runs the optimisation can use over its course. Therefore looking at the feasibility rates for the cost time optimisations might provide some insight into the reason for their poor performance in comparison to those of cases 1 and 2. The number of feasible runs obtained is shown against the number of optimisations completed, as a percentage of the whole, in Figure 7.9.

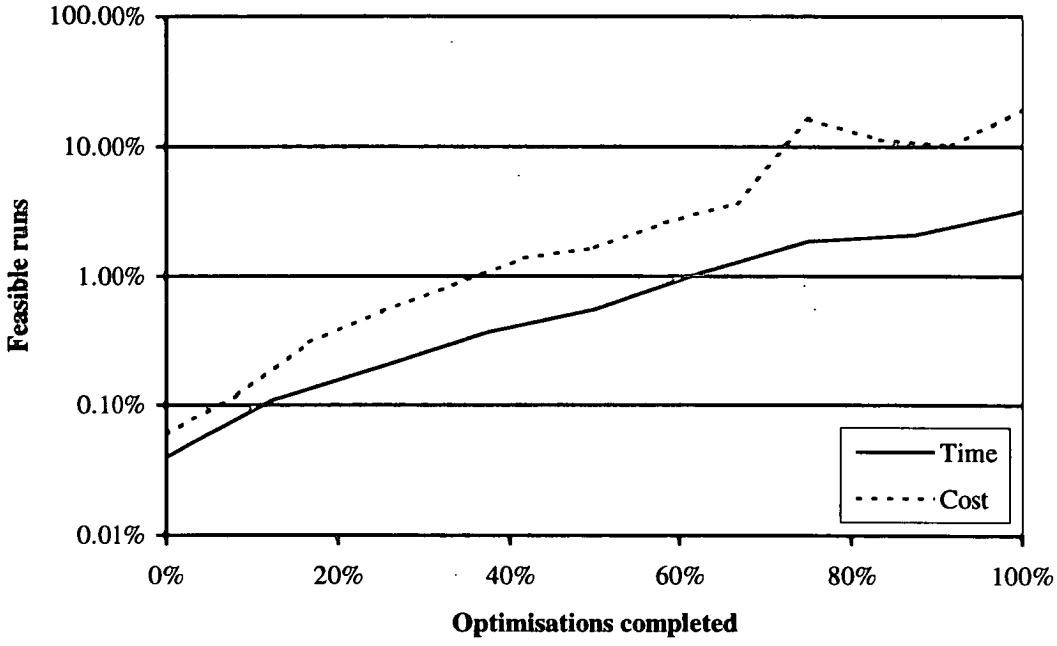


Figure 7.9. Feasibility for case 3 cost-time curves.

As with case 1, the feasibility rates are very low when the optimisations are at their most constrained. Unlike case 1, however, the time optimisation actually has lower feasibility rates than the cost optimisation. Additionally, the feasibility rates do not approach 100% towards the end of the optimisations. The cost optimisation obtains a 20% feasibility rate at its least constrained, and only 3% of the time optimisations are feasible at their least constrained. Higher performance of the optimisations is usually associated with higher feasibility rates. Therefore, one reason that the cost time curves do not approach their unconstrained optima is because the optimisation finds it difficult to obtain feasible runs.

This could pose a problem for obtaining cost time curves within a full global optimisation of a project. The project under study here is not too highly constrained, as only three resources are constrained in any way. However, where more constraints exist it must be expected that the optimisations will not perform as well. In order to solve this problem it will be necessary either to invest more computational resources to the optimisation, or to use a more efficient optimising algorithm.

The genetic algorithm being used for this project was quite a straightforward one. However, there is much academic research being devoted to improving the performance of GAs. In the future, it will be possible to use GAs which are much more efficient than the simple one which was used for this research. While these techniques may involve a greater overhead (i.e. the amount of time spent processing the GA as a proportion of the time spent running the model will increase), the increase in performance should outweigh this.

7.3.4 Resource Use

In section 7.2.3.1 it was asserted that it would be necessary to monitor the optimal solutions provided by the model in order to assess how close to their maximum efficiency the support work teams were operating. Although the maximum efficiency was known, it was not known for how long this could be sustained by the team without loss of efficiency. The absence of any technique which might permit any drop-off in efficiency to be modelled meant that it was necessary to check that the optimisations were not causing the teams to work at or very near their capacity for extended periods.

The results of the optimisations were analysed to see when in the project the teams were above certain levels of efficiency, which were expressed as a percentage of their absolute maximum efficiency. From this it was possible to determine the maximum number of days for which the team was required to work above certain level of efficiency. This is shown in Figure 7.10 for a number of levels of efficiency, varying from 95% to 70%.

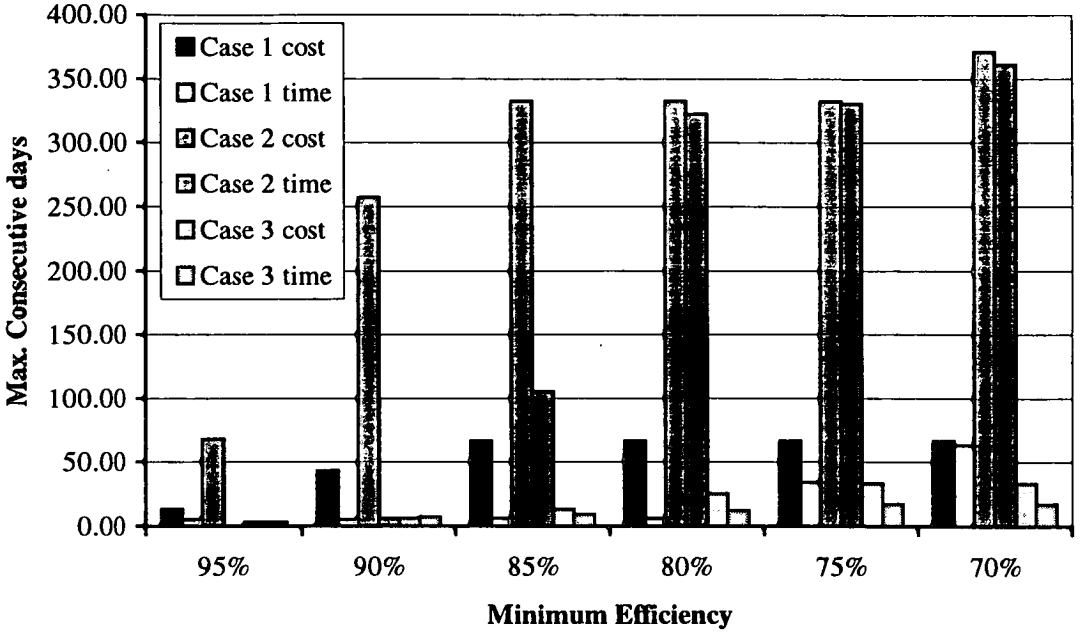


Figure 7.10. Maximum consecutive days at high work rate.

This figure shows how both case 1 time and case 3 only require higher levels of efficiency (above 85%) for short periods of time. Lower levels of efficiency are required for longer, but these are much easier to sustain. This suggests that, for these case 3, the optimal schedules of work identified for the support work teams are realistic ones.

However, the case 1 cost optimisation does require a team to operate at over 90% efficiency for 43 consecutive days, and above 85% for 66 consecutive days. The reason for this value being so much higher than the case 1 time optimisation, is that the use of the lower team size (12) in the optimal solution results in a lower consumption of resources for the same amount of work. This is how the reduction in cost is achieved by the optimisation. Whether this level of work is sufficiently high to result in a loss of efficiency is not known. Nevertheless, this does demonstrate that there is potential for the optimisation of cost to result in high levels of efficiency being required in order to execute the optimal schedule.

While the case 1 cost optimisation may be required to operate at high levels of efficiency for too long, the case 3 optimisation requires the teams to work at

maximum efficiency for much longer. The teams in the cost solution are required to operate at above 95% for 68 consecutive days. This is almost certainly too long. This phenomenon is caused by the fact that all the activities are very close to their ceilings from the outset, as was discussed in section 7.3.2. Thus, when the team size is increased, as it is for the case 2 time, the level of efficiency required is diminished. This is caused by an increase in manpower consumption, and causes the number of consecutive days at above 90% to diminish considerably. It might be argued that sustained periods of activity of under 90% are realistic, thereby validating the case 2 time optimisation. However, even if this were so, it would still mean that the only optimisations in the cost-time curves for case 2 which might be considered valid are those whose team size is 16.

The maximum number of consecutive days for which a team is required to operate at above a certain level of efficiency provides a very useful indication of how long the teams are required to work close to their maximum efficiency. However, they provide no indication of how often they are required to do so. This can be addressed by considering two other measures: the total number of days for which the team is required to operate above a certain level of efficiency, and the average number of consecutive days for each time the efficiency rises above that certain level. These are shown in Figure 7.11 and Figure 7.12.

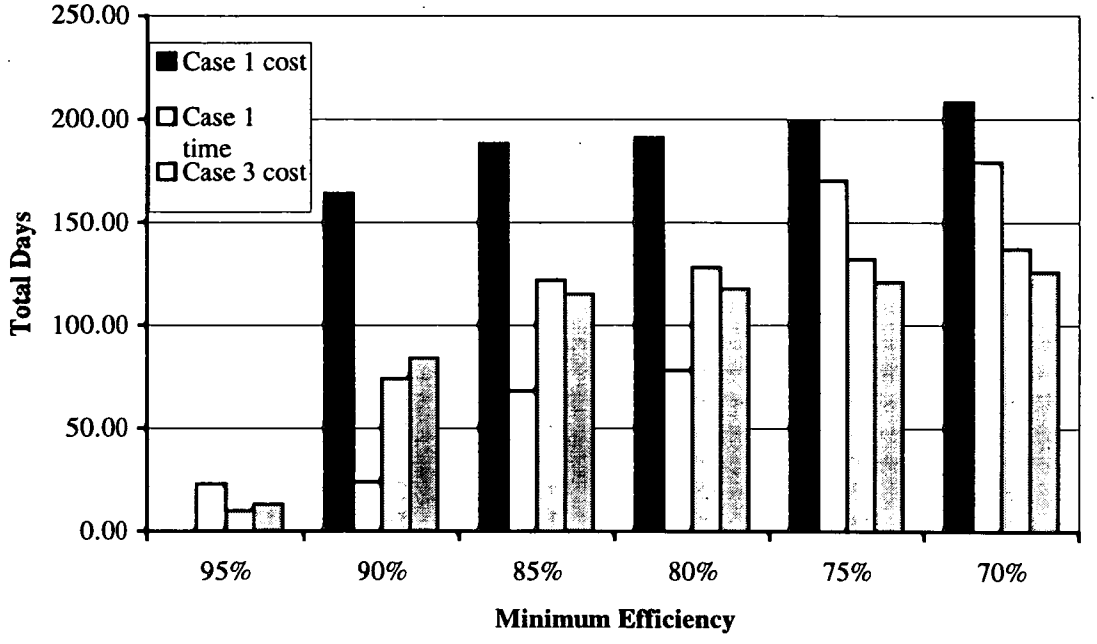


Figure 7.11. Total days at high work rate.

The total number of days over 85% is quite low for case 1 time, but constitutes almost a third of the total project duration for the case 3 optimisations, and even longer for case 1 cost. The high values for case 3 can be explained by considering the fact that there were some types of support work activity in case 3 which relied upon the same conservative assumptions as those in case 2. The activities for which this was the case can be expected to have the same effect on the case 3 optimisation as on the case 2. Therefore it is possible to assert that had all the resource levels been accurate, rather than having been conservatively estimated, the total number of days over 85% efficiency might have been less. However, it may also be possible to determine that the levels of efficiency are not required to be sustained for extended periods of time. This can be seen not only in Figure 7.10, but also in Figure 7.12, which shows the average number of consecutive days was above a certain level of efficiency.

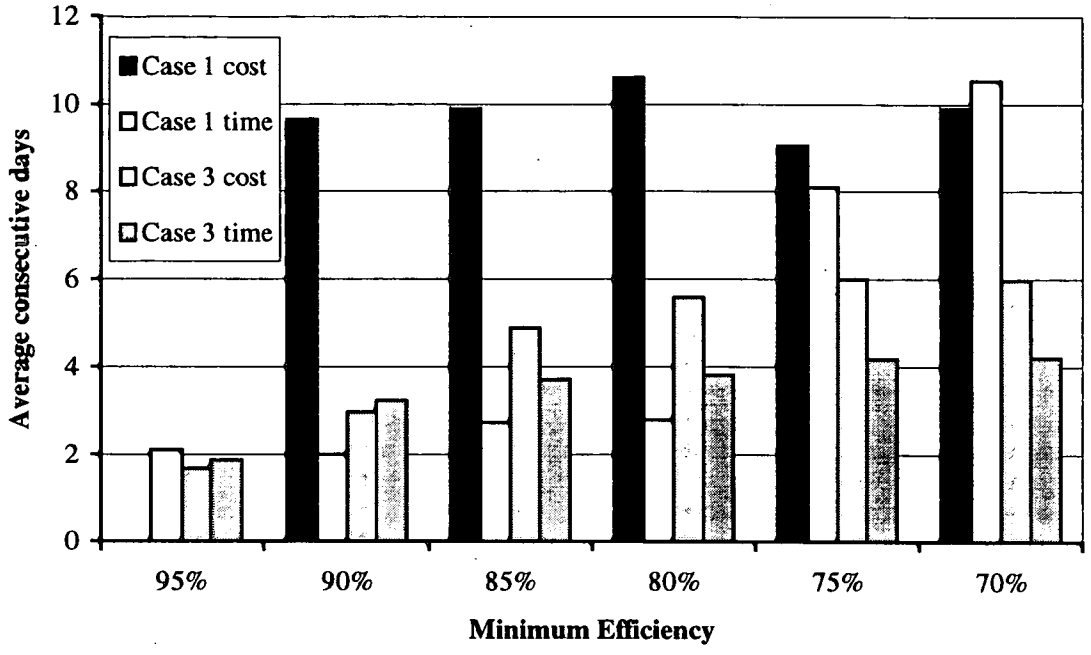


Figure 7.12. Average consecutive days at high work rate.

This figure shows that the average number of consecutive days at an efficiency of greater than 85% of the maximum is no greater than 5 for case 3 optimisations. Thus it can be concluded that the teams the teams are not required to work at high levels of efficiency for extended periods.

However, this graph also shows that the case 1 cost solution requires the teams to operate at above 90% efficiency for, on average, just less than 10 days. This could be a long time for them to consistently perform at such levels.

This analysis has served as a useful method of evaluating whether or not the optimisations have been generating schedules with feasible levels of support work team efficiency. However, it cannot be recommended for use with the global approach, should the question of efficiency over sustained periods of activity be raised for a project. The analysis of these projects has shown how the optimisation can tend to generate solutions which are unfeasible by this analysis. This is not acceptable, as there is not point in generating solutions which are unfeasible. It would therefore be necessary to incorporate considerations of efficiency into the

model itself. This could be done through project output criteria relating to efficiency, with some kind of limits set on them. However, it would be much better to incorporate them into the model directly. This would allow the effects of any drops in efficiency on any of the project output criteria to be modelled explicitly.

7.3.5 Times Of Optimisations

In the previous chapter it was argued that the relationship between the size of the project and the amount of time required to run the project model was a linear one. This was significant as it would be unlikely that an optimisation could be performed on a real project if the relationship were to be exponential. Based on this assertion a linear regression model was created to express cost in tems of the project size and duration. This linear regression model can be used to make a prediction of the run times for the project under analysis in this chapter. These predictions are shown for all 3 cases, along with the actual values of run time obtained for both the cost and time unconstrained optimisations, in Table 7.6.

	Case 1	Case 2	Case 3
Activities	1252	1252	1252
Initial duration	356	469	429
Predicted run time	17.0ms	17.4ms	17.3ms
Actual run time (cost)	16.4ms	19.7ms	16.4ms
Actual run time (time)	16.2ms	19.1ms	17.8ms

Table 7.6. Project size and run times against predicted.

Predicting the run times for the projects using the regression model form the previous chapter involves extrapolating to a project nearly 25 times the largest project used to create the model. Nevertheless, the values are very close. The highest deviation from the predicted value is only 13%, by the cost optimisation in case 2. If the relationship between the size of the project and the run time were exponential then the actual run times would have been much greater than those predicted. Therefore the relationship must be approximately linear.

This is very important, as it shows that run times for the very largest projects (around 10,000activities) are likely to be somewhere in the region of 200ms, which makes a 10,000 run optimisation a realistic proposition.

7.4 Conclusions

A model of part of the construction phase of the new car park at the Johannesburg International Airport has been obtained and optimised. The full set of data for the resources and their consumption were not available. However, sufficient detail was available for the concrete pour and support work activities to model the associated resources quite effectively.

The availability of cost data permitted a second project output criterion to be added to the criterion of total duration identified in the previous chapter. Therefore optimisation could be performed by minimising either time or cost.

7.4.1 Preparation of the model

In the preparation of the model for optimisation, a number of problems with the model were encountered, and had to be rectified:

- A number of delays to the project were encountered when the scheduling algorithm was used rather than a straightforward network calculation. This was because the project planner had chosen a set of relationships which would create the correct delays in activities to ensure that a correct sequence of events, but which did not reflect the precedence constraints which existed in reality.

- The resources for the concrete pour and support work were simply provided as one amalgamated resource which would include all manpower and material resources and costs. The manpower was assigned and cost using information which had been obtained by Harry Fleetwood-Bird. The actual cost of the materials was then calculated by subtracting the average cost of the manpower per unit from the unit cost of the resource as a whole.

- Accurate modelling was difficult to achieve because the team sizes were set by the contractor, and the durations by the scheduler. In the end this lead to three different possible representations of the project being considered, each of which would be optimised as if it were a different project.

These observations, as well as the fact that so little of the project had been modelled, suggested that insufficient modelling had been performed as part of the project to allow a global optimisation to be performed. While to use the results of this single case study to generalise about the amount of careful and accurate modelling currently used by project planners in the industry would be presumptuous, the fact that very little modelling has actually been carried out does seem to be consistent with other's criticism of project modelling within the construction industry. Therefore, if global project optimisation is to implemented it seems likely that improvements to the modelling associated with project planning must be improved in tandem.

Once the data for the project had been corrected, it was then necessary to identify additional analytical techniques to be employed

- The costing of the model was to be implemented. Two types of costing of resources were identified: costing the resource as it was used and costing the resource at ceiling. Costing the resource as it was used involved simply multiplying the cost per unit by the number of units used. Costing at ceiling involved multiplying the unit rate by its ceiling. This allowed the cost of a resource which had to be available at a fixed rate, irrespective of how much it was used, to be represented.

- Two resources were identified which were required to be allocated to an activity at a fixed rate irrespective of the duration of the activity or other resource allocations. Therefore a facility to permit such a resource type was implemented.

- A facility to divide resources into teams was required. The team size would be the resource allocation rate for all activities which used that resource.

- Although a maximum team size was available for the support work teams, the teams would not be able to operate at that level for extended periods. However, in the absence of any means of addressing this using analytical techniques within the model, it was necessary to check whether the optimal solutions were realistic in their requirements of the support work teams.

While these analytical techniques were added to the model, it was recognised that, with sufficient input from the contractor, it might be possible to develop and implement more analytical techniques. These would have been required for a truly accurate model of project costs. Unfortunately this was not possible.

Nevertheless, it did show how a true global project model may take into account much more data than is usually included in a resource constrained project model with costs. This would constitute other factors and relationships that affect the project which are not modelled within the traditional RCPS paradigm. Such factors might be identified by practitioners, or gleaned from the literature. However, in any event, it would be necessary to determine how these factors might be implemented in such a way as to constitute accurate modelling of the project. This would require identification of such factors, and detailed discussion of to what extent and in what way they would influence different aspects of the project. In order to be able to do this an intimate knowledge of the project would be required of the kind that only the project team would have.

After the analytical techniques had been identified it was then necessary to identify the project input variables. Only two were available: activity priority and the support work team size. The labourers on the concrete pour were also considered. However, that team size had to remain fixed for technological reasons.

7.4.2 Optimisation

Three different cases were optimised. These corresponded to the three different representations of support work rates identified. The results of the initial cost and time optimisations are shown in Table 7.7.

		Case 1	Case 2	Case 3
Minimising Cost	LOB	0.87%	0.00%	0.18%
	Full	0.94%	0.35%	0.42%
Minimising Time	LOB	2.5%	0.0%	8.9%
	Full	6.2%	16.0%	19.1%

Table 7.7. Optimal solutions for all three cases.

The fully blown global optimisation technique was compared to what was effectively a line-of-balance optimisation, where only changing the resource team size was considered. In each case the full optimisation achieved better results both in terms of time and cost, demonstrating the superiority of the global approach for this project.

Observations were also made of the duration of the cost optimal solutions and the cost of the time optimal solutions. It was found that the duration of the cost optimal solutions was less than the initial duration. Similarly, a reduction in cost was also observed for cases 1 and 3 in the duration optimisation. It was explained how this was due to the fact that solutions which were efficient in terms of time would also tend to be efficient in terms of cost. In case 3 it was even found that the support work team size was the same for both the cost and time optimisations. Nevertheless, it was also identified that there would be a trade-off between time and cost between the two optimal solutions found.

In order to assess this trade-off, optimal cost-time curves were made using repeated optimisations with varying constraints on whichever project output criterion was not being optimised. This allowed cost-time curves to be obtained for both cost and time optimisations. While smooth curves might have been expected, the results showed quite an irregular progression from one optimum to the other. While the general progression was for the cost to decrease as the durations increased, there were a number of cases where an increase in cost was observed along with an increase in duration. These results were indicative of the fact that the optimisations were not finding the true global optimum, but near optimal solutions.

Nevertheless, it was possible to observe the general trends of the optimisations. Initially, while the projects were very highly constrained, feasibility rates were very low, and the improvements made by both optimisations would be small, although the time optimisation would achieve more reduction at this stage. In cases 1 and 2 it was found that time optimisations performed better throughout the cost-time curves, because it was easier to achieve duration reductions without significant reductions in cost than vice-versa. This was not found to be the case for case 3, where the time

optimisation performed well initially, but then failed to yield any further improvement.

In this case it was found that that the cost-time curves did not approach the opposite (initial) optimal solutions from those they had started at. In both case 1 and case 2 the optimisations of cost and time in the determinations of the cost-time curves had approached the unconstrained cost and time solutions respectively. However, both the curves in the case 3 optimisation failed to do so. This suggests that, for some projects, applying constraints in order to find solutions between the optima of two project output criteria might not yield suitable solutions.

The results of the cost-time curves also showed that there was a significant cost difference between different team sizes. Any optimal solutions whose team size was a certain value would always have a similar cost to any others with that same team size. Conversely, if the team sizes were different then the difference in cost would be much greater. This fact might have had a significant detrimental impact on the ability of the optimisations to find good solutions.

7.4.2.1 Resource use

The results of the time and cost optimisations of all three cases were analysed to ensure that the teams were not expected to work at high levels of efficiency for too long. This was performed by considering three variables: the maximum number of consecutive days for which a team was expected to operate at above a certain level of efficiency, the mean number of consecutive days for each time that level of efficiency was exceeded, and the total number of days for which that level was exceeded. These showed that the time optimisation of case 1 and both the optimisations of case 3 were obtaining solutions which were realistic in terms of assumed team efficiencies, while the others were not. It was concluded that it would be much better to model the effects of such situations, should they arise, as part of the model, rather than after the optimisations had been completed, as it would allow the effects to be modelled explicitly.

7.4.2.2 Time

The run times for the optimisation were assessed for all three cases, and compared to the values predicted by the regression model obtained from the previous chapter. Most of the values were quite close to the predicted values, the largest deviation being 13%. This showed that the relationship between project size and execution time is approximately linear. Therefore, the run-times required for the optimisation of very large project models should not be so large as to make optimisation too computationally expensive.

Chapter 8 Conclusions And Suggestions For Further Work

The aims of this thesis have been to put forward a new approach to the optimisation of projects – one which considers the whole project within one single optimisation. The need for such a comprehensive technique was identified from the literature, and its fundamental requirements identified. A model was developed for some simple test projects before being expanded to include part of the construction phase of a real project.

The need for a global approach to project optimisation was identified from the literature. This began with the development of analytical project modelling techniques, building upon the early PERT/CPM model, including techniques which provided optimal or near optimal solutions to specific project problems. Subsequently the impact of the qualitative aspects of project management – particularly systems theory – were investigated. These demonstrated the need for a holistic approach to the modelling of projects. Techniques which optimise small subsystems of a project can have a significantly detrimental impact on other subsystems within the project. In addition, the optimisation of these subsystems fails to capitalise on the full scope for optimisation available within the project as a whole. Therefore a global project model, and subsequent global optimisation of that model, is required. Such a model would, in its final form, be required to encompass the entire project life cycle, and permit all variables within the project to be considered within one single optimisation.

Having identified this need, the requirements of such a model were defined, and a general model, or framework, was developed. It included only those factors which were common to all stages of all projects. This framework, which incorporated a simple resource constrained scheduling algorithm, was required to determine when each activity involved in the project would take place, and evaluate the resources used in its execution. Then, using this simple framework as a core, a set of analytical techniques could be incorporated into the model to create a specific model of a

particular project. These analytical techniques would be required to manipulate the project input data – the actual hard data which will be used to model the project – and generate values for project output criteria – the measures of project success.

In order to ensure that an accurate model would be produced from the general model, a number of logical steps were defined which, if followed, would permit the formulation of an accurate global project model for optimisation. Such an optimisation would require the definition of all variables available to the project, which would permit full advantage to be taken of the full scope for optimisation within the project.

8.1 Application Of The Model To 110 RCPS Problems

First, the model was applied to a simple set of 110 projects which had been used in the literature to test resource constrained scheduling algorithms. The global approach was applied, and a model formulated. Three different types of project input variables were identified: activity priority, MEST and PRU.

A computer program which allowed this model and optimisation to be performed was created. This model included the scheduling algorithm and the analytical techniques required to determine the activity durations. Furthermore a genetic algorithm was created in order to perform the optimisation. The completed program was developed as a fully blown 32 bit Microsoft® Windows application.

These models were optimised in terms of time and limited to using only 10,000 evaluations of the scheduling algorithm. Six cases were optimised for each project. In the first three cases the optimisations were performed using only one type of project input variable as variable, the other two being considered as fixed. A further three cases were considered, in which a combination of project input variables were considered as variable.

The results of these optimisations showed how optimising using only one type variable – which corresponded to a local optimisation – would not yield the best results. The optimisation which corresponded to the global approach – the

optimisation which considered PRU and activity priority as variable – yielded the best performance.

Further analysis of the performance of the optimisations showed the following to be the case:

- The global approach to optimisation performed better even at low run numbers, though the difference was more marked at higher run numbers. This demonstrated the possibility that the global approach might be able to provide better solutions even when low levels of computational effort (in terms of time and computer power) are used.

- The most significant improvements obtained by performing a global optimisation, rather than a local one, were found where the increase in scope created by implementing the global approach was large. As this increase in scope is expected to be large for a real project, the global approach is expected to be effective.

- It was shown that the ability of the optimisations to obtain or approach the true global optimum with a fixed amount of computational effort would diminish as project size and complexity increased. This was expected, and shows how the global approach must aim to find the best solution given the project and the computational effort available to optimise it.

- The times taken for each execution of the scheduling algorithm were found to be approximately linearly dependent upon the number of activities and the initial duration. This allowed predictions of execution times for much larger projects to be evaluated. It was shown that the optimisation of a real, large project was feasible using existing computer technology.

8.2 Application Of The Model To A Real Project

A model of the construction phase of the new car park at the Johannesburg International Airport was obtained and optimised. Data for all resources and their

consumption were not available. However, sufficient detail was available for the concrete pour and support work activities to be able to model the associated resources quite effectively.

The model was prepared using the same steps as for the first set of problems. Much of the data was missing, and some not available explicitly. Therefore the existing data was checked thoroughly to ensure that it constituted an accurate model of the project. Some modifications to activity relationships and resource allocations were required. The manpower allocations for the support work and decking activities were also not available. Three possible sets of assumptions for the values of these were identified, and so it was decided to optimise three cases, one for each set of assumptions.

A number of additional analytical techniques were identified and incorporated into the model. These permitted the modelling of the teams which operated on the support work and decking activities, and the project costs. Two project input variables were identified: activity priority and the team size for the support work and decking activities.

The optimisations were performed in terms of both cost and time for each case. The performance of a line-of-balance (LOB) optimisation was also evaluated to allow the global approach to be compared to a local optimisation technique. The performance of the global optimisations was better than the performance of the LOB for all cases in terms of both cost and time, particularly for case 2, which was the most realistic of the three cases.

Once these optimal solutions had been determined, a set of optimal cost-time curves was obtained for each case. These showed clearly the trade-off between time and cost between the cost-optimal and time-optimal solutions.

Further analysis was made, and the following characteristics of the optimal solutions identified:

- The cost optimal solution tended to have a shorter duration than the initial duration. This showed that a solution which was efficient in term of cost could also be efficient in terms of time.

- It was found that, for efficient solutions on the cost time curve, more variation in duration could be found with a relatively fixed cost than vice-versa. However, this was found to be due to the influence of the team size variable. As more resources would be variable in a true global project model it was asserted that this phenomenon could not be extrapolated to a project for which a full set of data was available.

- The times of the optimisations were found to be within 13% of those predicted by the regression model from the RCPS optimisations. This confirmed that the relationship between project size and execution time was approximately linear, and that optimisation of a large project was indeed a feasible undertaking.

8.3 Further Work

The purpose of the work in this thesis was to demonstrate that a global approach to project optimisation was feasible and beneficial, as well as providing examples of its implementation. However, there is still much development required before a true global optimisation model is fit for use of the industry. Some of the future development suggested here has been inferred from the literature, and some from the results of the two optimisations performed.

Firstly, it is necessary that the actual data used in the project is complete and accurate. In the optimisation of the project it was found that some relationships had been modelled in a way which did not accurately reflect the true relationships between the real activities. The optimisations of these three projects also showed how small changes in the assumptions could alter the optimal solutions found significantly. Therefore it is necessary to model the physical project as accurately as possible, rather than simply obtaining a model which appears to work. This philosophy must be key to all future development.

While this is the most important point about the future development of the model, it is also necessary to consider the accuracy and efficiency of the model and its optimising algorithm. A number of avenues for development of this have been identified as follows:

- The analytical techniques will have to be expanded. This is the only way in which the model itself can be made a more accurate representation of the real project. Those techniques identified in this thesis will be insufficient to model the complexities of the different stages of a major construction project. It is essential that such techniques are approved as realistic not only by academics, but also by the consultants and contractors. Indeed, it may be necessary to create a whole new range of analytical techniques for some types of project.

- The model discussed so far has been deterministic. However, real projects involve risk, and the exact outcome of many different parts of a project is not known. The consideration of risk is important to the accuracy of the project, and should therefore be included within the model. This might be achieved by making the model a stochastic one, or by using fuzzy sets[Brown, 1992].

- The efficiency of the optimising algorithm will have to be improved. The genetic algorithm was found not to perform very well in highly constrained situations. Therefore it may be necessary to use a different technique in those situations. Furthermore, it may also be possible to create a genetic algorithm which performs better generally. As the optimisations demand so much computational resources, any means of increasing the speed with which the optimisation is able to find good solutions is important.

All these aspects are important for developing what could be a very significant tool for project planning. The potential benefits of its implementation as a project planning tool are huge. However, it must be remembered that the holistic, global, approach was born out of the criticism of existing analytical techniques from the literature, and that this literature is also very critical of the way quantitative techniques were depended upon too much by project planners. Therefore, however

accurate the final model is, it must be used in conjunction with considerations of the qualitative aspects of the project which the planner must allow for. If it is depended upon to do all the planning, at the expense of considering these qualitative aspects of the project, then the detrimental impact of not considering these qualitative aspects could outweigh the benefits of using the model. Only when these considerations are made side by side will the full potential of the model as a planning aid be realised.

Bibliography

1. AbouRizk, S. M. and Wales, R. J. (1997) *Combined Discrete-Event/Continuous Simulation for Project Planning,* Journal of Construction Engineering and Management, **123,** 11-20.

2. Adby, P. R. and Dempster, M. A. H. (1982) *Introduction To Optimisation Methods,* Chapman & Hall.

3. Ahuja, H. N. and Nandakumar, V. (1985) *Simulation to Model Forecast Project Completion Time,* Journal of Construction Engineering and Management, **111,** 325-342.

4. Akkan, C. (1996) *Overtime Scheduling: An Application in Finite Capacity Real Time Scheduling,* Journal of the Operational Research Society, **47,** 1137-1149.

5. Arditti, D. and Albulak, M. Z. (1986) *Line-Of-Balance Scheduling In Pavement Construction,* Journal of Construction Engineering and Management, **112,** 411-424.

6. Arora, R. K. and Sachdeva, R. K. (1989) *Distributed Simulation of Resource Constrained Project Scheduling,* Computers And Operational Research, **16,** 295-304.

7. Baker, R. H. (1989) *Guide to SQL Database Management for IBM PCs and Compatibles,* Scott, Foresman and Company.

8. Becassi, W. and Tukel, O. I. (1996) *A New Framework For Determining Critical Success/Failure Factors In Projects,* International Journal Of Project Management, **14,** 141-151.

9. Bentley, J. L. (1982) *Writing Efficient Programs,* Prentice - Hall.

10. Boctor, F. F. (1993) *Heuristics for Scheduling Projects With Resource Restrictions and Several Resource Duration Modes,* International Journal of Production Research, **31,** 2547-2588.

11. Brown, C. B. (1992) *Fuzzy Sets Applied To Civil Engineering Systems,* Optimisation And Artificial Intelligence In Civil And Structural Engineering, **1,** 19-33.

12. Buchanan, D. A. (1991) *Beyond Content And Control: Project Vulnerability And The Process Agenda,* International Journal Of Project Management, **9,** 233-239.

13. Campbell, P. F., Miller, G. L. and Speechley, A. E. (1997) *East Spar Development: First Offshore Gas Field in Australia Developed by Alliance Approach,* Offshore Europe '97, Aberdeen, Society of Petroleum Engineers

14. Chambers, L. (1995) *Practical Handbook of Genetic Algorithms: Applications,* CRC Press.

15. Conley, W. (1988) *Controlling Costs By Finding The Shortest Route: A Multistage Monte Carlo Optimization Approach,* Cost Engineering, **30,** 20-24.

16. Cox, M. A. A. (1995) *Simple Normal Approximation To The Completion Time Distribution For A Pert Network,* International Journal Of Project Management, **13,** 265-270.

17. Davis, E. W. and Heidorn, G. E. (1971) *An Algorithm for Optimal Project Scheduling Under Resource Constraints,* Management Science, **17,** B803-816.

18. Davis, E. W. and Patterson, J. H. (1975) *A Comparison of Heuristic and Optimum Solutions in Resource Constrained Project Scheduling,* Management Science, **21,** 944-955.

19. Davis, K. R., Stam, A. and Grzybowski, R. A. (1992) *Resource Constrained Project Scheduling with Multiple Objectives: a Decision Support Approach,* Computers And Operational Research, **19,** 657-669.

20. De Wit, J. H., Willy (1990) *An Evaluation of Microcomputer-Based Software Packages for Project Management,* European Journal of Operational Research, **49,** 102-139.

21. Diaz, C. F. and Hadapriono, F. C. (1993) *Nondeterministic Networking Methods,* Journal of Construction Engineering and Management, **119,** 40-57.

22. Dubois, D. (1980) *Fuzzy Sets And Systems: Theory And Applications,* Academic Press.

23. Easa, S. M. (1992) *Optimum Cash-flow Scheduling Of Construction Projects,* Civil Engineering Systems, **9,** 69-85.

24. Elmaghraby, S. E. (1993) *Resource Allocation Via Dynamic Programming in Activity Networks,* European Journal of Operational Research, **64,** 199-215.

25. Elmaghraby, S. E. (1995a) *Activity nets: A Guided Tour Through Some Recent Developments,* European Journal of Operational Research, **82,** 383-408.

26. Elmaghraby, S. E. (1995b) *Activity Nets: a Guided Tour Through Some Recent Developments,* European Journal of Operational Research, **82,** 383-408.

27. Elms, D. G. (1992) *Network Structuring Algorithms,* Optimisation And Artificial Intelligence In Civil And Structural Engineering, **1,** 43-60.

28. European Construction Institute (1992) *Total Project Management Of Construction Safety, Health And Environment,* Report No.: 0-7277-1923-8, European Construction Institute,

29. Flanagan, R., Norman, G., Meadows, J. and Robinson, G. (1989) *Life Cycle Costing: Theory And Practice,* BSP Professional Books.

30. Fleetwoood-Bird, H. M. (1998) *Project Management: Theory and Practice,* Unpublished Honours Degree Thesis, Dept. Civil and Environmental Engineering, University Of Edinburgh, Edinburgh

31. Foulds, L. R. (1981) *Optimization Techniques,* Springer Verlag, New York.

32. Frame, J. D. (1994) *The New Project Management,* Josset Bass.

33. Grierson, D. E. and Hajela, P. (1996) *Emergent Computing Methods in Engineering Design. Applications of Genetic Algorithms and Neural Networks,* Springer-Verlag.

34. Guang-Yuan, W. and Dong-Yad, T. (1992) *Global Optimisation Of Large Scale Engineering Systems,* Engineering Optimization, **18**.

35. Hackman, S. T. and Leachman, R. C. (1989) *A General Framework for Modeling Production,* Management Science, **35,** 478-495.

36. Hallefjord, Å., Jörnsten, K. O. and Värbrand, P. (1993) *Solving Large Scale Generalized Assignment Problems - an Aggregation/Disaggregation Approach,* European Journal of Operational Research, **64,** 103-114.

37. Hand, A. F. (1997) *Curlew Project: An Elegant Solution,* Offshore Europe '97, Aberdeen, Society of Petroleum Engineers

38. Hegazy, T. (1999) *Optimization of Resource Allocation and Leveling Using Genetic Algorithms,* Journal Of Construction Engineering and Management, **125,** 167-175.

39. Heindel, L. E. and Kasten, V. A. (1996) *Next Generation PC Based Project Management Systems: The Path Forward,* International Journal Of Project Management, **14,** 249-253.

40. Hemmens, P. D. (1997) *Britannia: Preparing for Superior Operational Performance,* Offshore Europe '97, Aberdeen, Society of Petroleum Engineers

41. Hendrickson, C., Martinelli, D. and Rehak, D. (1987) *Hierarchical Rule-Based Activity Duration Estimation,* Journal of Construction Engineering and Management, **113,** 288-301.

42. Holmes, R. (1989) *Optimising Performance And Quality In Construction,* Report No.: 02626632, Chartered Institute Of Building,

43. Humphreys, K. K. (1991) *Jelen's Cost And Optimisation Engineering,* McGraw-Hill.

44. Igelmund, G. and J.Radermacher, F. (1983) *Preselective Strategies For The Optimization Of Stochastic Project Networks Under Resource Constraints,* Networks, **13,** 1-28.

45. Isahara, J. (1992) *Project Finance: Financing Schemes For Large Capital Projects*, MBA & DIC Thesis, Dept. Management School, Imperial College Of Science, Technology And Medicine (University Of London), London

46. Jaafari, A. (1996) *Time And Priority Allocation Scheduling Technique For Projects,* International Journal Of Project Management, **14,** 289-299.

47. Karkaria, D. J. (1987) *A Prototype Intelligent Project Management Environment*, MSc & DIC Thesis, Dept. Department of Computing, Imperial College Of Science, Technology And Medicine (University Of London), London

48. Kavanagh, C. P. (1985) *SIREN: A Repetitive Construction Simulation Model,* Journal of Construction Engineering and Management, **111,** 308-323.

49. Kazanji, A. Z. (1991) *Project Management Performance Analysis*, MBA & DIC Thesis, Dept. Management School, Imperial College Of Science, Technology And Medicine (University Of London), London

50. Kennington, J. L. and Helagson, R. V. (1980) *Algorithms For Network Programming,* John Wiley & Sons.

51. Kerzner, H. (1995) *Project Management: A Systems Approach to Planning, Scheduling And Controlling,* International Thompson Publishing.

52. Khattab, M. M. and Choobineh, F. (1991) *A New Approach For Project Scheduling With A Limited Resource,* International Journal Of Production Research, **29,** 185-198.

53. Knoepfel, H. (1989) *Cost And Quality Control In The Project Cycle,* International Journal Of Project Management, **7,** 229-235.

54. Kruglenski, D. J. (1994) *Inside Visual C++,* Microsoft Press.

55. Kurihara, K., Seki, S. and Akashi, K. (1985) *An Optimization Method For Investment In A Project Represented By GERT Network,* Electrical Engineering In Japan, **104,** 135-143.

56. Leachman, R. C. and Kim, S. (1993) *Revised Critical Path Method for Networks Inludind Both Overlap Relationships and Variable Duration Activities,* European Journal of Operational Research, **64,** 229-248.

57. Lee, J.-K. and Kim, Y.-D. (1996) *Search Heuristics For Resource Constrained Project Scheduling,* Journal of the Operational Research Society, **47,** 678-689.

58. Li, R. K.-Y. and Willis, R. J. (1993) *Resource Constrained Scheduling Within Fixed Project Durations,* Journal of the Operational Research Society, **44,** 71-80.

59. Li, S. (1996) *New approach for optimization of overall construction schedule,* Journal of Construction Engineering and Management, **122,** 7-13.

60. Lockyer, K. and Gordon, J. (1991) *Critical Path Analsis And Other Project Network Techniques,* Pitman.

61. Madhaviji, N. M. (1991) *Process Cycle,* Software Engineering Journal, **6,** 234-242.

62. Maren, A. J., Hartson, C. T. and Pap, R. M. (1990) *Handbook of Neural Computing Applications,* Academic Press Ltd., London.

63. McGartland, M. R. and Hendrickson, C. T. (1985) *Expert Systems For Construction Project Monitoring,* Journal of Construction Engineering and Management, **111,** 293-307.

64. McIntosh, L., Carrier & Laing Ltd. (1997) *Johannesburg International Airport Project Analysis: Scheme 1*, Report No.: , , Johannesburg

65. Mirham, G. A. (1972) *Simulation: Statistical Foundations and Methodology,* Academic Press, Inc.

66. Miskawi, Z. (1993) *A Numerical Analysis Method For Computing Schedule And Resource Forecasting For Industrial Projects,* Construction Management And Economics, **11,** 411-420.

67. Moselhi, O. and El-Rayes, K. (1993) *Scheduling Of Repetitive Projects With Cost Optimization,* Journal of Construction Engineering and Management, **119,** 681-697.

68. Moselhi, O. and Lorterapong, P. (1993) *Near Optimal Solutions For Resource Constrained Scheduling Problems,* Construction Management And Economics, **11,** 293-303.

69. M'silti, A. and Tolla, P. (1993) *An Interactive Multiobjective Nonlinear Programming Procedure,* European Journal of Operational Research, **64,** 115-125.

70. Nelder, J. A. and Mead, R. (1965) *A Function Method For Function Minimisation,* The Computer Journal, **17,** 308-313.

71. Norbis, M. I. and Smith, J. M. (1988) *Multiobjective, Multi-level Heuristic for Dynamic Resource Constrained Scheduling Problems,* European Journal of Operational Research, **33,** 30-41.

72. Patterson, J. H. (1984) *A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem,* Management Science, **30,** 854-867.

73. Patterson, J. H. and Huber, W. D. (1974) *A Horizon Varying Zero-one Approach to Project Scheduling,* Management Science, **20,** 990-998.

74. Patterson, J. H., Talbot, F. B., Slowinski, R. and Weglarz, J. (1990) *Computational Experience with a Backtracking Algorithm for Solving a General Class of Precedence and Resource Constrained Project Scheduling Problems,* European Journal of Operational Research, **49,** 68-79.

75. Phillips, S. (1996) *Project Management Duration/Resource Tradeoff Analysis: an Application of the Cut Search Approach,* Journal of the Operational Research Society, **47,** 697-701.

76. Phillips, S. and Dessouky, M. I. (1977) *Solving the Project Time/Cost Tradeoff Problem Using the Minimum Cut Concept,* Management Science, **24,** 393-400.

77. Pidd, M. (1989) *Computer Modelling For Discrete Simulation,* John Wiley & Sons.

78. Rawlins, G. J. E. (1991) *Foundations of Genetic Algorithms,* Morgan Kaufmann.

79. Raz, T. and Marshall, B. (1996) *Effect Of Resource Constraints On Float Calculations In Project Networks,* International Journal Of Project Management, **14,** 241-248.

80. Robinson, D. R. (1975) *A Dynamic Programming Solution To Cost Time Tradeoff For CPM,* Management Science, **22,** 158-166.

81. Rodrigues, A. and Bowers, J. (1996) *The Role Of System Dynamics In Project Management,* International Journal Of Project Management, **14,** 213-220.

82. Russell, D. L. (1970) *Optimization Theory,* W. A. Benjamin, Inc.

83. Schrage, L. (1972) *Solving Resource Constrained Network Problems by Implicit Enumeration - Preemptive Case,* Operations Research, **20,** 668-677.

84. Simpson, W. P. and Patterson, J. H. (1996) *A Multiple Tree Search Procedure for the Multiple Resource Constrained Scheduling Problem,* European Journal of Operations Research, **89,** 525-542.

85. Slowinski, R. (1981) *Two Approaches to Problems of Resource Allocation Among Project Activities - a Comparison,* Journal of the Operational Research Society, **31,** 711-723.

86. Soroush, H. M. (1994) *The Most Critical Path In A PERT Network: A Heuristic Approach,* European Journal of Operations Research, **78,** 93-105.

87. Stepniak, T. P. (1990) *Project Management System: Evaluation Of Object Oriented Paradigm,* MSc Thesis, Dept. Department of Computing, Imperial College Of Science, Technology And Medicine (University Of London), London

88. Stuckman, B. E., Care, M. C. and Stuckman, P. L. (1990) *System Optimization Using Experimental Evaluation Of Design Performance,* Engineering Optimization, **16,** 275-290.

89. Talbot, F. B. (1982) *Resource-constrained Project Scheduling with Resource Tradeoffs: the Nonpreemptive Case,* Management Science, **28,** 1197-1210.

90. Talbot, F. B. and Patterson, J. H. (1978) *An Efficient Integer Programming Algorithm with Network Cuts for Solving Resource Constrained Scheduling Problems,* Management Science, **24,** 1163-1174.

91. Teicholz, P. (1994) *Forecasting Final Cost And Budget Of Construction Projects,* Journal of Computing in Civil Engineering, **7,** 511-529.

92. Thabet, W. Y. and Beliveau, Y. J. (1994) *Modeling Work Space To Schedule Repetitive Floors In Multistorey Buildings,* Journal of Construction Engineering and Management, **120,** 96-116.

93. Vavasis, S. A. (1991) *Nonlinear Optimization: Complexity Issues,* Oxford University Press, Oxford.

94. Walker, A. (1989) *Project Management And Construction,* BSP Professional Books.

95. Ward, S. C. and Chapman, C. B. (1995a) *Extending The Use Of Risk Analysis In Project Management,* International Journal Of Project Management, **13.**

96. Ward, S. C. and Chapman, C. B. (1995b) *Risk Management Perspective On The Project Life Cycle,* International Journal Of Project Management, .

97. Weist, J. D. (1967) *A Heuristic Model For Scheduling Large Projects With Limited Resources,* Management Science, **13,** B359.

98. Williams, H. P. (1993a) *Model Building in Mathematical Programming,* John Wiley & Sons.

99. Williams, H. P. (1993b) *Model Solving in Mathematical Programming,* John Wiley & Sons.

100. Wollmer, R. D. (1985) *Critical Path Planning Under Uncertainty,* Mathematical Programming Study, **25,** 164-171.

101. Woolery, J. C. and Crandall, K. C. (1983) *Stochastic Network Model for Planning Scheduling,* Journal of Construction Engineering and Management, **109,** 342-354.

102. Yang, K.-K. and Sum, C.-C. (1993) *A Comparison of Resource Allocation and Activity Scheduling Rules in a Dynamic Multi-Project Environment,* European Journal of Operational Research, **11,** 207-218.

103. Yau, C. and Ritchie, E. (1990) *Project Compression: a Method for Speeding up Resource Constrained Projects Which Preserve the Activity Schedule,* European Journal of Operational Research, **49,** 140-152.

104. Yeo, K. T. (1991) *Project Cost Sensitivity And Variability Analysis,* International Journal Of Project Management, **9,** 111-116.

Appendix I Schedule Test: Problem 110

The scheduling algorithm was verified by observing the schedule of problem 110, optimised by priority. This problem had 27 activities and a duration of 33 days. There were three resources, each of which had a ceiling of daily use of 10 units. The schedule times and resource consumptions for all the activities are shown in the figure below. Additionally, the times that the activities were added to the queue have been calculated (TAQ – Time Added to Queue), as well as how long the activities spent in the queue, in days (TQ – Time in Queue).

ID	Duration	Priority	EST	EFT	TAQ	TQ	Resource rates		
							1	2	3
1	0	#N/A	0	0	0	0	0	0	0
2	2	45	4	6	0	4	3	5	2
3	4	72	0	4	0	0	5	4	3
4	1	75	0	1	0	0	5	2	2
5	1	75	6	7	6	0	4	1	4
6	2	75	1	3	1	0	5	5	4
7	4	25	6	10	1	5	3	5	2
8	2	95	4	6	4	0	2	4	4
9	4	40	12	16	10	2	3	2	2
10	7	14	6	13	6	0	3	2	4
11	3	85	7	10	7	0	3	3	2
12	2	63	3	5	3	0	4	1	4
13	2	75	17	19	16	1	1	4	4
14	4	60	13	17	13	0	2	2	2
15	2	75	10	12	10	0	5	5	4
16	5	70	12	17	10	2	1	5	4
17	3	86	17	20	17	0	4	5	4
18	1	95	19	20	19	0	3	2	3
19	5	37	20	25	17	3	5	3	3
20	6	35	20	26	17	3	2	4	6
21	1	67	26	27	26	0	1	6	2
22	3	34	25	28	20	5	3	2	1
23	2	19	26	28	25	1	1	0	4
24	7	36	20	27	20	0	2	2	1
25	5	75	28	33	28	0	0	1	3
26	5	68	28	33	28	0	2	2	2
27	0	#N/A	33	33	33	0	0	0	0

Activity schedule data.

The start and finish times generated here show that the program does not violate any precedence constraints. The times at which any activity is added to the queue must be less than or equal to the time at which it was scheduled. This is the case. The fact that many of the activities are scheduled straight away also demonstrates that the

scheduling algorithm is adding activities to the queue no later than it should be. The fact that some of the activities may not be scheduled until later can be demonstrated by considering the resource consumptions, which are shown in the table below. The values of this histogram were output by the program, and have also been calculated using a spreadsheet for verification. The values of both of these calculations are shown.

	Resource 1				Resource 2				Resource 3			
	Output		Spreadsheet		Output		Spreadsheet		Output		Spreadsheet	
Day	Use	Cum.	Use	Cum.	Use	Cum.	Use	Cum.	Use	Cum.	Use	Cum.
0	10	10	10	10	6	6	6	6	5	5	5	5
1	10	20	10	20	9	15	9	15	7	12	7	12
2	10	30	10	30	9	24	9	24	7	19	7	19
3	9	39	9	39	5	29	5	29	7	26	7	26
4	9	48	9	48	10	39	10	39	10	36	10	36
5	5	53	5	53	9	48	9	48	6	42	6	42
6	10	63	10	63	8	56	8	56	10	52	10	52
7	9	72	9	72	10	66	10	66	8	60	8	60
8	9	81	9	81	10	76	10	76	8	68	8	68
9	9	90	9	90	10	86	10	86	8	76	8	76
10	8	98	8	98	7	93	7	93	8	84	8	84
11	8	106	8	106	7	100	7	100	8	92	8	92
12	7	113	7	113	9	109	9	109	10	102	10	102
13	6	119	6	119	9	118	9	118	8	110	8	110
14	6	125	6	125	9	127	9	127	8	118	8	118
15	6	131	6	131	9	136	9	136	8	126	8	126
16	3	134	3	134	7	143	7	143	6	132	6	132
17	5	139	5	139	9	152	9	152	8	140	8	140
18	5	144	5	144	9	161	9	161	8	148	8	148
19	7	151	7	151	7	168	7	168	7	155	7	155
20	9	160	9	160	9	177	9	177	10	165	10	165
21	9	169	9	169	9	186	9	186	10	175	10	175
22	9	178	9	178	9	195	9	195	10	185	10	185
23	9	187	9	187	9	204	9	204	10	195	10	195
24	9	196	9	196	9	213	9	213	10	205	10	205
25	7	203	7	203	8	221	8	221	8	213	8	213
26	7	210	7	210	10	231	10	231	8	221	8	221
27	4	214	4	214	2	233	2	233	5	226	5	226
28	2	216	2	216	3	236	3	236	5	231	5	231
29	2	218	2	218	3	239	3	239	5	236	5	236
30	2	220	2	220	3	242	3	242	5	241	5	241
31	2	222	2	222	3	245	3	245	5	246	5	246
32	2	224	2	224	3	248	3	248	5	251	5	251

Resource use – program output and Spreadsheet calculations.

This table shows that the scheduling algorithm is calculating the resources properly, as they are completely consistent with the values calculated on the spreadsheet. It

also shows that the resource ceilings are not violated on any day, as a daily use above 10 units is not observed.

These values can be used to verify that those activities which were not scheduled straight away could not have been scheduled earlier. If the free resources available in the project from the day it was added to the queue to the day before it was scheduled are considered, it can be demonstrated that there would not be enough free resources to allow the activity to be scheduled on any of these days without violating the resource constraints.

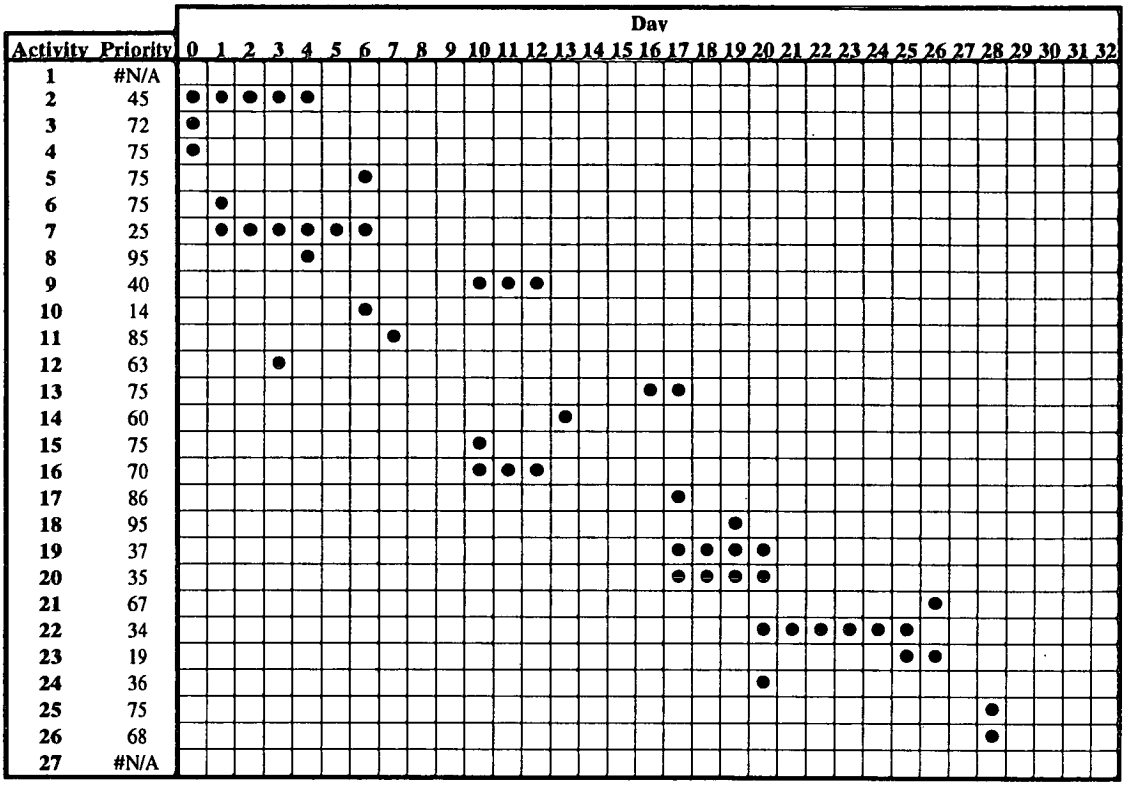
		Day	Resource 1	2	3
Activity 2	***Required***		*3*	*5*	*2*
	Available	0	0	4	5
		1	0	1	3
		2	0	1	3
		3	1	5	3
Activity 7	***Required***		*3*	*5*	*2*
	Available	1	0	1	3
		2	0	1	3
		3	1	5	3
		4	1	0	0
		5	5	1	4
Activity 9	***Required***		*3*	*2*	*2*
	Available	10	2	3	2
		11	2	3	2
Activity 13	***Required***		*1*	*4*	*4*
	Available	16	7	3	4
Activity 16	***Required***		*1*	*5*	*4*
	Available	10	2	3	2
		11	2	3	2

		Day	Resource 1	2	3
Activity 19	***Required***		*5*	*3*	*3*
	Available	17	5	1	2
		18	5	1	2
		19	3	3	3
Activity 20	***Required***		*2*	*4*	*6*
	Available	17	5	1	2
		18	5	1	2
		19	3	3	3
Activity 22	***Required***		*3*	*2*	*1*
	Available	20	1	1	0
		21	1	1	0
		22	1	1	0
		23	1	1	0
		24	1	1	0
Activity 23	***Required***		*1*	*0*	*4*
	Available	25	3	2	2

Resource consumption while delayed activities in queue.

This table shows that the scheduling algorithm is indeed unable to schedule the activities for those days which the activity is left in the queue. Therefore it is correct in delaying these activities. All that remains, then, is to check that the scheduling algorithm should be delaying these activities, and not others. Other activities may appear in the queue while these activities are in it, and be scheduled before them. Therefore, it is necessary to assert that the scheduling algorithm is indeed prioritising the activities correctly.

In order to show that this is the case, a table has been prepared which shows exactly when all the activities appear in the queue. Thus, when two activities appear at the same time, and one is scheduled and the other not, it is possible to check that the scheduling algorithm is scheduling the one with the highest priority, as it should be.



Queue for the project, showing which activities were present.

The delayed activities are all those which remain in the queue for more than one day. Those which appear only for one day are scheduled immediately, and therefore suffer no delay. It can be seen that those activities scheduled always have a higher priority number than any which, for the same period, are not scheduled.

In conclusion, therefore, the resource constrained scheduling algorithm in the program is working properly. This has been shown by verifying that the following criteria are met.

- The algorithm must add the activities to the queue at the right time.
- The algorithm must accurately calculate the resource consumptions.

- The algorithm must delay activities for which there are insufficient resources.
- The algorithm must prioritise activities correctly.

Appendix II Setting Up The GA

In chapter 4, three selection techniques and five replacement techniques were identified for the reproduction phase of the GA. The three selection techniques were *Fit-Fit*, *Random* and *Weighted Random*. The five replacement techniques were *Weakest Individual*, *Weakest Parent*, *Both Parents*, *Random* and *Weighted Random*. Furthermore, it was also necessary to determine the size of the population, and the number of parents to be selected each generation, as well as the crossover and mutation rates.

The values of these variables were all determined using a sensitivity analysis. This kind of analysis tries to determine the best value of one variable, before moving on to another. The first variable type to be determined was the Selection and replacement methods. All possible combinations of these two values were observed, under varying population size and number of parents to be selected. The parent sizes were selected appropriate to the size of population, such that a very low number of parent pairs were selected, the whole population was selected, and a value in between. This was to ensure that, in the first instance, the selection of the selection and replacement methods was not too influenced by the population size.

Three population sizes were used in all: 10, 30 and 100. 15 parent pairs were not observed for population size 30 because the results from selecting the whole population were so poor from population sizes 10 and 100. The optimisations were performed using 10,000 runs on the PRU & priority cases from problems 100-103, and the average performance, as a proportion of the target duration, measured. The results are as follows.

pop:10 parents: 1	**Replacement Method**				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	93%	92%	98%	93%	95%
Weighted Random	89%	91%	98%	94%	97%
Random	91%	91%	98%	96%	99%

pop:10 parents: 3	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	90%	93%	98%	96%	97%
Weighted Random	90%	92%	97%	96%	97%
Random	92%	91%	98%	97%	98%

pop:10 parents: 5	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	96%	93%	97%	98%	97%
Weighted Random	95%	92%	98%	97%	97%
Random	98%	90%	96%	99%	98%

pop:30 parents: 1	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	92%	92%	97%	90%	91%
Weighted Random	90%	92%	97%	94%	97%
Random	91%	91%	98%	96%	99%

pop:30 parents: 5	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	91%	93%	96%	91%	94%
Weighted Random	90%	92%	97%	95%	97%
Random	91%	91%	98%	96%	97%

pop:100 parents: 1	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	93%	93%	97%	90%	90%
Weighted Random	91%	92%	98%	94%	96%
Random	92%	93%	97%	97%	98%

pop:100 parents: 10	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	90%	94%	96%	90%	93%
Weighted Random	91%	92%	97%	94%	96%
Random	92%	92%	98%	96%	98%

pop:100 parents: 50	Replacement Method				
Selection Method	Weakest	Weakest Parent	Both Parents	Weighted Rnd.	Random
Fit-Fit	94%	94%	97%	96%	97%
Weighted Random	95%	93%	98%	96%	97%
Random	97%	93%	98%	97%	98%

The results of this analysis show that the best selection techniques are the Fit-Fit and the Weighted Random, and the best replacement techniques the Weakest Individual, the Weakest Parent and the Weighted Random. On this small sample of four projects, the population size of 10, selecting only one parent pair, performed the best. Therefore this will be used in further analysis of the selection and replacement techniques.

The selection of the Weakest Individual or the Fit-Fit methods over the Weighed Random method cannot be done without proper consideration of the values of the weights used by the Weighted Random method. If the weighting is small, i.e. if the variation in weight is between two similar values, then it approaches the Random technique. If it is large, it approaches the Fit-Fit/Weakest Individual technique. Therefore the methods are retested using a population of 10 and selecting only one parent pair per generation. This is performed on problems 100-104, and the results shown below.

	Select					
Replace	**Fit-fit**	**50 to 1**	**50 to 5**	**50 to 10**	**50 to 25**	**Random**
Weakest	91%	91%	91%	90%	91%	92%
Weak Parent	94%	91%	92%	93%	93%	92%
50 to 1	91%	92%	92%	93%	94%	94%
50 to 5	92%	94%	93%	93%	95%	96%
50 to 10	91%	93%	95%	95%	95%	96%
50 to 25	92%	94%	96%	95%	96%	96%
Random	94%	96%	96%	97%	96%	97%

Refined parent selection and replacement, with weights.

The best combination, as identified here, is 50 to 10 Weighted Random selection, and Weakest Individual replacement. Therefore, this combination of parent selection and replacement methods will be used.

Having determined the selection and replacement methods, it is necessary to determine the final values of the population and number of parent pairs. This is performed by considering a larger set of problems, 100-107, because there was found not to be as much variation in the results as was obtained with the

selection/replacement techniques. The population was varied between 10 and 40, in steps of 10, and the number of parent pairs between 1 and 3.

The results were as shown in the table below.

	Population			
Parents	10	20	30	40
1	90.7%	90.6%	90.6%	90.9%
2	91.5%	90.4%	90.2%	91.5%
3	91.3%	90.9%	90.4%	90.6%

Population and number of parent pairs.

The best performance was obtained when the population size was 30, and the number of parent pairs was 2. Therefore these two values were selected to be used in the GA.

The two values which still had to be determined were the crossover rate and the mutation rate. They were varied between 5 and 20, and the results of the combinations on problems 100-107 were as follows.

	Crossover		
Mutation	5	10	20
5	93.9%	94.3%	93.9%
10	91.3%	90.2%	90.4%
20	91.1%	90.9%	91.3%

Crossover and mutation rates.

The best value was obtained when both the crossover and mutation rates were 10. Therefore these rates will be used in the GA.

So the final values for all the variables within the GA are as follows.

Population 30

Parent Pairs 2

Parent Selection Weighted Random, with maximum weight (that taken by the best solution) 50, and minimum weight (taken by the worst) 10.

Replacement	Weakest individual.
Crossover Rate	10
Mutation Rate	10

Despite the fact that a number of variables were considered in combination, rather than one at a time, these results have not been obtained by a rigorous search through all the possible combinations. Although this sensitivity analysis used cannot guarantee that the combination of variables presented here is the best possible combination, it is still a good combination, and this makes it an efficient optimising algorithm, suitable for use in the preliminary analyses presented in this thesis.

Appendix III Results of the 110 RCPS Problems

The results of the optimisations of all 6 cases are shown for all 110 RCPS problems.

		Value						Proportion Of Target					
Proj.	Target	PRU	Priority	MEST	PRU & MEST	PRU & Priority	All Three	PRU	Priority	MEST	PRU & MEST	PRU & Priority	All Three
1	19	19	21	19	18	19	18	100%	111%	100%	95%	100%	95%
2	7	7	7	7	7	7	7	100%	100%	100%	100%	100%	100%
3	20	21	20	20	21	20	20	105%	100%	100%	105%	100%	100%
4	6	6	6	6	6	6	6	100%	100%	100%	100%	100%	100%
5	7	7	7	7	7	7	7	100%	100%	100%	100%	100%	100%
6	8	8	8	8	8	8	8	100%	100%	100%	100%	100%	100%
7	8	8	8	8	8	8	8	100%	100%	100%	100%	100%	100%
8	11	11	11	11	10	11	10	100%	100%	100%	91%	100%	91%
9	19	21	20	20	20	19	20	111%	105%	105%	105%	100%	105%
10	14	11	14	14	11	11	11	79%	100%	100%	79%	79%	79%
11	18	14	18	18	14	14	14	78%	100%	100%	78%	78%	78%
12	13	12	13	13	12	12	12	92%	100%	100%	92%	92%	92%
13	20	20	20	20	20	20	20	100%	100%	100%	100%	100%	100%
14	43	36	43	43	38	34	39	84%	100%	100%	88%	79%	91%
15	43	33	43	43	35	33	34	77%	100%	100%	81%	77%	79%
16	32	32	32	33	32	31	31	100%	100%	103%	100%	97%	97%
17	29	28	29	29	27	27	28	97%	100%	100%	93%	93%	97%
18	41	39	41	41	40	38	38	95%	100%	100%	98%	93%	93%
19	31	29	31	31	30	28	29	94%	100%	100%	97%	90%	94%
20	37	33	37	37	34	33	34	89%	100%	100%	92%	89%	92%
21	48	42	48	48	46	42	44	88%	100%	100%	96%	88%	92%
22	36	33	36	37	34	34	34	92%	100%	103%	94%	94%	94%
23	32	29	33	32	30	28	29	91%	103%	100%	94%	88%	91%
24	40	37	40	40	36	36	37	93%	100%	100%	90%	90%	93%
25	33	29	33	33	29	28	28	88%	100%	100%	88%	85%	85%
26	43	42	43	43	41	41	41	98%	100%	100%	95%	95%	95%
27	36	35	36	36	34	33	34	97%	100%	100%	94%	92%	94%
28	43	42	43	44	41	40	40	98%	100%	102%	95%	93%	93%
29	29	27	29	30	27	27	27	93%	100%	103%	93%	93%	93%
30	32	29	32	32	30	29	29	91%	100%	100%	94%	91%	91%
31	35	34	35	35	34	33	33	97%	100%	100%	97%	94%	94%
32	22	23	23	23	22	22	22	105%	105%	105%	100%	100%	100%
33	31	29	31	31	29	28	28	94%	100%	100%	94%	90%	90%
34	30	28	30	30	28	28	28	93%	100%	100%	93%	93%	93%
35	31	29	31	31	30	30	31	94%	100%	100%	97%	97%	100%
36	33	31	34	33	31	31	31	94%	103%	100%	94%	94%	94%
37	28	26	28	28	27	25	26	93%	100%	100%	96%	89%	93%
38	30	31	30	30	30	29	29	103%	100%	100%	100%	97%	97%
39	31	29	33	32	29	28	29	94%	106%	103%	94%	90%	94%
40	31	30	31	31	30	30	30	97%	100%	100%	97%	97%	97%
41	36	34	36	36	35	33	34	94%	100%	100%	97%	92%	94%
42	28	27	28	29	27	27	28	96%	100%	104%	96%	96%	100%
43	41	37	41	41	37	36	35	90%	100%	100%	90%	88%	85%
44	31	29	31	31	30	30	29	94%	100%	100%	97%	97%	94%
45	39	38	39	41	37	36	37	97%	100%	105%	95%	92%	95%
46	33	31	33	33	31	30	30	94%	100%	100%	94%	91%	91%
47	35	33	35	36	34	33	33	94%	100%	103%	97%	94%	94%

Proj.	Target	Value						Proportion Of Target					
		PRU	Priority	MEST	PRU & MEST	PRU & Priority	All Three	PRU	Priority	MEST	PRU & MEST	PRU & Priority	All Three
48	23	23	23	23	23	23	24	100%	100%	100%	100%	100%	104%
49	18	16	18	18	16	15	16	89%	100%	100%	89%	83%	89%
50	25	23	25	25	23	23	23	92%	100%	100%	92%	92%	92%
51	25	23	25	26	24	22	23	92%	100%	104%	96%	88%	92%
52	27	23	27	27	23	23	23	85%	100%	100%	85%	85%	85%
53	28	25	28	28	25	25	25	89%	100%	100%	89%	89%	89%
54	50	45	50	50	46	45	45	90%	100%	100%	92%	90%	90%
55	29	26	29	29	26	26	26	90%	100%	100%	90%	90%	90%
56	27	18	27	27	18	19	19	67%	100%	100%	67%	70%	70%
57	21	16	21	21	16	16	16	76%	100%	100%	76%	76%	76%
58	35	33	35	37	33	32	33	94%	100%	106%	94%	91%	94%
59	31	28	31	31	27	27	28	90%	100%	100%	87%	87%	90%
60	39	37	39	40	37	35	36	95%	100%	103%	95%	90%	92%
61	36	35	36	36	35	34	35	97%	100%	100%	97%	94%	97%
62	37	35	37	37	35	35	35	95%	100%	100%	95%	95%	95%
63	40	36	40	40	35	34	34	90%	100%	100%	88%	85%	85%
64	37	35	37	38	35	34	34	95%	100%	103%	95%	92%	92%
65	40	41	40	41	41	39	40	103%	100%	103%	103%	98%	100%
66	38	32	38	38	33	32	32	84%	100%	100%	87%	84%	84%
67	27	27	27	28	27	25	27	100%	100%	104%	100%	93%	100%
68	41	38	42	41	39	37	38	93%	102%	100%	95%	90%	93%
69	30	28	30	30	28	28	29	93%	100%	100%	93%	93%	97%
70	31	27	31	31	28	28	28	87%	100%	100%	90%	90%	90%
71	32	31	32	33	30	30	30	97%	100%	103%	94%	94%	94%
72	41	35	41	41	38	35	35	85%	100%	100%	93%	85%	85%
73	36	33	37	37	35	33	35	92%	103%	103%	97%	92%	97%
74	30	27	30	31	27	25	27	90%	100%	103%	90%	83%	90%
75	34	32	35	34	32	31	31	94%	103%	100%	94%	91%	91%
76	43	34	44	43	35	33	35	79%	102%	100%	81%	77%	81%
77	64	59	64	64	59	57	58	92%	100%	100%	92%	89%	91%
78	53	50	55	54	50	49	49	94%	104%	102%	94%	92%	92%
79	45	42	47	46	41	41	41	93%	104%	102%	91%	91%	91%
80	38	38	38	40	37	36	37	100%	100%	105%	97%	95%	97%
81	36	33	36	37	33	33	33	92%	100%	103%	92%	92%	92%
82	34	30	36	35	30	29	30	88%	106%	103%	88%	85%	88%
83	34	27	34	34	28	27	28	79%	100%	100%	82%	79%	82%
84	33	25	33	33	26	25	26	76%	100%	100%	79%	76%	79%
85	31	25	31	31	24	24	24	81%	100%	100%	77%	77%	77%
86	31	22	31	31	23	22	23	71%	100%	100%	74%	71%	74%
87	29	28	29	29	27	26	27	97%	100%	100%	93%	90%	93%
88	40	36	40	40	36	34	36	90%	100%	100%	90%	85%	90%
89	31	28	31	31	27	27	28	90%	100%	100%	87%	87%	90%
90	39	38	40	39	38	37	39	97%	103%	100%	97%	95%	100%
91	35	34	37	37	34	32	33	97%	106%	106%	97%	91%	94%
92	28	26	28	29	26	26	26	93%	100%	104%	93%	93%	93%
93	26	24	26	27	24	23	24	92%	100%	104%	92%	88%	92%
94	36	32	37	37	33	32	33	89%	103%	103%	92%	89%	92%
95	33	31	33	33	31	30	31	94%	100%	100%	94%	91%	94%
96	26	26	27	26	26	26	26	100%	104%	100%	100%	100%	100%
97	30	32	31	31	30	29	30	107%	103%	103%	100%	97%	100%
98	41	38	42	43	40	38	39	93%	102%	105%	98%	93%	95%
99	37	34	38	38	35	33	34	92%	103%	103%	95%	89%	92%
100	33	31	33	34	30	30	32	94%	100%	103%	91%	91%	97%

Proj.	Target	Value						Proportion Of Target					
		PRU	Priority	MEST	PRU & MEST	PRU & Priority	All Three	PRU	Priority	MEST	PRU & MEST	PRU & Priority	All Three
101	75	71	77	79	78	71	77	95%	103%	105%	104%	95%	103%
102	83	73	83	86	82	70	83	88%	100%	104%	99%	84%	100%
103	56	48	56	57	55	47	55	86%	100%	102%	98%	84%	98%
104	79	72	79	79	79	74	79	91%	100%	100%	100%	94%	100%
105	76	71	76	77	77	71	75	93%	100%	101%	101%	93%	99%
106	60	56	60	61	59	54	60	93%	100%	102%	98%	90%	100%
107	78	71	78	78	76	71	76	91%	100%	100%	97%	91%	97%
108	61	55	61	62	63	54	61	90%	100%	102%	103%	89%	100%
109	60	56	61	61	59	53	61	93%	102%	102%	98%	88%	102%
110	50	46	51	51	48	45	48	92%	102%	102%	96%	90%	96%

Appendix IV Resource Histograms for the Car Park At JIA

The resource histograms of both the cost and time optimisations of the three cases for the Car Park at the Johannesburg International Airport, from chapter 7, are shown in this section. In addition, the figures for analysis of the days above certain levels of efficiency are shown in full.

	Efficiency Over									
	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
Case 1 cost	0.00	9.65	9.89	10.61	9.05	9.90	9.90	23.90	26.67	26.67
Case 1 time	2.09	2.00	2.72	2.79	8.10	10.53	11.06	11.11	13.13	18.83
Case 2 cost	20.92	59.00	184.50	184.50	184.50	371.00	371.00	371.00	371.00	371.00
Case 2 time	0.00	4.15	19.07	118.67	179.50	361.00	361.00	361.00	361.00	361.00
Case 3 cost	1.67	2.96	4.88	5.57	6.00	5.96	6.59	19.14	19.36	19.36
Case 3 time	1.86	3.23	3.71	3.81	4.17	4.20	6.14	26.70	30.44	30.44

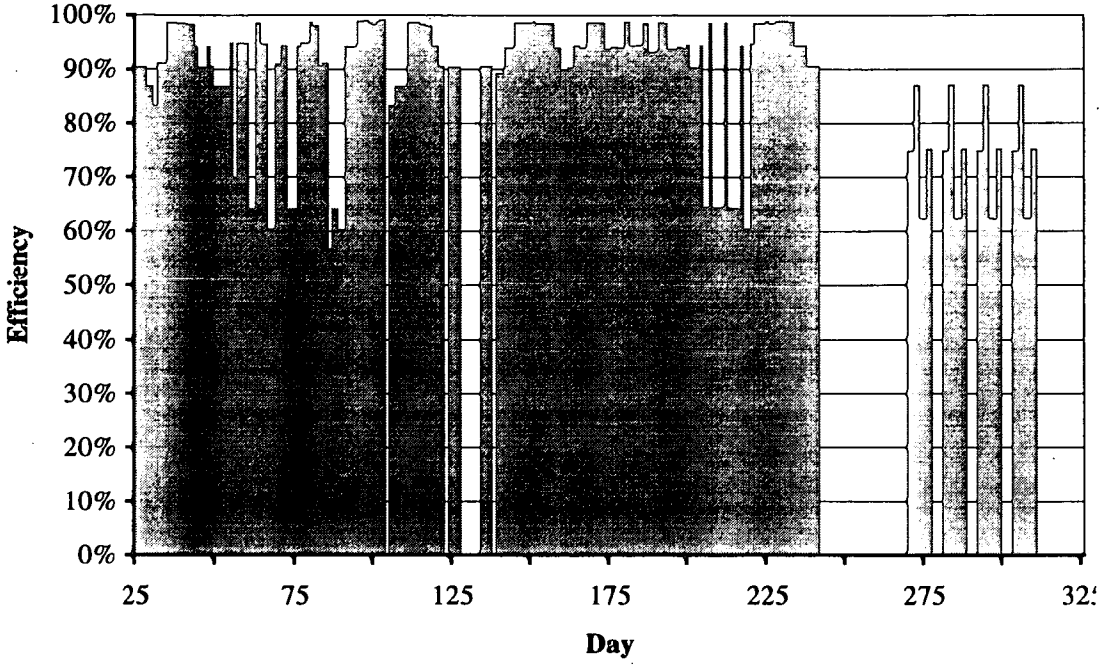
Average consecutive days above efficiency levels.

	Efficiency Over									
	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
Case 1 cost	13.00	43.00	66.00	66.00	66.00	66.00	66.00	103.00	103.00	103.00
Case 1 time	5.00	5.00	6.00	6.00	34.00	63.00	63.00	63.00	63.00	67.00
Case 2 cost	68.00	257.00	332.00	332.00	332.00	371.00	371.00	371.00	371.00	371.00
Case 2 time	0.00	6.00	105.00	322.00	330.00	361.00	361.00	361.00	361.00	361.00
Case 3 cost	3.00	6.00	13.00	25.00	33.00	33.00	33.00	96.00	96.00	96.00
Case 3 time	3.00	7.00	9.00	12.00	17.00	17.00	24.00	65.00	93.00	93.00

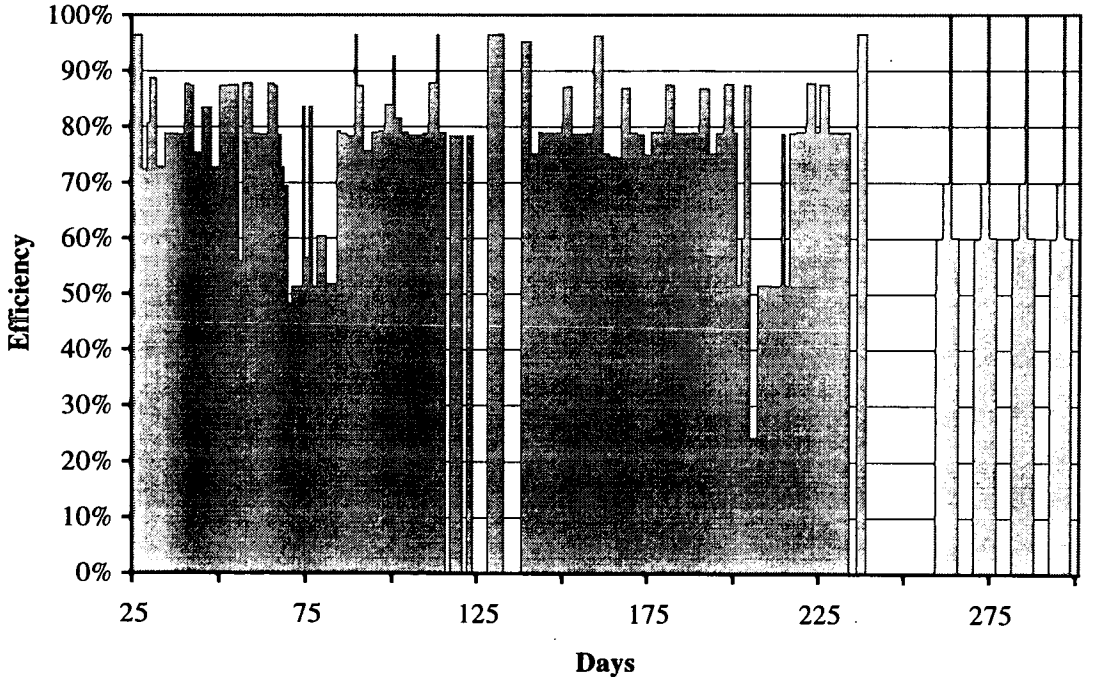
Maximum consecutive days above efficiency levels.

	Efficiency Over									
	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
Case 1 cost	0.00	164.00	188.00	191.00	199.00	208.00	208.00	239.00	240.00	240.00
Case 1 time	23.00	24.00	68.00	78.00	170.00	179.00	188.00	200.00	210.00	226.00
Case 2 cost	272.00	354.00	369.00	369.00	369.00	371.00	371.00	371.00	371.00	371.00
Case 2 time	0.00	54.00	286.00	356.00	359.00	361.00	361.00	361.00	361.00	361.00
Case 3 cost	10.00	74.00	122.00	128.00	132.00	137.00	191.00	268.00	271.00	271.00
Case 3 time	13.00	84.00	115.00	118.00	121.00	126.00	172.00	267.00	274.00	274.00

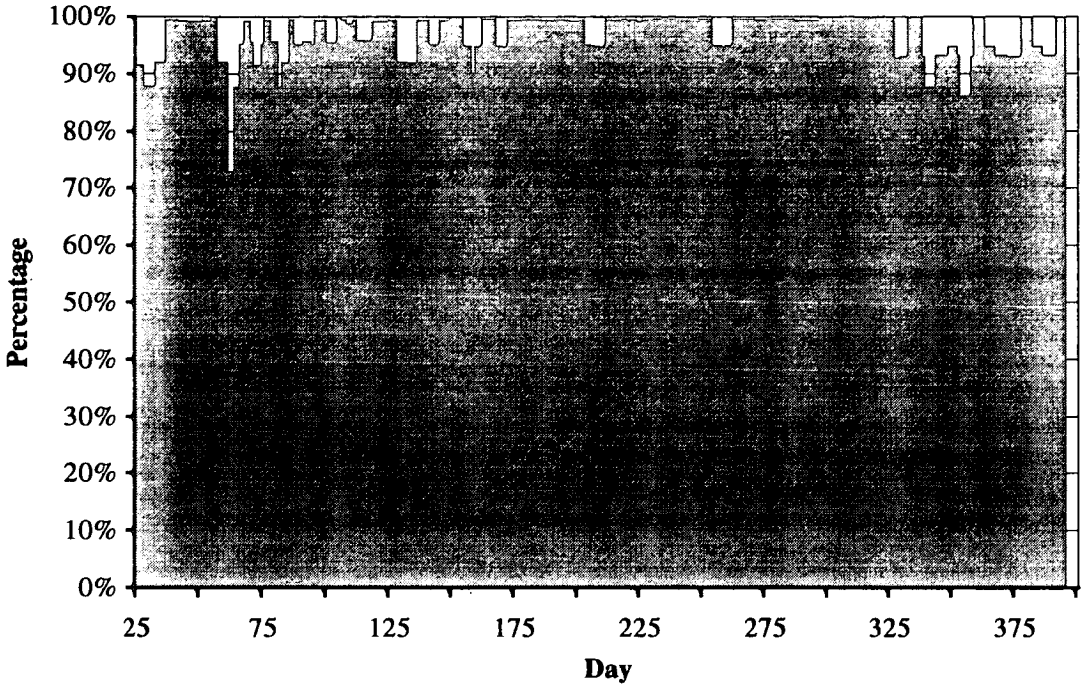
Total days above efficiency levels.



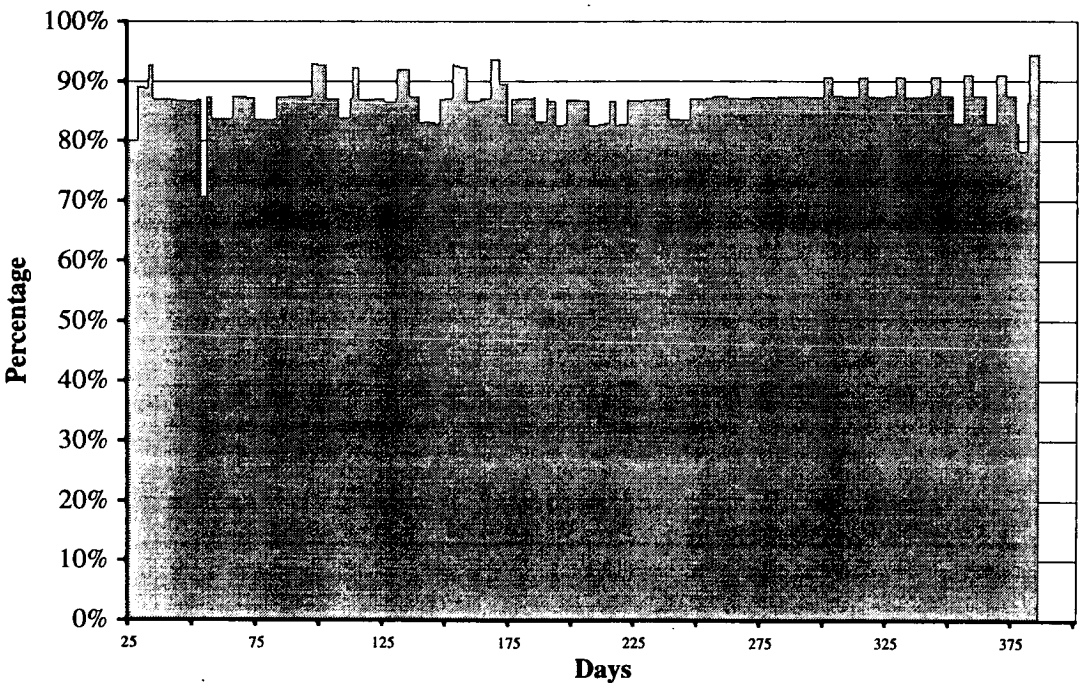
Case 1 cost optimal solution.



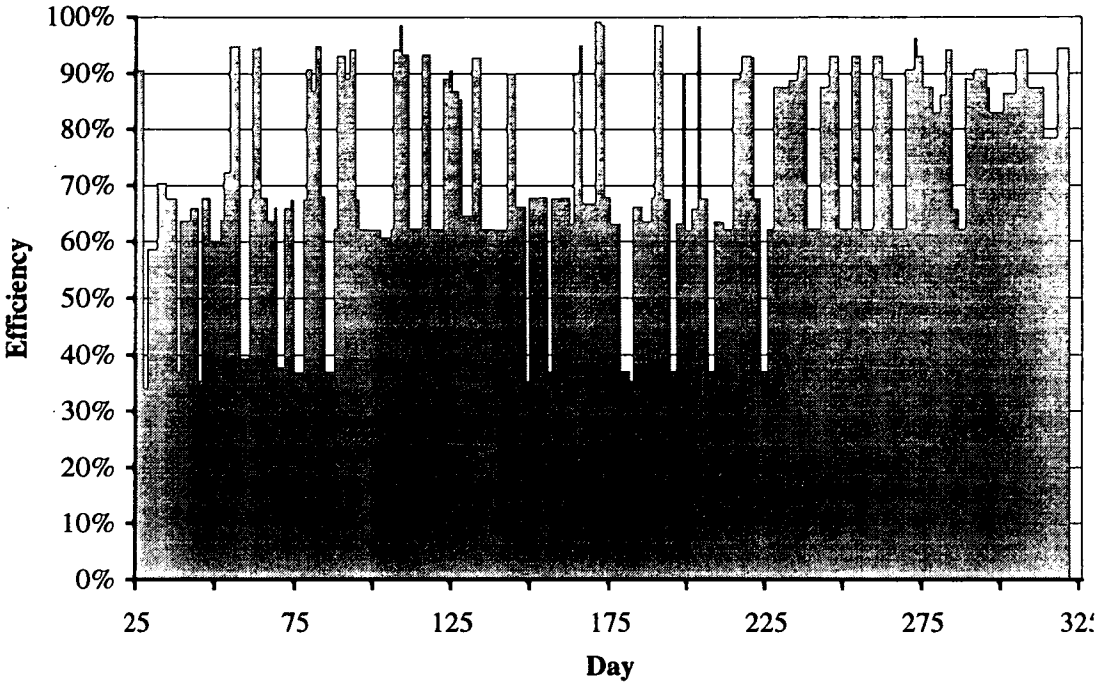
Case 1 time optimal solution.



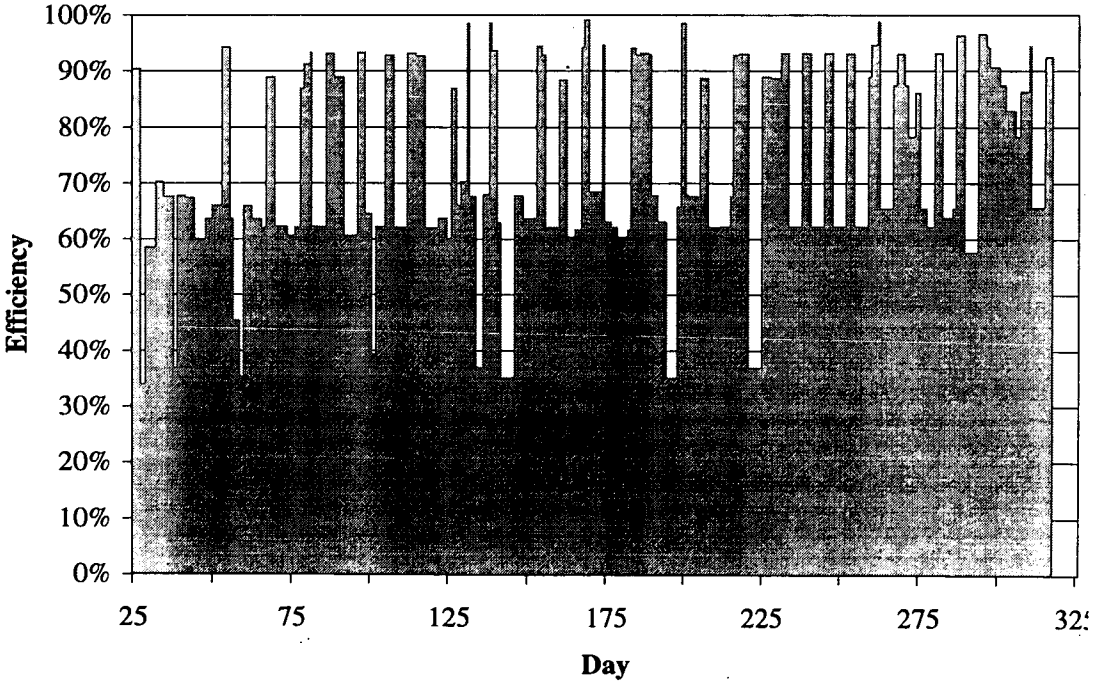
Case 2 cost optimal solution.



Case 2 time optimal solution.



Case 3 cost optimal solution.



Case 3 time optimal solution

Appendix V Previously Published Papers

The following pages contain the three papers currently published from this research.

A Global Approach to Project Optimisation

A M Harding, University of Edinburgh, Scotland
D A Ponniah, University of Edinburgh, Scotland

ABSTRACT

The evaluation and adoption of a major project is dependent upon the economic estimation and a cost - benefit analysis. Once adopted, the various stages in the life cycle of the project are compartmentalised and the 'best solution' through optimisation for each stage is identified. This paper addresses the problem that optimisation of individual aspects of a project do not necessarily constitute the optimisation of the whole project. The various stages in the project life cycle are given with important factors effecting the optimisation, and the parameters which have to be examined are described. Furthermore, a variety of optimising techniques are considered. This on-going research is currently at the stage of computerising the optimisation model.

1 INTRODUCTION

History is littered with major projects that promised much and delivered much but at many times the original forecast costs. The channel tunnel was completed, albeit late, but the costs of the project are far from being reconciled. It may be argued that such large projects which require a level of technological innovation and over large time spans would inevitably lead to overruns in time and cost. However the alternative view is that with rigorous planning and control during the project such deficiencies may be avoided or minimised. Basic to such a proper plan is optimisation which considers the numerous factors and uncertainties associated with such projects.

Optimisation is a means by which the numerous scenarios may be analysed with respect to a specific outcome, with the course of action based on the optimal solution. As the project progresses the optimisation would contain an element of completed work and yielding a more relevant optimal solution.

In recent years much attention has been given to the scope for optimisation within projects, particularly construction projects, in terms of cost, time and performance (or quality). Indeed, much research has already been done into optimisation of various phases of the project life cycle, particularly the design and implementation phases. However, optimisation tends to be focused on one phase of the project life. Furthermore, this optimisation consists of optimising many small systems within the project, such as the delivery schedule of a single resource, or the crew allocations to a small series of activities. However, optimising local systems individually will not necessarily provide an optimal solution for the project as a whole. In fact it almost certainly will not, because it ignores all the interrelationships between the different aspects of the project. Thus if a global optimum is to be found, some analysis of globally optimising the whole project life cycle is required.

2 THE PROJECT LIFE CYCLE

The project life cycle is often considered to be the phases through which a project passes between concept and completion (commissioning and handover). However, a common cause of conflicting interests within a project is the fact that the project manager is primarily concerned with the success of the project only up to what he sees as completion, and fails to understand fully the needs of the customer beyond this point.[5,8] The project as the customer sees it begins with a need, and ends with the decommissioning of the project at the end of its useful life. The aim of the project is not necessarily to ensure completion of construction and commissioning on time and on budget, but to ensure that the project succeeds in meeting the need of the customer in the most efficient way possible, and not necessarily only in terms of "On time and on budget". For this reason the project life cycle should be considered from concept to decommissioning, and any approach should be flexible to the needs of the customer. Figure 1 illustrates the various stages of the life cycle, which are briefly described below.

2.1 CONCEPT

In this phase possible solutions to the need are addressed. This phase may produce possible solutions to be passed on to the feasibility phase, or it may allow the need to be resolved in some alternative way.

2.2 FEASIBILITY

In this phase one or, more usually, several options are considered in more detail than the concept phase. They would usually come up with more accurate cost estimates, performance criteria and time factors for

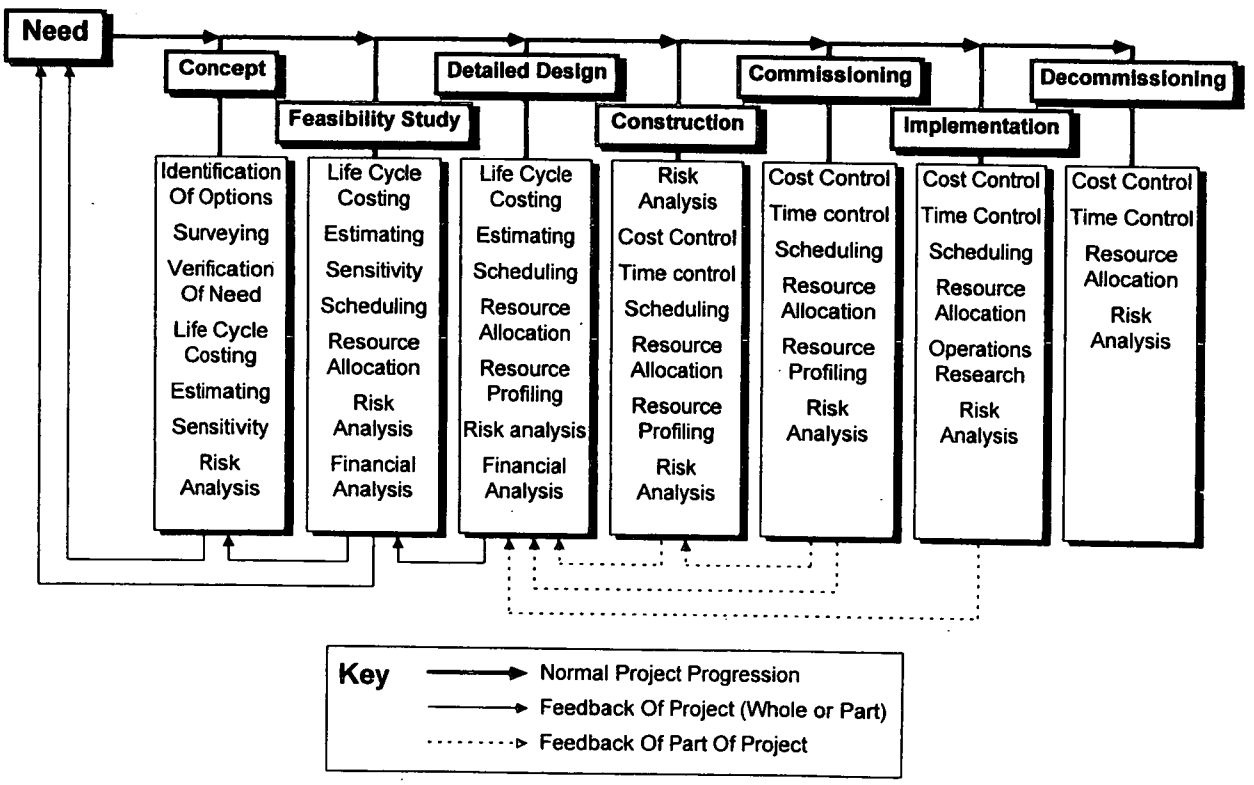


Figure 1: Analytical Techniques Within The Project Life Cycle

these options, as well as an approximate design. For this phase to pass into detailed design, it is usually necessary to select one option. This phase can also reveal the need for re-evaluation at the concept stage, or reveal an alternative means for resolution of the need.

2.3 DETAILED DESIGN

As well as producing a detailed design of the solution, this phase is also concerned with planning, including contractual arrangements, project scheduling and procurement of materials. If the project is acceptable, then it will progress to the construction phase. It is very uncommon and undesirable for a project to be reconsidered at this stage of the project, as considerable resources will have already been committed. This is particularly so in the later stages of this phase. However, it is theoretically possible to pass the project back to the feasibility stage, and re-evaluate the project.

2.4 CONSTRUCTION

When the project enters the construction phase, feedback of the whole project to previous stages becomes impossible, because part of the project already exists physically. However, feedback of parts of the project to the design phase may prove necessary if assumptions in the design are revealed on construction, and these need to be reconsidered.

2.5 COMMISSIONING

This stage involves the inspection and testing of all aspects of the project to ensure that it is able to meet the customer's need. This may reveal that parts of the project have not been designed or constructed to the required specification. These may have to be redesigned or reconstructed.

2.6 IMPLEMENTATION

This is usually the longest and most costly phase of the project, and involves the operation of the project. This is the phase of the project which will demonstrate that the 'need' has been met. Again, it is possible for portions of the project to be fed back to the design phase, as elements of the solution may become inadequate over time. However, unlike the construction and commissioning phases, this is not necessarily to do with errors in the design or construction. The implementation phase is generally much more subject to a changing environment than any other phase, particularly to advancing technology. Thus partial redesign is often inevitable in this stage.

2.7 DECOMMISSIONING

This phase takes place when the solution has reached the end of its functional life, and involves the freeing up of resources taken up by it.

At each of these stages key decisions are made which influence subsequent decisions made later in the project. Each of the phases of the project has a number of analytical techniques which are applied to the existing project information for analysis of the project. This is also shown in Figure 1, as well as where part or all of the project can be passed back for revaluation at an earlier phase.

At the start of the project the scope for optimisation is very large. There are many possible solutions to the problem. As the project progresses beyond feasibility, one particular solution is opted for, and so scope for optimisation is restricted to fit within that solution. A great deal of optimisation is then performed in the detailed design phase, and continues to be performed throughout the project life cycle. However, as each stage progresses, the amount of optimisation that can still be performed is reduced, due to restrictions imposed in previous phases. By the time the decommissioning stage is reached, there will be very little further optimisation that can be performed. This is shown in Figure 2. Therefore at each stage, or perhaps even repeatedly within each stage, it is necessary to perform a global optimisation of the remainder of the project, given its existing status and available information.

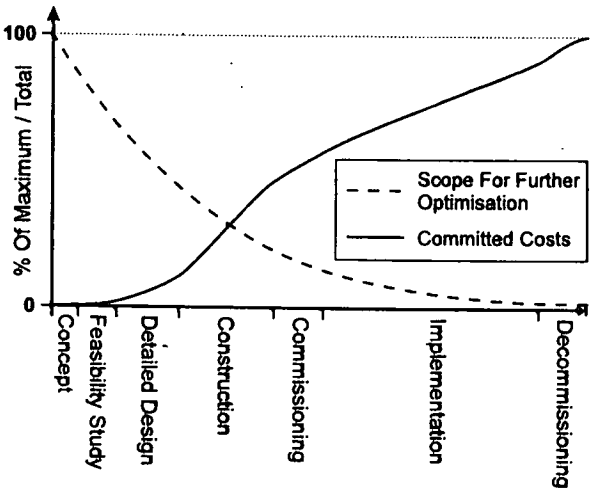


Figure 2: Committed Costs And Scope For Further Optimisation In The Project Life Cycle

3 THE PROJECT MODEL USED FOR OPTIMISATION

In order to perform any optimisation of a project, it is necessary to model it in a way which is accurate and consistent with the level of detail of the information available within the project. Thus there are three basic conditions which the model must meet to be effectively used in the optimisation of a project.

1. Representation. At any stage the model must be capable of representing a project as a whole, from concept to decommissioning.
2. Increasing Detail. As the project passes from stage to stage, the information available becomes increasingly detailed. Thus the model must be able to incorporate and accurately model these increasing levels of detail.
3. Numerical Input And Output. For an optimisation, precise numerical input and output is required. Therefore, the model needs to be numerically based.

A precedence network is a simulation model of a construction project. The model is first created by breaking the project down into activities. These activities are then assigned values to represent their behaviour in the real world as well as they can be modelled using the information available at that stage of the project. These values may be items as simple as duration, cost and the activities which must be completed. This is a very simple model, and is the elementary form of an optimisation. With increasing complexity other network input characteristics can be taken into account. This would include items such as resource demands, productivity, crew allocations and resource allocations for each activity, as well as crew and resource availability, cash flow and other considerations for the project as a whole.

This would allow representation of the project at all stages, as well as supporting increasing detail. The precedence network is an entirely

numerical model, so it is consistent with the third condition as well as the first two.

3.1 TERMINOLOGY

Within the considered model terminology has been developed to define the information used and is shown in Figure 3.

1. **Network Input Characteristics.** Network input characteristics are all the characteristics used to evaluate the network: characteristics of activities, resource availability and costing etc.
2. **Network Input Attributes.** While there are many network input characteristics, many of these will be predetermined, either because they are fixed inherently, probability distributions, or a function of other characteristics. However, with many network input characteristics it would be possible to use one of several values. Generally these values would be determined using engineering judgement, or at best using some sort of local optimisation.
3. **Network Output Characteristics.** The network output characteristics are the various values calculated by the scheduling logic from the network input characteristics. This includes things like the start times of all activities, resource usage at any time period, and bank balance at any time period.
4. **Network Output Attributes.** The network output attributes are the most important network output characteristics of the project, which will be monitored as part of the optimisation. The actual network output characteristics used as network output attributes will vary from project to project.
5. **Solution Set.** A solution set for a network is a set of network input attributes, and the corresponding network output attributes.

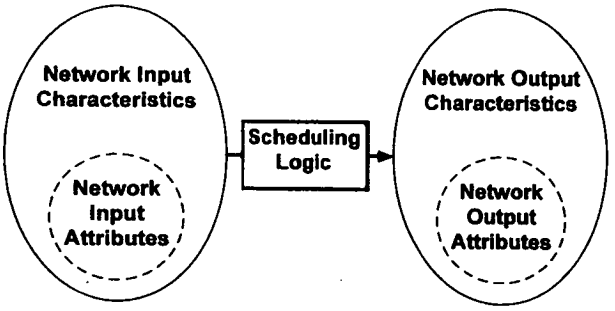


Figure 3: Basic Format Of The Model

3.2 NETWORK INPUT CHARACTERISTICS

Network input characteristics constitute all the information used to schedule the network. This includes both numerical values, such as project duration, and the logical constraints imposed on activities dictating when they can begin or finish relative to other activities. Network input characteristics must be precisely expressed. This rules out the inclusion of aspects which are essentially qualitative. Attempts have been made in the past to try and quantify such aspects numerically and use them within the analysis of the project. It would probably be possible to add these to the model without any difficulty, although their validity for optimisation is uncertain.

This is because any numerical approach to representing this kind of information would only be an abstract and approximate representation of reality. It is impossible to put a precise numerical value on engineering, aesthetic, or any other type of qualitative judgement, because they are very subjective. While useful for simple methods which can help the planner establish their priorities, numerical expressions of qualitative aspects are inappropriate for use within a model which, though not necessarily very accurate, is very precise.

3.3 RELATIONSHIPS WITHIN THE MODEL

Although network input characteristics are often considered as being independent, in reality many are related to each other in such a way that one directly influences another. The main internal relationships are discussed in this section.

3.3.1 RESOURCE ALLOCATION

The allocation of resources to an activity can affect other aspects of the activity. The productivity of the crew is dependent upon the manpower allocated to it. Productivity can also depend on things like the allocation of plant and the supply of other resources such as water or electricity.

Although activity duration can be influenced by resource allocation, this relationship can operate in two directions. The first direction is the effect that the allocation of a resource of a certain level or the crew productivity affects activity duration. However, to complete the activity within a certain time, it may be necessary to make certain demands on resources. Thus the activity duration can also dictate resource requirements.

The simplest way to incorporate this into the model is for activity duration to be dependent upon only crew size and productivity, with all other resource demands being dependent upon duration.

3.3.2 CREW CONTINUITY

Some groups of activity involve the same basic elements. For instance, casting a floor is essentially the same on the first floor of a building as the subsequent storeys. These activities are known as repeated activities. If the crew is maintained from one repeated activity to the next, then an increased productivity is observed on all but the first repeated activity.[10,16]

3.3.3 CREW PRODUCTIVITY

The crew productivity is the average productivity of one crew member within the activity. It has a direct effect on project duration, which is directly proportional to the multiple of the crew size and productivity.

3.3.4 SOURCE OF FUNDING AND REVENUE

The source of funding plays a large part in determining the amount of money a project has. The time at which money becomes available, the time at which it should be repaid and interest rates relate directly to the bank balance.[3] Revenue is often also earned within a project. This also contributes directly to the bank balance of the project.

3.4 SCHEDULING LOGIC

The scheduling logic is responsible for scheduling activities at certain times, allocating resources, ensuring that resource demand does not exceed any limits imposed on the project, monitoring bank balance, et cetera.

Network logic is simple and easy to implement, and many aspects of the project, such as bank balance, involve simply adding up numerical values for each time period. There are two types of resource, which need to be considered in slightly different ways.

1. Sustained Resources. These resources remain at the level of supply allocated to the project for each particular period. Examples include manpower, plant usage, and water supply.

2. Non Sustained Resources. These resources, once used by an activity, are no longer available. Examples of this type of resource are materials and money.

For sustained resources, it is necessary for the model to ensure that the resource supply for each time period is not exceeded by resource demand. For non sustained resources, demand should not exceed what is currently available. This is done by a scheduling logic, which prioritises some activities over others in a resource constrained situation.

The simplest way of prioritising activities is to have activity priority numbers associated with each activity. If an activity has a higher priority number, then the model will select it for starting at the time period under consideration over and above activities with lower priority numbers. Usually activities close to the critical path and large or expensive activities are given high priority numbers. The range of priority numbers is fairly arbitrary, but it is usually between 1 and 100.

3.5 NETWORK INPUT ATTRIBUTES

A network input attribute is any independently variable network input characteristic. Network input characteristics which are not network input attributes are either a fixed value, variable dependent upon some other network input characteristic, or a probability distribution. There are many network input attributes which might be considered within an optimisation. This section provides a full list of all the possible network input attributes considered so far. These are network input characteristics which may be independently variable in some projects, although not necessarily all.

3.5.1 RESOURCE CEILING

In any real project most of the resources, if not all, are limited in some way. The resource ceiling is the amount of sustained resource consumption which cannot be exceeded in any time period. The ceiling may differ from time period to period, but it is fixed for any one period regardless of how much of the resource was used in previous periods. Sometimes this is fixed by the nature of the project. However, sometimes it is possible to decide upon the level while the phase of the project is being planned, as in the case of manpower allocation. The best resource ceiling to apply could form a network input attribute or series of network input attributes covering different time periods.

3.5.2 RESOURCE STOCK SIZE

Resource stock size is for the consideration of resources which are not sustainable. It differs from the resource ceiling in that rather than having a fixed usage for each period, it has a fixed stock size which is reduced each time an activity uses the resource. The stock which is not used can also contribute to the usage of space in the model. In a fairly simple model, it would be adequate to express this in terms of total stock required. However, as the level of detail increases, the stock size needs to be broken down into a resource supply schedule, with resource supply rate considerations if desired.

The resource supply schedule specifies when stock is delivered. When this is employed, the resource stock size ceases to be a network input attribute. This is because at any time period the stock size is dependent upon two factors: the amount of stock which has been delivered and the amount which has been used.

The optimisation should be concerned with both how much of a resource is required, and how long the resources should be delivered before they are required. This becomes particularly important when the risks of late delivery and defective material is considered within the model. Perhaps the most simple way to incorporate this would be to have each day's resource supply as a network input attribute. More complex methods may consider numbers and timing of deliveries.

For any resource, there will be an upper limit on the supply rate. Sometimes this is simply fixed, but often it is possible to gain a trade-off between cost and supply rate.[1] The most simple means of expressing this is as a constant rate available throughout the project, which would constitute one network input attribute. At an increased level of detail, the supply rate becomes more difficult to define, because it is not necessarily constant throughout the project. An optimisation at this level of detail would need to determine when and by how much the rate should change.

3.5.3 RESOURCE ALLOCATION

Resource allocation considers how much of a resource is allocated to an activity while it progresses. For instance, casting a floor slab would require a certain amount of concrete: a fixed resource requirement. However, if the plant allocated to the activity was only capable of mixing it at a certain rate, then the duration of the activity would be limited by this value. If the amount of plant allocated to the activity were not fixed, then it would be a network input attribute.

3.5.4 SOURCES OF INCOME

In some projects, choices of where to obtain funding for a project exist. In some cases the project may simply be allocated some money by a governmental organisation or independent body. However, it may be necessary to provide some or all of the funding for the project by borrowing money or obtaining it by some other means. How much money to obtain from each possible source (between 0% and 100% of the project cost), and when to obtain it could form part of the optimisation.

3.6 NETWORK OUTPUT ATTRIBUTES.

The network input attributes are very general, and could take many forms, depending upon the project aims and the level of detail and of the model. They should be carefully defined so that the model can accurately optimise in terms of them.

3.6.1 COST

On the face of it, cost is a simple value. However, in reality the cost can be considered in several different ways. These methods of costing obtain different figures for the project which are all valid. It is important that the method chosen is consistent with the aims of the project.

The methods of observing cost include simple calculation of the total cost, net present value or internal rate of return. If the model is monitoring cash flow, then the bank balance at a milestone, minimum bank balance or mean minimim balance may be considered.

3.6.2 TIME

Time may also be considered as a network output attribute which is simple to evaluate, if one simply takes the overall project duration. However, if a project (including the implementation phase) has fixed duration, then it would appear that it is impossible to vary the time. However, rather than considering overall project duration, it may be better to consider certain milestones, such as the beginning of the implementation phase or the break even point. One or a combination of such milestones could be considered, provided a combination could yield some sort of acceptability indicator for time. The choice of how to monitor time should be carefully considered individually for any project.

3.6.3 RESOURCES

Within a project it may be desirable to carefully monitor and control a resource beyond the level of maintaining a resource profile which is advantageous in terms of cost, time or risk. Resource usage, stock size, et cetera can be specified as a network output attribute if required.

3.6.4 PERFORMANCE

Performance, which is also often referred to as quality, within a project is very difficult to quantify, as it is such a broad category. Performance includes factors like aesthetics, operability, efficiency, safety to the public, and many more. The relative importance and validity of these factors depends very much upon the problem/need which the project is aimed to solve/meet. Performance is often the most important output from the project.

Because performance is so difficult to represent precisely in a numerical way, it may be better to compare the network output attributes of two optimisations at two levels of quality, and then use professional judgement to discern which is most consistent with the project aims.

3.6.5 RISK

Within the project some risks are qualitative, some quantitative. As with performance, to try and quantify qualitative risks would be inappropriate. Therefore it is better to use existing techniques of qualitative analysis separately, and only incorporate quantitative aspects of risk into the model.

Many network input characteristics have risk elements associated with them, which means that they are not truly explicit values, but probability distributions. Evaluating risk in a project usually means obtaining a mean and standard deviation for various network output characteristics.

The main approach for quantifying risk within precedence networks is the Monte Carlo simulation method. For a Monte Carlo Simulation (MCS), the probability distributions are used to generate random numbers which act as fixed network input characteristics for the project, and the model is evaluated using these values. This process of randomising and re-evaluating the model is repeated many times (usually around 10,000 for an accurate analysis).[2] The results are collated, and this gives a mean and standard deviation for the project. If a more complex logging system is used, a probability distribution profile can also be found.

For an optimisation, we are interested gaining means and standard deviations for all network output attributes. If the model incorporates risk, then the expected cost of the project (however it is expressed) would be the mean cost as evaluated by a Monte Carlo simulation. Similarly, time and resources would be mean time and mean resources. Therefore it would be reasonable to use the mean values for each network output attribute (excluding risk). However, it may also be possible to use the confidence level. The *n*% confidence level is the value which one is *n*% confident that the value will not be worse, and can be evaluated statistically using the mean and standard deviation of a network output attribute. The problem with this is that it incorporates a value of risk into other network output attributes, and control over risk and other network output characteristics is reduced because any value of a confidence level could be composed of one of many combinations of the two.

To express the risk on its own as a network output attribute it is necessary to take the standard deviation. In a project there may be several risk network output attributes, one for cost, one for time, et cetera.

4 THE OPTIMISING MODEL

In order to perform optimisation of a system, the system must b represented in a particular way. This representation has two parts. Th first is the domain, or feasible region. The domain is the set of all th possible combinations of values for the independent variabl parameters which define the system. This is a subset of n dimensiona space, where n is the number of the independent parameters, and ma be an infinite set. The second component of the system is the objectiv function. The objective function maps from the domain to a single on dimensional value.

In a network the function is the scheduling logic which calculates th network, giving costs, times, resource demands, et cetera. Thes network output attributes should be figures which the optimiser is abl to minimise or maximise. There may be several network outpu attributes defined for any particular project. However, the functio used for optimisation may only map to a one dimensional value. Fo optimisation to be only in terms of a one dimensional value it i necessary that the optimisation can only be performed in terms of on network output attribute at a time. This attribute should be eithe minimised or maximised. Into this optimisation it would be possible t incorporate other network output attributes, but only as constraints This would involve setting maximum or minimum values for each o these attributes. Any set of network input attributes whose networ output attributes do not meet the requirements of these constraint would no longer be a part of the feasible region.

The domain is an n dimensional set, where n is the number of networ input attributes. The constraints upon this set are defined by th constraints of each individual network input attribute, as well as th constraints of the network output attributes which are not bein optimised. Each network input attribute represents some network inpu characteristic in the project, and the optimising model will vary it. Al network input attributes are represented in the same way. They have maximum value, a minimum value and a step. The step is the numbe by which any value of that network input attribute is divisible. If it i zero then the network input attribute is continuous between th minimum and maximum. The step is for things like resources whic are only available in batches of a fixed size, and crew sizes (which ma only be a whole number). The representation of network inpu attributes in a common form means that the optimiser can optimise al attributes together using a logic common to all network inpu attributes.

The optimiser is capable of manipulating the network input attribute within their constraints. Once the network has been rescheduled usin the current values of the network input attributes the optimiser ma read the network output attributes. If the network output attributes d not meet the requirements made by their constraints then the optimise rejects the solution set as unfeasible. Depending on the values of th network output attributes more rescheduling may be required by th optimiser based on further changes to the network input attributes. Th optimiser would also need to maintain one or more solution sets for th comparisons which will be necessary within the optimisatio algorithms.

5 OPTIMISING ALGORITHMS

Perhaps the greatest difficulty with optimising the network is that th behaviour of the optimising surface is unknown. How certain change in network input attribute will affect the output is not known until th scheduling logic is run, and cannot easily be predicted, if at all, in an other way. How many local optima exist within the feasible region i not known. The sensitivity of the global optimum is not known

Because of these problems many widely used methods of optimisation were rejected. Function based methods like the simplex method, and gradient methods could not be applied because there are discontinuities in the optimising surface. Dynamic programming was rejected because the feasible region would be infinite (it would have an infinite number of members), and even if a finite approximation were used the number of recalculations of the network required would be prohibitively large.

Other techniques have been developed specifically for networks. However, they are only concerned with optimisation of specific aspects of the project, rather than the project as a whole. Optimising each of these aspects individually will not necessarily find the global optimum. The following two search techniques were selected as possible methods of optimisation which could be applied.

5.1 MONTE CARLO SIMULATION

Monte Carlo simulation has already been extensively used in project networks to simulate risk. However, this random input generating method can also be used in optimisation, and has been in other fields. The network is simulated (recalculated) many times over using random values for each network input attribute under consideration, and the solutions logged. Only the best solution, or at most the best few solutions, should be logged rather than all solutions. Otherwise the amount of data to be stored could be extremely high. For a large number of simulations this would be expected to yield a solution very close to the true global optimum.

However, the number of calculations required to be confident that all the network input attributes are at or very near the global optimum is very large. This is particularly true where there are a large number of network input attributes. Although the solution is nearly at the optimum, there is a good chance that some of the network input attributes are far from their optimum value.

A further problem with this method is that if the optimum is very sensitive, then it may select a result close to a local optimum which is not the global optimum because it is better than the solution near the true optimum.

5.2 ONE AT A TIME METHOD

This method of optimisation involves the fixing of all network input attributes but one. Then the value of that attribute which yields the best overall result for that network input attribute is found. That network input attribute is then fixed, and the algorithm moves on to the next network input attribute and varies that one in the same way. This, when repeated for all network input attributes under consideration, constitutes one iteration. Several iterations should yield a highly optimal result.

In its most simple form this technique involves optimising one network input attribute at a time in terms of global output attributes, and repeating this until the solution is sufficiently accurate. In a more complex arrangement it may be possible to optimise several at a time using existing local optimising techniques. However, any technique used in this way should be able to optimise in terms of network output attributes rather than local characteristics.

The one at a time method will not, however, necessarily find the global optimum. It may only find the local optimum. This would particularly be a problem if there are many local optima, when the chances of the technique starting sufficiently close to the global optimum would be small.

5.3 COMBINING THE METHODS

The one at a time method needs to start sufficiently near the global optimum to find it, rather than another local optimum. This near optimal starting point can be provided by the Monte Carlo method. Uncertainties as to which solution found is closest to the global optima generated by the Monte Carlo method can be resolved by performing one at a time optimisation of the best few solutions found rather than just the one best solution.

6 FURTHER WORK

The suggested model is currently being coded into a computer. Once created, its validity will be tested by analysing its ability to find optima within local optimisation problems analysed by existing methods and previous case studies. Once this has been done, and any refinements required are made to the model, it will be used to analyse a large project within the oil and gas extraction industry. The optimum found will be compared to the solution used, and other data, such as the sensitivity of the optimum will be found. Thus it will be possible to refine the method itself to see how the method could search for the optimum most efficiently.

7 CONCLUSION

Existing work on the optimisation of projects has been based on using many different locally optimising techniques separately to arrive at a solution, and has often been focused only on certain parts of the project rather than the project as a whole. However, this will not necessarily find the true optimum for the project. This paper proposes a method of using the precedence network to represent the whole project. Then, after identifying which elements of the project are available for optimisation and what to optimise in terms of, the project may be globally optimised using the techniques outlined.

References

1. Arditti, D - Albulak, M. Z.; Line-Of-Balance Scheduling In Pavement Construction; American Society Of Civil Engineers, Journal of Construction Engineering and Management , vol. 112, 1986, pp411-424
2. Diaz, C. F. - Hadapriono, F. C.; Nondeterministic Networking Methods; American Society Of Civil Engineers, Journal of Construction Engineering and Management, vol. 119, 1993, pp40-57
3. Easa, S. M.; Optimum Cash-flow Scheduling Of Construction Projects; Gordon And Breach Science Publishers, Civil Engineering Systems, vol. 9, 1992, pp69-85
4. Elms, D. G.; Network Structuring Algorithms; Kluwer Academic Publishers, Optimisation And Artificial Intelligence In Civil And Structural Engineering, vol. 1, 1992, pp43-60
5. Flanagan, R. - Norman, G. et al.; Life Cycle Costing: Theory And Practice; BSP Professional Books, 1st edition 1989
6. Foulds, L. R.; Optimization Techniques; Springer Verlag, New York, 1st edition, 1981
7. Guang-Yuan, W. - Dong-Yad, T.; Global Optimisation Of Large Scale Engineering Systems; Gordon & Breach Science Publishers, Engineering Optimization (18), 1992
8. Holmes, R.; Optimising Performance And Quality In Construction; Chartered Institute Of Building, 1989

9. Humphreys, K.; Jelen's Cost And Optimisation Engineering; McGraw-Hill, 3rd edition, 1991

10. Kavanagh, C. P.; SIREN: A Repetitive Construction Simulation Model; American Society Of Civil Engineers, Journal of Construction Engineering and Management, vol. 111, 1985, pp308-323

11. Kerzner, H.; Project Management: A Systems Approach to Planning, Scheduling And Controlling; International Thompson Publishing, 5th edition, 1995

12. Lockyer, K. - Gordon, J.; Critical Path Analysis And Other Project Network Techniques, 5th edition, Pitman, 1991

13. McGartland, M. R. - Hendrickson, C. T.; Expert Systems For Construction Project Monitoring; American Society Of Civil Engineers, Journal of Construction Engineering and Management, 111, 1985, pp293-307

14. Mirham, G. A.; Simulation: Statistical Foundations and Methodology; Academic Press Inc., 1st edition, 1972

15. Moselhi, O. - Lorterapong, P.; Near Optimal Solutions For Resource Constrained Scheduling Problems; E. &F. N. Spon, Construction Management And Economics, vol. 11, 1993, pp293-303

16. Mosheli, O. - El-Rayes, K.; Scheduling Of Repetitive Projects With Cost Optimization; American Society Of Civil Engineers, Journal of Construction Engineering and Management, vol. 119, 1993, pp681-697

17. Robinson, D. R.; A Dynamic Programming Solution To Cost Time Trade-off For CPM; Providence, R. I., Management Science, vol. 22, 1975, pp158-166

18. Russell, D. L. Russell; Optimization Theory, W. A. Benjamin Inc., 1st edition, 1970

19. Thabet, W. Y. - Beliveau, Y. J.; Modeling Work Space To Schedule Repetitive Floors In Multi-storey Buildings; American Society Of Civil Engineers, Journal of Construction Engineering and Management, vol. 120, 1994, pp96-116

20. Ward, S. C. - Chapman, C. B.; Risk Management Perspective On The Project Life Cycle, International Journal Of Project Management, vol. 13,1995

Society of Petroleum Engineers

SPE 38488

Experience With a Global Optimisation Approach to Project Scheduling and Resource Allocation

A.M. Harding, University of Edinburgh, and D.A. Ponniah, University of Edinburgh

This paper was prepared for presentation at the 1997 Offshore Europe Conference held in Aberdeen, Scotland, 9–12 September 1997.

Abstract

In recent years there has been a growing trend in major projects towards considering the whole project life cycle and its costs. This analysis initially considered concept to commissioning, and later, particularly after Brent Spar, concept to decommissioning. Within this project life cycle, optimal solutions are sought. However, the field of optimisation still tends to focus on improving the performance of elements of the project or small subsystems within the project as a whole. An optimisation across the whole project life cycle is not normally carried out. This paper proposes a general system for such a global optimisation model, which will allow the whole project to be considered within a single optimisation. The strategy consists of a basic scheme for input to an objective function and the definition of its constraints. The objective function can map to a value relating to cost, time, performance or risk, allowing the most important criteria to be maximised or minimised. Other values not mapped to by the objective function can be used as constraints to restrict any detrimental impact to them during the optimisation. The size and complexity of a true globally optimising model is extremely large. If the model is to be able to optimise globally, then it must also be able to arrive at an optimal solution for a local optimisation problem. Therefore its suitability for optimisation is tested on project scheduling and resource allocation, a subsystem of the project. A new approach to modelling project schedule and resource allocation, which will allow the model to fit into the globally optimising model, is developed. Using examples from the literature, the performance of this model on specific locally optimising problems is investigated, using Monte Carlo random search. This, although not normally considered as being an efficient means of finding an optimum, was considered the best approach to initial investigation where the nature of the optimising surface is unknown. In particular, the model is used to solve resource constrained scheduling problems. The difficulty of finding the optimum within the model is also studied, and extensions to the model for further research are proposed.

Introduction

Project optimisation has become an important field of study in recent years, fuelled by a desire to complete projects more quickly, for less cost and to a higher quality. Many of the techniques arising from this research have been simple mathematical optimisations, aimed at solving specific problems within specific stages of projects. However, the majority of the techniques developed by this research have not been applied to real problems. This is principally for two reasons.

1. The optimisations often take the form of models which consider only a few aspects of the project, and the true mathematical optimum for the model is often an unfeasible solution for the real project it is set up to simulate[1,2].

2. The techniques can often be very demanding on computational resources, particularly for very large projects. Any savings they might make are offset by the cost of computing them, if it is even possible to compute them within a reasonable timescale[3].

Given these criticisms, it is clear that any optimisation, if it is to be applicable to real problems, must use a sufficiently accurate model of the project to ensure that solutions considered are realistic, and find either an optimal solution or an acceptable near optimal solution with a reasonable amount of computational effort. This is the aim of the globally optimising model which is proposed in this paper.

Beyond this, the globally optimising model has been implemented for the solution of the resource constrained project scheduling problem. This was done to demonstrate the advantage of considering more than one type of variable within a simple project analysis problem.

odelling the Project Life Cycle

e project life cycle is often considered to be the phases ough which a project passes between concept and mpletion (commissioning and handover). However, a mmon cause of conflicting interests within a project is the ct that the project manager is primarily concerned with the ccess of the project only up to what he sees as completion, d fails to understand fully the needs of the customer beyond s point. The project as the customer sees it begins with a ed, and ends with the decommissioning of the project at the d of its useful life[4].

The aim of a project is not necessarily to ensure mpletion of construction and commissioning on time and on dget, as it is so often considered today[5]. The true aim ould be to ensure that the project succeeds in meeting the ed of the customer in the most efficient way possible, and t necessarily only in terms of "On time and on budget". For is reason any consistent model of the project life cycle ould be considered from concept to decommissioning, and y approach to optimisation of it should be sufficiently xible to be sensitive to the needs of the customer. Figure 1 ustrates the various stages of the life cycle, along with the alytical techniques which are usually used during these ases. These stages are briefly described below.

oncept. In this phase possible solutions to the need are ldressed. This phase may produce possible solutions to be assed on to the feasibility phase, or it may allow the need to e resolved in some alternative way.

easibility. In this phase one or, more usually, several ptions are considered in more detail than the concept phase. hey would usually come up with more accurate cost stimates, performance criteria and time factors for these ptions, as well as an approximate design. For this phase to ass into detailed design, it is usually necessary to select one ption. This phase can also reveal the need for re-evaluation t the concept stage, or reveal an alternative means for esolution of the need.

Detailed Design. As well as producing a detailed design of ne solution, this phase is also concerned with planning, ncluding contractual arrangements, project scheduling and rocurement of materials. If the project is acceptable, then it vill progress to the construction phase. It is very uncommon nd undesirable for a project to be reconsidered at this stage of he project, as considerable resources will have already been ommitted. This is particularly so in the later stages of this hase. However, it is theoretically possible to pass the project ack to the feasibility stage, and re-evaluate the project.

Construction. When the project enters the construction hase, feedback of the whole project to previous stages becomes impossible, because part of the project already exists physically. However, feedback of parts of the project to the design phase may prove necessary if mistakes in the design are revealed on construction, and these need to be amended.

Commissioning. This stage involves the inspection and testing of all aspects of the project to ensure that it is able to meet the customer's need. This may reveal that parts of the project have not been designed or constructed to the required specification. These may have to be redesigned or reconstructed.

Implementation. This is usually the longest and most costly phase of the project, and involves the operation of the project. This is the phase of the project which will demonstrate that the 'need' has been met. Again, it is possible for parts of the project to be fed back to the design phase, as elements of the solution may become inadequate over time. However, unlike the construction and commissioning phases, this is not necessarily to do with errors in the design or construction. The implementation phase is generally much more subject to a changing environment than any other phase, particularly to advancing technology. Thus partial redesign is often inevitable in this stage.

Decommissioning. This phase takes place when the solution has reached the end of its functional life, and involves the freeing up of resources taken up by it.

At each of these stages key decisions are made which restrict the scope of decisions made later in the project. Each of the phases of the project has a number of analytical techniques which are applied to the existing project information to produce data to help the making of this decision. These techniques are shown in Figure 1, as well as where part or all of the project can be passed back for revaluation at an earlier phase.

At the start of the project the scope for optimisation is very large. There are many possible solutions to the problem. As the project progresses beyond feasibility, one particular solution is opted for, and so scope for optimisation is restricted to what is available within that solution. A great deal of optimisation is then performed in the detailed design phase, and continues to be performed throughout the project life cycle. However, as each stage progresses, the amount of optimisation that can still be performed is reduced, due to restrictions imposed in previous phases. By the time the decommissioning stage is reached, there will be comparatively little further optimisation which can be performed. Therefore at each stage, or perhaps even repeatedly within each stage, it is necessary to perform a global optimisation of the remainder of the project to help determine how the project would best continue. This should be performed using the most up to date information both on how the project has progressed so far, and how it would be expected to progress in the future.

obal Project Optimisation

order to perform any optimisation of a project, it is cessary to model it in a way which is both accurate and nsistent with the level of detail of the information available thin the project. Thus there are three basic conditions ich the model must meet to be effectively used in the timisation of a project.

1. Representation. At any stage the model must be able of representing a project as a whole, from concept to commissioning.

2. Increasing Detail. As the project passes from stage to ge, the information available becomes increasingly detailed. us the model must be able to incorporate and accurately del these increasing levels of detail.

3. Numerical Input And Output. For an optimisation, ecise numerical input and output is required. Therefore, the del needs to be numerically based.

A precedence network is a simulation model of a nstruction project. The model is first created by breaking e project down into activities. These activities are then signed values to represent their behaviour in the real world well as they can be modelled using the information ailable at that stage of the project. These values may be ems as simple as duration, cost and the activities which must completed before the activity can be scheduled. This is a ry simple model, and a critical path schedule is the ementary form of an optimisation. With increasing mplexity other network input characteristics can be taken to account. This would include items such as resource emands, productivity, crew allocations and resource locations for each activity, as well as crew and resource vailability, cash flow and other considerations for the project a whole. This would involve modelling the project as a ueuing system. In such a system activities would queue for esources once their precedence relationships have been lfilled. This is commonly held to be a more accurate model f a project than an ordinary precedence network[6,7].

Such a model would allow representation of the project at l stages, as well as supporting increasing detail. It would lso be an entirely numerical model, and so fulfils the three equirements.

erminology. Within the considered model terminology has een developed to define the information used within the odel.

1. *Network Input Characteristics.* Network input haracteristics are all the characteristics used to evaluate the etwork: characteristics of activities, resource availability and osting etc.

2. *Network Input Attributes.* While there are many network input characteristics, many of these will be predetermined, either because they are fixed inherently, defined by probability distributions, or a function of other characteristics. However, with many network input characteristics it would be possible to use one of several values. These values are described as network input attributes.

3. *Network Output Characteristics.* The network output characteristics are the various values calculated by the scheduling logic from the network input characteristics. This includes items such as the start times of all activities, resource usage at any time period, and bank balance at any time period.

4. *Network Output Attributes.* The network output attributes are the most important network output characteristics of the project, which will be monitored as part of the optimisation. They can impose criterion on how the project should behave, and can be used as the output of the objective function used in optimisation. The actual network output characteristics used as network output attributes will vary from project to project.

5. *Solution Set.* A solution set for a network is a set of network input attributes, and the corresponding network output attributes.

Execution and Optimisation of the Global Model. The global model operates by applying a scheduling logic to determine the network output characteristics using the network input characteristics. The complexity of the logic used to do this would depend upon the level of detail of the network input characteristics available. This constitutes one run of the scheduling logic, and will produce a single solution set.

Generally this solution is currently considered as being fixed, because the network input attributes are considered as being fixed. Their values are usually determined using engineering judgement and subsequently considered as being fixed, or, at best, using some sort of local optimisation.

When an optimisation is performed upon the project, the network input attributes are considered as being variable. An optimisation technique would first set the values of the network input attributes, before the scheduling logic is run to determine the network output characteristics. From these network output characteristics the network output attributes are extracted and a solution set obtained.

The constraints imposed on the network input attributes define the region which is initially considered by the optimising algorithm as being the feasible region. Further restrictions on the feasible region can be applied using network output attributes, which will be discussed later. Each network input attribute has a maximum and minimum value, as well as a base unit. Any value of a network input attribute should be a multiple of its base unit and lie between the maximum and minimum values.

The network output attributes can be used in two ways. Firstly, they can impose restraints on which solutions should be considered as feasible. These exist in addition to those which are imposed on the network input attributes. One

xample of the use of network output attribute constraints is vhere the overall construction time for a project is considered s being a network attribute. For this a maximum construction ime could be specified. If this time is exceeded, then the optimisation technique should consider that solution as nfeasible. This prevents the optimisation technique from inding a solution which is optimal to one aspect of a project, ut at the same time unacceptably detrimental to other aspects f the project. Imposing such restraints can be done with na ny network output attributes, or none.

The second way in which network output attributes may be considered is as a the output of the objective function for an optimisation. The optimisation technique would seek to either minimise or maximise this variable, depending upon the nature of the network output attribute. For example, it may be desirable to maximise the NPV of a project, or minimise the time taken to reach its break even point. One and only one network output attribute should be considered in this way in any single optimisation, because the output of an objective function for optimisation should always be a one dimensional value.

Optimising the model. Having defined a feasible region and output for the objective function, it is possible to perform an optimisation on the model.

Perhaps the greatest difficulty with optimising the model is that the behaviour of the optimising surface is unknown. How certain changes in network input attribute will affect the output is not known until the scheduling logic is run, and cannot easily be predicted, if at all, in any other way. How many local optima exist within the feasible region is not known. The sensitivity of the global optimum is also not known. Because of these problems many widely used methods of optimisation were rejected.

The solution of the model using rigorous mathematical methods (such as linear programming, integer programming or geometric programming) would be exceedingly difficult. The number of constraints and different inputs would undoubtedly give rise to enormous matrices having to be solved for the linear programming model, if it were possible to use this technique at all. The number of evaluations an integer or geometric programming method would require would be prohibitive.

The use of gradient methods was also rejected. The behaviour of the optimising surface is not known, and these methods tend to find the nearest optimum only. If there are many local optima a gradient method may not necessarily find a good optimum.

Random search techniques are proposed for optimisation of the model. They provide a simple means of searching the whole optimising surface for good solutions, and are not sensitive to any unusual or complex behaviour of the optimising surface. This type of technique only requires storage space for a small number of solutions. A random search could take the form of a simple Monte Carlo simulation, or more complex methods, such as genetic algorithms or simulated annealing, which tend to focus on areas where good solutions are being found.

These methods will often not find the true mathematical optimum of the model, but they should find good, near optimal solutions. There is often a trade-off between the amount of computational effort used and the quality of the solution found. Therefore the best solution with the available computational resources can be found and used, while many other methods will simply yield no solution without sufficient resources.

Incorporating the Resource Constrained Project Scheduling Problem into the Global Model

In order to test the validity of the model in a simple case, it was used to optimise the resource constrained scheduling problem. Although this is a simple and well documented problem which is subject to many of the criticisms raised about project models, it was used to attempt to demonstrate that there were clear advantages of considering more than one type of variable at the same time within a project optimisation.

The resource constrained scheduling problem attempts to find the minimum duration of a project given certain upper limits on resource usage. This model is usually considered as having several renewable resources. Renewable resources, like manpower or plant, can be reused in the next and subsequent time periods. This means that to solve the problem the overall resource use for all activities should not exceed the maximum allowable resource use (or ceiling) for that time period. This ceiling is usually considered as being constant for all time periods. The fixed ceiling problem with renewable resources is the type of problem considered here.

There are two approaches to solving this problem. The first is to delay certain activities in such a way as to yield a minimum duration without exceeding the resource demands. The second, sometimes called project compression, involves compressing activities in parts of the project where the resource usage is lower than the limit. In this paper, both of these methods are considered.

Activity Priority. The rearranging of activities is considered by allowing the scheduling logic to prioritise certain activities in different ways. Each activity is assigned a priority value. If, at any time period, more than one activity is available to be scheduled, they are sorted into descending order of priority. These activities are then scheduled one by one, and all activities which would cause resource demands to exceed the limit are not scheduled, but made available for scheduling at the next time period. The priority value is considered as a network input attribute.

This has been used before by Lee and Kim[8] to reduce project durations. In most cases this method should eventually yield an optimal solution.

Resource Use. In considering the resource usage of an activity, the activity is given a total resource requirement for each resource. For the project compression solution to the problem it is assumed that this total requirement remains constant irrespective of the allocation of resources per period. Therefore the relationship between the activity duration and the per period usage of a resource can be expressed by:

$$d = R / r \tag{1}$$

There are two ways of assigning an network input attribute to this. Firstly, it is possible to control the duration. This, for this problem, would appear to be the easiest way to control the resource allocation. However, if, as the detail and realism of the model is increased, the total resource requirement may change with changes in the allocation per period. to include some consideration of the efficiency of manpower. In this case using the duration may become quite cumbersome, because it is only a secondary characteristic, dependent upon the resource allocations. Therefore, a new method of controlling the resource allocations and durations is controlled is proposed.

Often, the most consequential resource allocation is of manpower. The efficiency of crews on site can vary with different factors, such as whether or not the crew has just come from another similar activity type, or the size of the crew. Also, the fact that a detailed model will consider activities as queuing for crews suggests that the size of crew may be the least flexible aspect of the resource allocation to an activity. Therefore, it is proposed that this be used as the network input attribute. Given that there may be cases where another type of resource may control the duration in a similar fashion, this network input attribute has been given the more general name *Primary Resource Use.*

The allocation of a primary resource is used to calculate a projected duration of the activity. From this it is then possible to calculate the activity's requirements of other resources per time period, by simply dividing the overall requirement by the projected duration. This is expressed mathematically as follows:

$$d = R_p / r_p$$
$$r_i = R_i / d \quad \forall i \in S \tag{2}$$

Analysis

Analysis was performed on the 110 problems assembled by Patterson[9] for the comparison of heuristic rules for near-optimal solution of resource constrained scheduling problems. These problems were used because the optimal solutions for the case of fixed activity duration was known for each problem, and could be used as a benchmark value for measuring the performance of each optimising technique.

The model was programmed into a computer using C++, and ran on a personal computer. A Monte Carlo random search procedure was created for the optimisation. The random selection of values for the network input attributes was unweighted, so the probability distribution across the whole field was even for each network input attribute. The random number generator used by the program was reseeded with a constant value at the start. This ensured that the results were repeatable, as the same sequence of random numbers would be generated if the same optimisation were performed.

Priority was assigned integer values between 1 and 100. Primary resource use was assigned a range between 1 and the greatest number permissible before the activity would demand more then the maximum availability of any resource it required. The optimisation procedure was limited to selecting integer values of primary resource use. If this lead to noninteger requirements of other resources, they were rounded up to the nearest integer. Similarly, if the allocation of a primary resource use lead to a noninteger activity duration, then the duration was rounded up to the nearest integer.

The network output attribute is the finish time for the last activity.

Results

A Monte Carlo random search was performed on the 110 Patterson problems, taking the best solution found. This search used 10000 schedules of the project using randomised network input attributes. First of all priority and resource use were considered on their own, and then in combination. In each run, an initial value was obtained from prioritising activities in order of appearance in the network, and the resource allocations specified for the project.

The optimisation's progress was then measured against target durations. These values were the optimal durations supplied with the problems, which are the minimum project durations for the network given no change in activity duration. The progress of each optimisation was measured as a proportion of the distance travelled from the initial duration to the target duration, with 1 representing no change from the initial duration, 0 representing the target duration and a negative value indicating a project duration less than the target duration. Where there was no difference between the initial and target durations, the mean difference between the two for all 110 problems was used to calculate the proportion change for the individual problem.

The results of these runs are summarised in Table 1. This shows several performance indicators for the methods. *Best solutions* shows for how many projects that method found the best duration. *Uniquely best solutions* indicates for how many of these best solutions the method was alone in finding this value, and *target value reached* shows how often the method achieved the target value. Mean performance indicators for 1000 and 10000 runs are also shown.

The priority method was more successful than the other methods at finding uniquely best solutions, and at reaching the

target duration. The combined method's performance in terms of average distance to target is superior to the other two methods for both 1000 and 10000 runs. It also succeeded in finding the best duration found for nearly 10% more runs. However, it was unique in achieving this duration in 18% of cases, rather than the 30% for which solution by the priority method was unique.

This apparent disparity can be explained by considering the standard deviation of the performance indicators for both 1000 and 10000 runs in the table. It is evident from this that the performance of both the resource use and combined methods varies considerably from problem to problem. Where their performance is low, priority is likely to be the only method which yields significant optimisation. This is because the large number of poor solutions from the resource use make it more difficult for the method to yield good solutions for the priority problem.

This can also be seen in the median values. Priority obtains its minimum median value of 0% by 1000 runs, whereas the combined method can only reach this after 10000 runs, despite it's mean value being lower than priority at both points.

The mean progress of the methods over the 10000 runs can be observed in Figure 2. It can be seen from this that priority yields very good results with only a few runs, while after a large number little more optimisation is possible. Both resource use and the combined method fall steadily, exceeding the performance of the priority only after a large number of runs. The mean performance of the combined method exceeds that of resource usage for all but a very small number of runs. This shows that inclusion of priority rules in a resource use optimisation will almost always represent an improvement.

Summary. Incorporating both types of network input attribute in the same optimisation has increased the size of the optimising surface considerably. From the results, a reasonable size of unbiased search finds, on average, better solutions with this combined search space. However, a better solution than in both the restricted search spaces is only found in just under 20% of cases. It can be seen from the large standard deviation of the mean performance of the combined optimisation that much of the increased search space in some problems is unfavourable, and impedes the progress of the optimisation.

Further Work

The testing of two types of network input attribute on this problem has indicated that it is advantageous to consider them in combination. This demonstrates that the Global Optimisation model proposed is applicable to this small problem. However, if the model is to be used in practical situations then it needs considerable development beyond the simple application which has been demonstrated. This applies to three areas: the realism of the data used, the realism of the model used and the efficiency of the optimising algorithm used.

Data. The problems tested here, though useful for demonstrating a simple application of the global model, were not designed to be realistic models of projects. They were created to test optimising algorithms and near optimal heuristic rules for resource constrained scheduling.

To properly appreciate how this approach would benefit real projects, problems should be created whose characteristics are a much improved model of the behaviour of real projects. An even better approach would be to use real project data.

Realism of modelling. The model used here is a very simple model of the project. It ignores many of the details and relationships between different aspects of the project. This is why this approach is often criticised for not giving feasible solutions.

There are many ways in which the model's accuracy could be increased. These could include considerations of non-renewable resources, non-constant resource availability, space availability, formations of crews, costing, financing and risk, as well as the interactions between all these aspects of projects.

Simply adding all these aspects to the project model at once would suddenly cause an enormous increase in its size. While this would provide an accurate model of the project, it would give no indication of how each new aspect of the model affects its behaviour in an optimisation. This in turn would make strategies for increasing the efficiency of optimising techniques very difficult to formulate. If the model were increased in complexity in a series of small stages, then how each increase in complexity affects the model could be carefully monitored. This is very useful in developing optimisation strategies.

Optimisation Strategies. In this paper was an unbiased random search method was used to find near optimal solutions. In a large project, where the number of network input attributes is high, and the model is large, a large number of time consuming runs would be required to find reasonable optima. There are several ways in which the number of runs required can be reduced. They all hinge on concentrating on the areas of the feasible region where good solutions are coming from.

The first method is to reduce the size of the feasible region. It may be possible to identify areas in the feasible region where optimal solutions will either definitely not be found, or are very unlikely to be found. These areas can then be either eliminated from the feasible region, or given a low probability of being selected by the optimising algorithm. The latter would be done by weighting the selection of values for network input attributes.

The second method is to use techniques which operate by performing small changes to solutions which are known to be good, and give a higher probability of the changes being rejected if the solution causes an deterioration in the value of the objective function. This causes the solutions to stay in the regions where good solutions are coming from. Genetic algorithms and simulated annealing are two good examples of this type of technique.

If both reduction of the feasible region and improvement of the optimising algorithms can be used in combination, then it may be possible to substantially increase the speed with which good solutions are found.

Overall. The three ways of progressing with the development of the model proposed would be best used in parallel. The realism of data used to monitor the performance of the model and optimising techniques should be increased. The detail of the model itself should be increased in stages, with the performance of optimising techniques observed and refined for each stage. This would allow a constant monitoring of how the feasible region is affected by increases in detail of the model.

If data from real projects were used for this, then it would facilitate the comparison of different stages, as data relevant to every stage of the model should be available.

Conclusions

A global model for the optimisation of projects has been proposed. This model aims to produce improved results over currently implemented forms of more localised optimisation within a project by considering all variable factors within a project in a single optimisation.

The model has been implemented for the simple resource constrained scheduling problem. This has demonstrated that there are advantages of using a global approach to optimisation even with an unweighted random search technique.

Based on the success of this preliminary investigation, possible fields for investigation have been identified, along with a recommended method of investigation.

Nomenclature

d = activity duration, in time periods
r = rate of resource use by an activity, in resource units per time period
r_p = rate of primary resource usage, resource units per time period
r_i = rate of use of resource i, resource units per time period

R = total number of units of a resource required by an activity, resource units
R_p = total primary resource requirement, resource units
R_i = total requirement of resource i, resource units
S = the set of non-primary resources required by the activity

Acknowledgements

The authors would like to thank James Patterson for the 110 problems for analysis.

References

1. Li, R. K.-Y.; Willis, R. J. "Resource Constrained Scheduling Within Fixed Project Durations" *Journal of the Operational Research Society* (1993) **44** 71
2. Yau, C.; Ritchie, E. "Project Compression: a Method for Speeding up Resource Constrained Projects Which Preserves the Activity Schedule" *European Journal of Operational Research* (1990) **49** 140
3. Norbis, M. I.; Smith, J. M. "Multiobjective, Multi-level Heuristic for Dynamic Resource Constrained Scheduling Problems" *European Journal of Operational Research* 1988 **33** 30
4. Flanagan, R.; Norman, G.; Meadows, J.; Robinson, G. *Life Cycle Costing: Theory And Practice*, first edition, BSP Professional Books (1989) 6
5. Becassi, W., Tukel, O. I. "A New Framework For Determining Critical Success/Failure Factors In Projects" *International Journal Of Project Management* (1996) **14** 141
6. Jaafari, A. "Time And Priority Allocation Scheduling Technique For Projects" *International Journal Of Project Management* (1996) **14** 289
7. Kavanagh, C. P. "SIREN: A Repetitive Construction Simulation Model" *Journal of Contruction Engineering and Management* (1985) **111** 308
8. Lee, J.-K., Kim, Y.-D.: "Search Heuristics For Resource Constrained Project Scheduling" *Journal of the Operational Research Society* (1996) **47** 678
9. Patterson, J. H.: "A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem" *Management Science* (1984) **30** 854

Table 1 - COMPARISON OF METHODS

	Uniquely Best Solutions	Best Solutions	Target Value reached	Distance to target: 1000 runs Mean	1000 runs Median	1000 runs Std. Dev.	10000 runs Mean	10000 runs Median	10000 runs Std. Dev.
Resource Use	4%	40%	53%	14%	25%	76%	-7%	0%	80%
Priority	30%	57%	81%	9%	0%	18%	5%	0%	13%
Combined	18%	65%	67%	4%	17%	79%	-18%	0%	82%

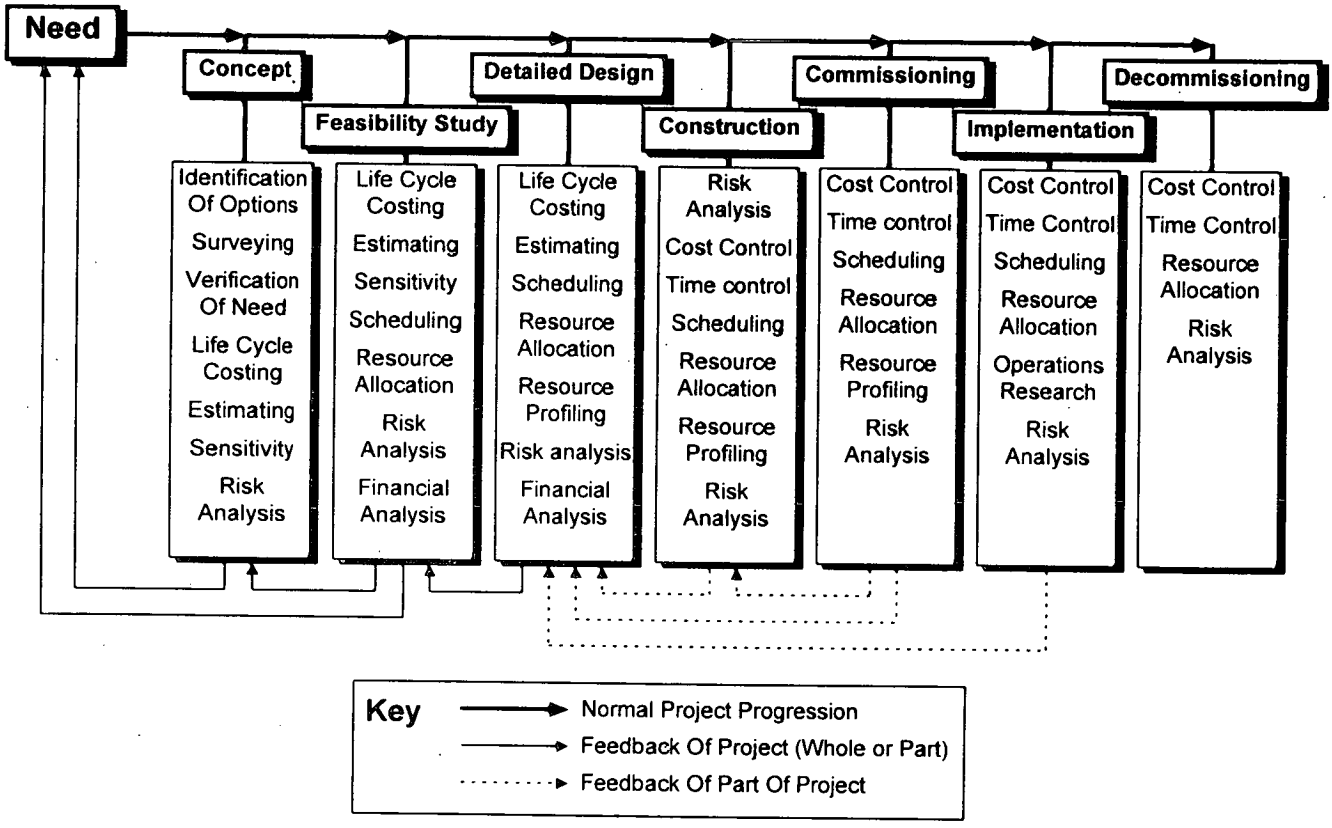


Figure 1 - Analytical Techniques Within the Project Life Cycle

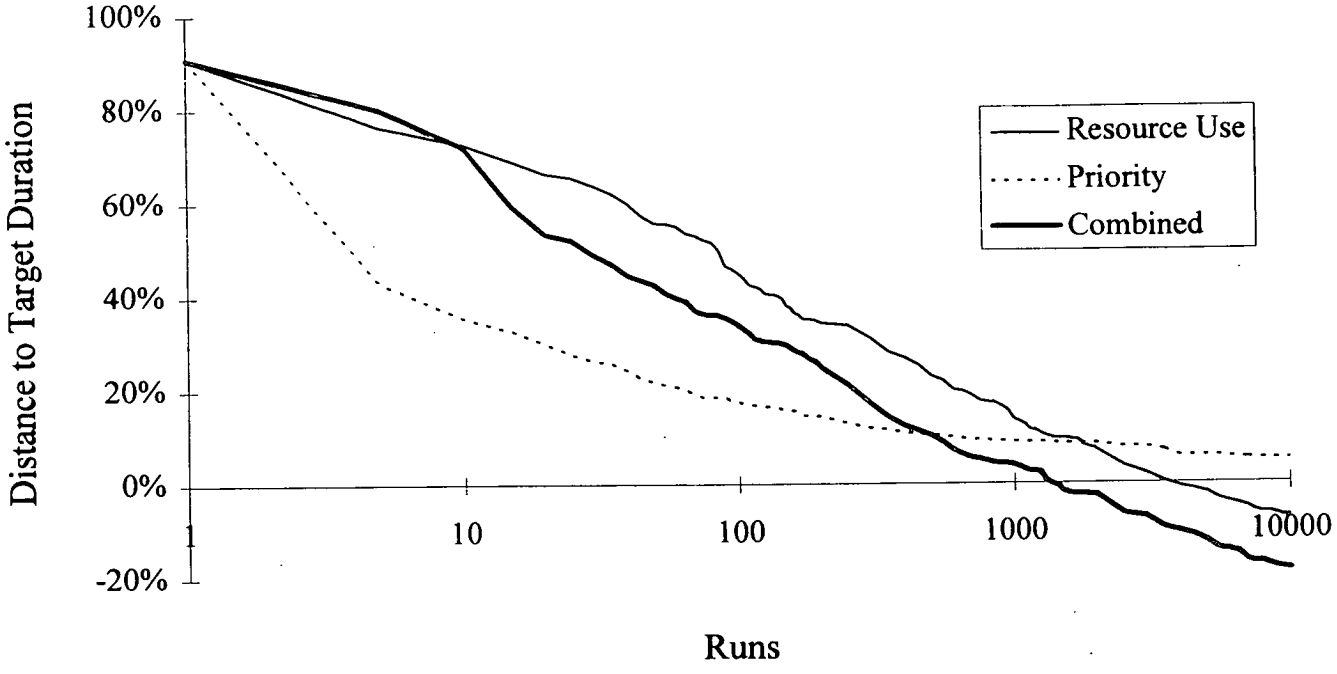


Figure 2 - Mean Performance of Optimisation

EXPERIENCE WITH A GLOBAL-OPTIMIZATION APPROACH TO PROJECT SCHEDULING AND RESOURCE ALLOCATION

A.M. Harding and D.A. Ponniah, U. of Edinburgh

SUMMARY

In recent years, considering the whole project life cycle and its costs has become a growing trend in major projects. This analysis initially considered concept to commissioning; later, particularly after Brent Spar, concept to decommissioning was considered. Optimal solutions are sought within this whole project life cycle. However, the field of optimization still tends to focus on improving the performance of project elements or small subsystems within the project. Optimization across the whole project life cycle is not normally carried out. This paper proposes a general system for a global-optimization model that allows the whole project to be considered within a single optimization.

The strategy consists of a basic scheme for input to an objective function and the definition of its constraints. The objective function can map to a value relating to cost, time, performance, or risk, allowing the most important criteria to be maximized or minimized. Values not mapped by the objective function can be used as constraints to restrict any detrimental effects on them during the optimization. A true globally optimizing model is extremely large and complex. For the model to be able to optimize globally, it must also be able to arrive at an optimal solution for a local optimization problem. Therefore, its suitability for optimization is tested on project scheduling and resource allocation, a subsystem of the project.

A new approach to modeling project schedule and resource allocation is developed that allows the model to fit into the globally optimizing model. With examples from the literature, the performance of this model on specific locally optimizing problems is investigated by use of Monte Carlo random search. Although this is not normally considered to be an efficient means of finding an optimum, we considered it to be the best approach to an initial investigation where the nature of the optimizing surface is unknown. In particular, the model is used to solve resource-constrained scheduling problems. The difficulty of finding the optimum within the model is also studied, and extensions to the model for further research are proposed.

INTRODUCTION

Project optimization has become an important field of study in recent years, fueled by a desire to complete projects more quickly, for less cost, and to a higher quality. Many of the techniques arising from this research have been simple mathematical optimizations aimed at solving specific problems within specific stages of projects. However, the majority of the techniques developed by this research have not been applied to real problems, principally for two reasons.

Original SPE manuscript received 9 September 1997. Paper (SPE 38488) originally presented at the 1997 SPE Offshore Europe Conference held in Aberdeen, 9–12 September. This paper has not been peer reviewed.

1. Optimizations often take the form of models that consider only a few aspects of the project, and the true mathematical optimum for the model is often an unfeasible solution for the real project it is set up to simulate.[1,2]

2. Techniques can often be very demanding on computational resources, particularly for very large projects. Any savings they might make are offset by the cost of computing them, even if computation time is reasonable.[3]

Given these criticisms, if any optimization is to be applicable to real problems, it clearly must use a sufficiently accurate model of the project to ensure that solutions considered are realistic and it must find either an optimal solution or an acceptable near-optimal solution with a reasonable amount of computational effort. These are the aims of the globally optimizing model proposed in this paper. In addition, the globally optimizing model is implemented to solve the resource-constrained project-scheduling problem to demonstrate the advantage of considering more than one type of variable within a simple project-analysis problem.

MODELING PROJECT LIFE CYCLE

Project life cycle is often considered to be the phases through which a project passes between concept and completion (commissioning and handover). However, a common cause of conflict within a project is that the project manager is primarily concerned with the success of the project only up to what he/she sees as completion and fails to understand fully the needs of the customer beyond this point. As the customer sees the project, it begins with a "need" and ends with the decommissioning of the project at the end of its useful life.[4]

The aim of a project is not necessarily to ensure completion of construction and commissioning on time and on budget, as so often considered today.[5] The true aim should be to ensure that the project succeeds in meeting the needs of the customer in the most efficient way possible, and not only in terms of on time and on budget. For this reason, any consistent model of a project life cycle should be considered from concept to decommissioning and any approach to optimization of it should be sufficiently flexible to be sensitive to the needs of the customer. **Fig. 1** illustrates the various stages of the life cycle and the analytical techniques that are usually used during these phases. These stages are briefly described next.

Project Phases. ***Concept.*** In this phase, possible solutions to the need are addressed. This phase may produce possible solutions to be passed on to the feasibility phase, or it may allow the need to be resolved in some alternative way.

Feasibility. In this phase, one or more (usually several) options are considered in more detail than in the concept phase. These usually include more accurate cost estimates, performance criteria, and time factors for these options and also an approximate design. For this phase to pass into the detailed-design phase, one option usu-

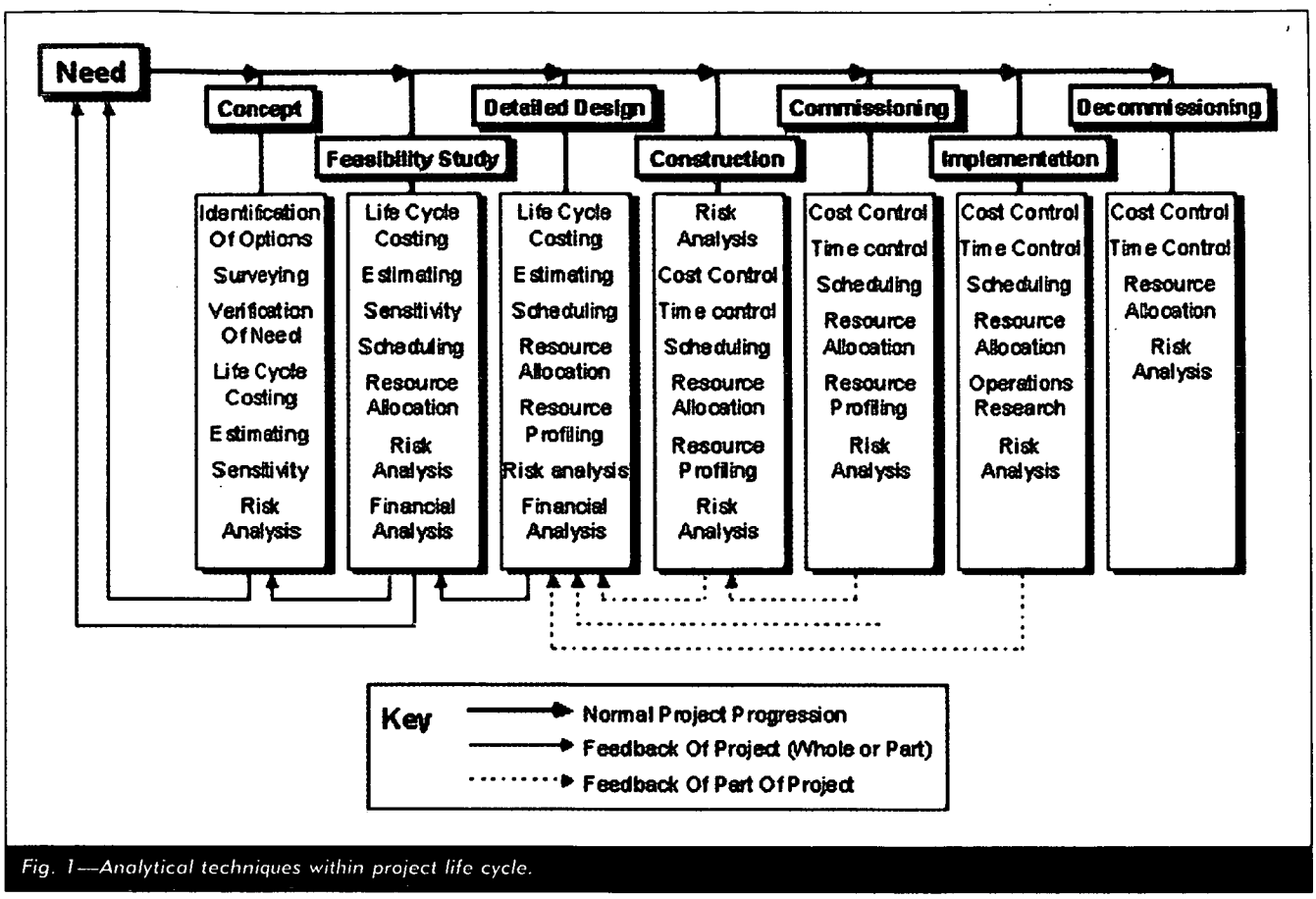


Fig. 1—Analytical techniques within project life cycle.

ally must be selected. This phase can also reveal the need for re-evaluation at the concept stage or reveal an alternative means for resolution of the need.

Detailed Design. In addition to producing a detailed design of the solution, this phase is also concerned with planning, including contractual arrangements, project scheduling, and procurement of materials. If the project is acceptable, it progresses to the construction phase. It is very uncommon and undesirable for a project to be reconsidered at this stage because considerable resources already have been committed. This is particularly true in the later stages of this phase. Theoretically, however, the project can be passed back to the feasibility stage and re-evaluated.

Construction. When the project enters the construction phase, returning the whole project to previous stages becomes impossible because part of the project already exists physically. However, parts of the project may have to be passed back to the design phase if mistakes in the design are revealed during construction and need to be amended.

Commissioning. This stage involves inspection and testing of all aspects of the project to ensure that it is able to meet the customer's need. This may reveal that parts of the project have not been designed or constructed to the required specification, and these parts may have to be redesigned or reconstructed.

Implementation. This project phase, usually the longest and most costly, involves operation of the project and demonstrates whether the need has been met. Again, parts of the project can be passed back to the design phase because elements of the solution may become inadequate over time. However, unlike the construction and commissioning phases, this is not necessarily caused by design or construction errors. The implementation phase is generally more strongly affected by a changing environment than any other phase, particularly by advancing technology. Thus, partial redesign is often inevitable in this stage.

Decommissioning. This phase takes place when the solution has reached the end of its functional life and involves the freeing up of resources taken up by it.

At each stage, key decisions are made that restrict the scope of decisions made later in the project. Each project phase has a number of analytical techniques that are applied to the existing project information to produce data to help make these decisions. Fig. 1 shows these techniques and also where part or all of the project can be passed back to an earlier phase for re-evaluation.

At the start of the project, the scope for optimization is very large. Many possible solutions to the problem exist. As the project progresses beyond feasibility and one particular solution is chosen, the scope for optimization is restricted to what is available within that solution. A great deal of optimization is performed in the detailed-design phase and continues throughout the project life cycle. However, as each stage progresses, the amount of optimization that can still be performed is reduced because of restrictions imposed in previous phases. By the time the decommissioning stage is reached, comparatively little further optimization can be performed. Therefore, at each stage (or perhaps even repeatedly within each stage), a global optimization of the remainder of the project must be performed to help determine the best way to continue the project. This should be performed with the most-up-to-date information on both progress of the project to date and progress expected in the future.

GLOBAL PROJECT OPTIMIZATION

To perform any project optimization, the project must be modeled in a way that is both accurate and consistent with the level of detail of the information available within the project. Thus, the model must meet three basic conditions to be used effectively in the optimization.

1. Representation. At any stage the model must be capable of representing the project as a whole, from concept to decommissioning.

2. Increasing detail. As the project passes from stage to stage, the information available becomes increasingly detailed. The model must be able to incorporate and model accurately these increasing levels of detail.

3. Numerical input and output. Precise numerical input and output is required for optimization; therefore, the model needs to be numerically based.

A precedence network is a simulation model of a construction project. The model is first created by breaking the project down into activities. By modeling with information available at that stage of the project, these activities are assigned values to represent as closely as possible their behavior in the real world. These values may be items as simple as duration, cost, and activities that must be completed before the activity can be scheduled. This is a very simple model, and a critical-path schedule is the elementary form of an optimization. With increasing complexity, other network-input characteristics can be taken into account. These network-input characteristics include such items as resource demands, productivity, crew and resource allocations for each activity, crew and resource availability, cash flow, and other considerations for the project as a whole. Accounting for these characteristics involves modeling the project as a queuing system. In such a system, activities line up for resources once their precedence relationships have been fulfilled. This is commonly held to be a more accurate model of a project than an ordinary precedence network.[6,7] Such a model allows representation of the project at all stages, supports increasing detail, and is entirely numerical; therefore, it fulfills the three requirements.

Terminology. Within the considered model, terminology has been developed to define the information used within the model.

1. Network-input characteristics. Network-input characteristics are all those characteristics used to evaluate the network: activity, resource-availability, costing, and other such factors.

2. Network-input attributes. Numerous network-input characteristics exist. While many of these are predetermined because they are fixed inherently, defined by probability distributions, or a function of other characteristics, a large number of network-input characteristics may be assigned one of a range of values. These latter values are called the network-input attributes.

3. Network-output characteristics. Network-output characteristics are the various values calculated by the scheduling logic from the network-input characteristics. They include such items as start times of all activities, resource usage at any time period, and bank balance at any time period.

4. Network-output attributes. Network-output attributes are the most important network-output characteristics of the project and are monitored as part of the optimization. They can be used to impose criteria on how the project should behave and to form the output of the objective function used in optimization. Actual network-output characteristics used as network-output attributes vary from project to project.

5. Solution set. A solution set for a network is a set of network-input attributes and the corresponding network-output attributes.

Global-Model Execution and Optimization. The global model operates by applying a scheduling logic to determine network-output characteristics by use of network-input characteristics. The complexity of the logic used to do this depends on the level of detail of the network-input characteristics available. This constitutes one run of the scheduling logic and produces a single solution set. Currently, this solution generally is considered to be fixed because the network-input attributes are considered to be fixed. Their values are usually determined with engineering judgment and subsequently considered as fixed, or, at best, they are determined with some sort of local optimization.

When a global optimization is performed on a project, the network-input attributes are considered to be variable. An optimization technique sets the values of the network-input attributes before the scheduling logic is run to determine the network-output characteristics. The network-output attributes are extracted from these network-output characteristics, and a solution set is obtained.

Constraints imposed on network-input attributes define the region that is initially considered to be the feasible region by the optimizing algorithm. Further restrictions on the feasible region can be applied by use of network-output attributes, which are discussed later. Each network-input attribute has a maximum and minimum value, as well as a base unit. Any value of a network-input attribute should be a multiple of its base unit and lie between the maximum and minimum values.

Network-output attributes can be used in two ways. First, they can impose restraints on which solutions should be considered feasible. These restraints are in addition to those that are imposed on the network-input attributes. One example of the use of network-output-attribute constraints is where the overall construction time for a project is considered to be a network attribute. In this case, a maximum construction time could be specified. If this time is exceeded, the optimization technique should consider the solution as unfeasible. This prevents the optimization technique from finding a solution that is optimal to one aspect of a project but at the same time unacceptably detrimental to other aspects of the project. Imposing such restraints can be done with either many network-output attributes or none.

The second way in which network-output attributes may be considered is as the output of the objective function for an optimization. The optimization technique seeks either to minimize or maximize this variable, depending on the nature of the network-output attribute. For example, maximizing the net present value of a project or minimizing the time it takes to reach a project's break-even point may be desirable. Only one network-output attribute should be considered in this way in any single optimization because the output of an objective function for optimization should always be a one-dimensional value.

Optimizing the Model. Once a feasible region and output for the objective function is defined, an optimization can be performed on the model. The greatest difficulty with optimizing the model may be that the behavior of the optimizing surface is unknown. The effects of certain changes in network-input attribute are not known until the scheduling logic is run and also cannot be predicted, if at all, in any other way. The number of local optima within the feasible region and the sensitivity of the global optimum are also unknown. Because of these problems, we rejected many widely used optimization methods.

Solution of the model with rigorous mathematical methods (such as linear, integer, or geometric programming) is exceedingly difficult. The number of constraints and different inputs undoubtedly would make it necessary to solve enormous matrices for the linear-programming model if this technique could be used at all. The number of evaluations required by an integer- or geometric-programming method would be prohibitive. Use of gradient methods was also rejected because the behavior of the optimizing surface is unknown and these methods tend to find the only nearest optimum. If many local optima are present, a gradient method may not necessarily find a good optimum.

Random-search techniques are proposed for optimization of the model. They provide a simple means of searching the whole optimizing surface for good solutions and are not sensitive to any

unusual or complex behavior of the optimizing surface. This type of technique requires storage space for only a small number of solutions. A random search could take either the form of a simple Monte Carlo simulation or of more complex methods, such as genetic algorithms or simulated annealing, which tend to focus on areas where good solutions are being found. While these methods often do not find the true mathematical optimum of the model, they should find good near-optimal solutions. A trade-off often exists between the amount of computational effort used and the quality of the solution found. Therefore, the best solution with the available computational resources can be found and used, while many other methods simply yield no solution without sufficient resources.

INCORPORATING THE RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEM INTO THE GLOBAL MODEL

To test the validity of the model in a simple case, it was used to optimize the resource-constrained scheduling problem. Although this is a simple and well-documented problem that is subject to many of the criticisms raised about project models, we used it to try to demonstrate that considering more than one type of variable at the same time within a project optimization has clear advantages.

The resource-constrained scheduling problem attempts to find the minimum duration of a project given certain upper limits on resource usage. This model is usually considered to have several renewable resources. Renewable resources, like manpower or plant, can be reused in the next and subsequent time periods. This means that to solve the problem, overall resource use for all activities should not exceed the maximum allowable resource use (or ceiling) for that time period. This ceiling is usually considered to be constant for all time periods. The fixed-ceiling problem with renewable resources is the type of problem considered here.

Two approaches can be taken to solve this problem. The first is to delay certain activities in a way that yields a minimum duration without exceeding the resource demands. The second, sometimes called project compression, involves compressing activities in parts of the project where the resource usage is lower than the limit. This paper considers both methods.

Activity Priority. Rearranging activities is considered by allowing the scheduling logic to prioritize certain activities in different ways. Each activity is assigned a priority value. If, at any time period, more than one activity is available for scheduling, they are sorted in descending order of priority. These activities are then scheduled one by one, and all activities that would cause resource demands to exceed the limit are not scheduled; instead, they are made available for scheduling at the next time period. The priority value is considered as a network-input attribute. Lee and Kim[8] used activity priority to reduce project durations. In most cases, this method eventually should yield an optimal solution.

Resource Use. In considering the resource usage of an activity, the activity is given a total resource requirement for each resource. For the project-compression solution to the problem, it is assumed that this total requirement remains constant irrespective of the allocation of resources per period. Therefore, the relationship between activity duration and per period usage of a resource can be expressed by

$$t = n/r. \qquad (1)$$

A network-input attribute can be assigned to this in two ways. First, the duration can be controlled. For this problem, this would appear to be the easiest way to control the resource allocation. However, as the detail and realism of the model is increased, the total resource requirement may change with changes in the allocation per period. to include some consideration of manpower efficiency. In this case, use of the duration may become quite cumbersome because it is only a secondary characteristic, dependent on resource allocations. Therefore, we propose a new method of controlling resource allocations and durations.

Often, the most consequential resource allocation is of manpower. Crew efficiency on site can vary with different factors, such as whether the crew has just come from another similar activity type or the size of the crew. A detailed model usually considers activities that are ready to be scheduled as a queue of activities waiting for a finite number of crews. This suggests that the size of crew may be the least flexible aspect of the resource allocation to an activity. Therefore, we propose that this be used as the network-input attribute. Given that cases may occur where another type of resource may control the duration in a similar fashion, this network-input attribute has been given the more general name "primary resource use."

Allocation of a primary resource is used to calculate a projected duration of the activity. From this, the activity's requirements of other resources per time period can be calculated by simply dividing the overall requirement by the projected duration. This is expressed mathematically as

$$t = n_p/r_p \qquad (2)$$

and $t_i = n_i/t \, \forall i \in S.$ (3)

ANALYSIS

We analyzed the 110 problems assembled by Patterson[9] to compare heuristic rules for near-optimal solution of resource-constrained scheduling problems. These problems were used because the optimal solutions for the case of fixed activity duration was known for each problem and could be used as a benchmark value for measuring the performance of each optimizing technique.

The model was programmed into a computer with C++ and run on a personal computer. A Monte Carlo random-search procedure was created for the optimization. The random selection of values for the network-input attributes was unweighted, so the probability distribution across the whole field was even for each network-input attribute. The random-number generator used by the program was reseeded with a constant value at the start. This ensured repeatable results because the same sequence of random numbers would be generated if the same optimization was performed.

Priority was assigned integer values between 1 and 100. Primary resource use was assigned a range between 1 and the greatest number permissible before the activity would demand more than the maximum availability of any resource it required. The optimization procedure was limited to selecting integer values of primary resource use. If this led to noninteger requirements of other resources, they were rounded up to the nearest integer. Similarly, if allocation of a primary resource use led to a noninteger activity duration, the duration was rounded up to the nearest integer. The network-output attribute is the finish time for the last activity.

RESULTS

A Monte Carlo random search was performed on the 110 Patterson problems, and the best solution found was taken. The search used 10,000 schedules of the project with randomized network-input attributes. All priority and resource use were first considered on their own and then in combination. In each run, an initial

TABLE 1—COMPARISON OF METHODS

	Uniquely Best Solutions (%)	Best Solutions (%)	Target Value Reached (%)	Distance to Target: 1,000 Runs Mean (%)	1,000 Runs Median (%)	1,000 Runs Standard Deviation (%)	10,000 Runs Mean (%)	10,000 Runs Median (%)	10,000 Runs Standard Deviation (%)
Resource Use	4	40	53	14	25	76	−7	0	80
Priority	30	57	81	9	0	18	5	0	13
Combined	18	65	67	4	17	79	−18	0	82

value was obtained from prioritizing activities in order of appearance in the network and the resource allocations specified for the project.

The optimization's progress from this initial value was then measured vs. target durations. These target durations are the minimum project durations for the network when no change in activity duration and requirements is permitted. They were with the projects as optimal durations supplied. The progress of each optimization was measured as a proportion of the distance traveled from initial to target duration, with 1 representing no change from the initial duration, 0 representing the target duration, and a negative value indicating a project duration less than the target duration. Where there was no difference between initial and target durations, the mean difference between the two for all 110 problems was used to calculate the proportion change for the individual problem.

Table 1 summarizes the results of these runs. The table shows several performance indicators for the methods. Best solutions shows the number of projects in which that method found the best duration. Uniquely best solutions indicates the number of these best solutions that the method was alone in finding this value. Target value reached shows how often the method achieved the target value. Mean performance indicators for 1,000 and 10,000 runs are also shown.

The priority method was more successful than the other methods at finding uniquely best solutions and at reaching the target duration. The performance of the combined method in terms of average distance to target is superior to the other two methods for both the 1,000 and 10,000 runs. It also succeeded in finding the best duration found for nearly 10% more runs. However, it was unique in achieving this duration in only 18% of the cases, while solution by the priority method was unique in 30% of the cases. This apparent disparity can be explained by considering the standard deviation of the performance indicators for both 1,000 and 10,000 runs in the table. This clearly shows that performance of both the resource use and combined methods varies considerably from problem to problem. Where their performance is low, priority is likely to be the only method that yields significant optimization. This is because the large number of poor solutions from the resource use make it more difficult for this method to yield good solutions for the priority problem.

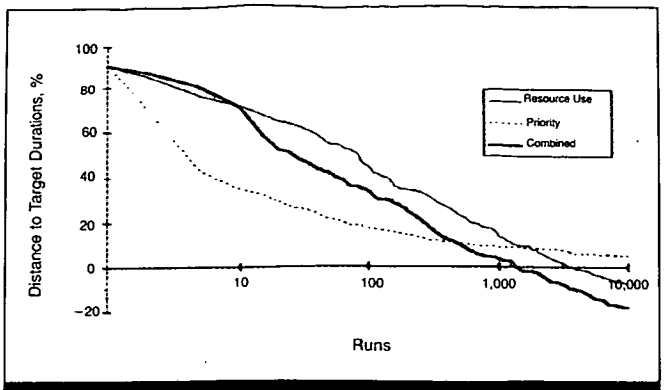


Fig. 2—Mean performance of optimization.

This can also be seen in the median values. Priority obtains its minimum median value of 0% by 1,000 runs, whereas the combined method can reach this only after 10,000 runs, despite the fact that its mean value is lower than that of the priority method at both points.

Fig. 2 shows the mean progress of the methods over the 10,000 runs. One can see that priority yields very good results with only a few runs and that little more optimization is possible after a large number of runs. Both resource use and the combined method fall steadily, exceeding the performance of the priority only after a large number of runs. The mean performance of the combined method exceeds that of resource usage for all but a very small number of runs. This shows that inclusion of priority rules in a resource-use optimization almost always represents an improvement.

Summary. Incorporating both types of network-input attributes into the same optimization has increased the size of the optimizing surface considerably. From the results, a reasonable-sized unbiased search finds (on average) better solutions with this combined search space. However, a better solution is found after a large number in only just under 20% of cases than found in both the restricted search spaces. The large standard deviation of the mean performance of the combined optimization shows that much of the increased search space in some problems is unfavorable and impedes the progress of the optimization.

FURTHER WORK

Testing of two types of network-input attribute on this problem indicated that considering them in combination is advantageous. This demonstrates that the global-optimization model proposed is applicable to this small problem. However, if the model is to be used in practical situations, it needs considerable development beyond the simple application demonstrated here. This applies to three areas: the realism of the data used, the realism of the model used, and the efficiency of the optimizing algorithm used.

Realism of Data. The problems tested here, although useful for demonstrating a simple application of the global model, were not designed to be realistic project models. They were created to test optimizing algorithms and near-optimal heuristic rules for resource-constrained scheduling. To appreciate properly how this approach would benefit real projects, problems should be created that have characteristics that are a much improved model of the behavior of real projects. An even better approach is to use real project data.

Realism of Model Used. The model used here is a very simple model of the project that ignores many of the details and relationships between different aspects of the project. This is why this approach is often criticized for not giving feasible solutions. Model accuracy could be increased in many ways, including consideration of nonrenewable resources, nonconstant resource availability, space availability, crew makeup, costing, financing, risk, and the interactions between all these aspects of projects.

Adding all these aspects to the project model at one time causes a sudden enormous increase in its size. While this addition provides an accurate model of the project, it gives no indication of how each new aspect of the model affects its behavior in an optimization. This, in turn, makes strategies for increasing the efficiency of optimizing techniques very difficult to formulate. If model complexity is increased in a series of small stages, the effect of each increase in complexity on the model can be carefully monitored. This is very useful in developing optimization strategies.

Optimization Strategies. We used an unbiased random-search method to find near-optimal solutions in this study. In a large project where the number of network-input attributes is high and the model is large, a considerable number of time-consuming runs would be required to find reasonable optima. The number of runs required can be reduced in several ways that all hinge on concentrating on the areas of the feasible region that are providing good solutions.

The first method is to reduce the size of the feasible region. It may be possible to identify areas in the feasible region where optimal solutions either definitely will not be found or are very unlikely to be found. These areas then either can be eliminated from the feasible region or given a low probability of being selected by the optimizing algorithm. The latter is done by weighting the selection of values for network-input attributes.

The second method is to use techniques that operate by performing small changes to solutions that are known to be good and, at the same time, to give a higher probability of the changes being rejected if the solution causes a deterioration in the value of the objective function. This causes the solutions to stay in those regions that provide good solutions. Genetic algorithms and simulated annealing are two good examples of this type of technique. If both reduction of the feasible region and improvement of the optimizing algorithms can be used in combination, substantially increasing the speed with which good solutions are found may be possible.

Overall. The three ways of progressing with the development of the proposed model are used best in parallel. The realism of data used to monitor the performance of the model and optimizing techniques should be increased. The detail of the model itself should be increased in stages, with the performance of optimizing techniques observed and refined for each stage. This allows constant monitoring of how the feasible region is affected by increases in detail of the model. Use of data from real projects facilitates comparison of different stages because data relevant to every stage of the model should be available.

CONCLUSIONS

A global model for optimization of projects has been proposed. The model aims to produce improved results over currently implemented forms of more localized optimization within a project by considering all variable factors within a project in a single optimization. The model has been implemented for the simple resource-constrained scheduling problem. This demonstrated that using a global approach to optimization has advantages even with an unweighted random-search technique. On the basis of the success of this preliminary investigation, possible fields for