

Ashton, Andrew R. (1989) Computer aided analysis and design of mine transportation systems. PhD thesis, University of Nottingham.

Access from the University of Nottingham repository:

<http://eprints.nottingham.ac.uk/13263/1/328425.pdf>

Copyright and reuse:

The Nottingham ePrints service makes this work by researchers of the University of Nottingham available open access under the following conditions.

- Copyright and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners.
- To the extent reasonable and practicable the material made available in Nottingham ePrints has been checked for eligibility before being made available.
- Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.
- Quotations or similar reproductions must be sufficiently acknowledged.

Please see our full end user licence at:

http://eprints.nottingham.ac.uk/end_user_agreement.pdf

A note on versions:

The version presented here may differ from the published version or from the version of record. If you wish to cite this item you are advised to consult the publisher's version. Please see the repository url above for details on accessing the published version and note that access may require a subscription.

For more information, please contact eprints@nottingham.ac.uk

*University of Nottingham
Department of Mining Engineering*



***COMPUTER AIDED ANALYSIS AND DESIGN
OF MINE TRANSPORTATION SYSTEMS***

by

Andrew R. Ashton, BEng.

*Thesis Submitted to the University of Nottingham
for the degree of Doctor of Philosophy,*

May, 1989

AFFIRMATION

The work submitted for this thesis is my own and has not been previously submitted for any other degree. The following publication has been based on this research:

Brown, D.J., Ashton, A.R., Croghan, J.A., Johnson, S.M.
"Concepts in Computer Aided Mine Design and Planning",
Mining Science and Technology, Vol. 7, 1988, pp 99-119.

ACKNOWLEDGEMENTS

The author would like to express his sincere gratitude to the following:

Dr Bryan Denby for his continuous assistance and his enthusiastic leadership of the NUmine project;

Mr Martin Waller for his guidance throughout the course of study and his help in the preparation of this thesis;

Prof. Tom Atkinson for his ever-present support and for making the department such a creative and dynamic place to work;

and finally,

James Croghan and **Siavash Kordestani**, the other team members, for their constructive suggestions and persistent encouragement.

ABSTRACT

Haulage Costs account for a considerable portion of a surface mine's operational budget. It is therefore vital that, for a particular pit configuration, the optimum utilisation of the available truck fleet is adopted during the mine's life. Also, if the optimisation methods are established beforehand, it is possible to determine exactly how many trucks will be required. Both decisions can be made at the planning stage by the application of linear programming and discrete simulation to computer models of the haulage network.

The project presented herein investigates the practicality of developing a general-purpose mine transportation selection and scheduling system within the context of a Computer Aided Design (CAD) environment. Compatibility with a purpose-built, interactive graphics package is shown to enable rapid, semi-automatic generation of model networks and the planning engineer is assisted further by the robust and friendly user-interface which has also been developed. Unlike a number of existing packages, which either make use of commercially available software on a stand-alone basis or were specifically designed for the analysis of a particular operation, this system is completely integrated with a central database which makes it applicable to any mine.

The enhanced ability to produce valid mathematical solutions and their associated network models using the above systems, allows a large number of configurations and dispatching policies to be compared in a relatively short space of time. However, attention is also paid to the degree of correspondence with what can be achieved in reality since this will also effect the selection decision.

All the modules mentioned form part of a much larger planning system currently being developed at The University of Nottingham, Department of Mining Engineering, known as NUmine.

TABLE OF CONTENTS

AFFIRMATION	i
ACKNOWLEDGEMENTS.....	ii
ABSTRACT.....	iii
TABLE OF CONTENTS.....	iv
LIST OF FIGURES.....	x
LIST OF TABLES.....	xiv
LIST OF PLATES.....	xv

Chapter 1 Introductory Chapter 1

1.1	Introduction	2
1.1.2	Truck Dispatching.....	3
1.1.3	The Modelling Process	3
1.1.4	Computer Aided Design.....	4
1.1.5	Objectives	5
1.1.6	Thesis Format.....	5
1.2	NUMine.....	7
1.2.1	NUMine Philosophy.....	7
1.2.2	NUMine Structure.....	9
1.3	Simulation Modelling.....	11
1.3.1	Stages in the Modelling Process.....	11
1.3.2	Simulation Categorisation	12
1.3.2	Simulation Languages	13
1.4	Simulation in the Mining Industry.....	17
1.4.1	Production.....	17
1.4.2	Locomotives	18
1.4.3	Conveyors.....	19
1.4.4	General Systems.....	20
1.4.5	Language Comparisons.....	21
1.5	Conclusions	22

Chapter 2 A Review of the Development of Truck/Shovel 23
Simulation and Dispatching Systems

2.1	Introduction	24
2.2	Selection of Haulage Equipment.....	26
2.2.1	Cycle Time Calculations.....	27
2.2.2	Performance Simulation Packages	31
2.2.3	Prediction of Productivity Variations	34
2.3	Simulation of Truck/Shovel Systems.....	35
2.3.1	The Model Building Process.....	35
2.3.2	Truck/Shovel Simulation Models.....	37
2.3.3	Problems with Truck/Shovel Simulation	40
2.4	Truck Monitoring and Control.....	41
2.4.1	The Need for Greater Control.....	41
2.4.2	Manual Control Systems	42
2.4.3	Semi-Automated Control Systems	42
2.4.4	Automated Control Systems	44
2.5	Analysis of Dispatching Systems by Simulation	50
2.5.1	Standard Dispatching Policies	51
2.5.2	Non-Standard Dispatching Policies	56
2.5.3	Summary of Dispatching Policies	57
2.6	Automated Truck Dispatching.....	58
2.6.1	DISPATCH.....	66
2.6.2	Micro-Based Systems.....	69
2.6.3	Summary of Automated Dispatching.....	69
2.7	Advanced Dispatching Techniques	70
2.7.1	Linear Programming	70
2.7.2	Dynamic Programming	74
2.7.3	Summary of Advanced Dispatching Techniques	76
2.8	Conclusions	77

Chapter 3 NUmine - A Computer Aided Mine Design and 78
Planning System

3.1	Introduction	79
------------	---------------------------	-----------

3.2	The NUmine Environment.....	80
3.2.1	Hardware.....	81
3.2.2	Software.....	83
3.3	NUmine Database.....	84
3.3.1	Relational Database Management System (R.D.B.M.S.)	84
3.3.2	Preliminary Truck Selection Example.....	86
3.3.3	Project Organisation.....	89
3.4	User Interface.....	90
3.4.1	Display of Information.....	90
3.4.2	Event Handling.....	91
3.4.3	Feature Selection	92
3.4.4	Error Handling.....	95
3.4.5	Data Entry and Editing.....	96
3.4.6	Windows	97
3.4.7	Multiple Windows.....	100
3.4.8	Help Facility	100
3.4.9	NUmine Screen Editor.....	101
3.5	Conclusions.....	104
Chapter 4	NUgraph - A Graphics Package	105
	for Computer Aided Mine Design	
<hr/>		
4.1	Introduction	106
4.2	Hewlett_Packard Graphics Library.....	109
4.3	Viewing information.....	111
4.3.1	Graphics Primitives.....	111
4.3.2	Pictures.....	117
4.3.3	Picture Manipulation	119
4.4	The Graphics Package.....	122
4.4.1	Ground Rules.....	122
4.4.2	Segmentation.....	124
4.4.3	Modelling.....	126
4.4.4	Interactivity.....	130
4.5	Conclusions	134

5.1	Introduction	136
5.2	NUsim Fundamentals.....	137
5.2.1	Compilation and Calculation	137
5.2.2	Random Functions.....	139
5.3	Network Modelling	141
5.3.1	Modelling the Passage of Time (Activities)	143
5.3.2	Routing Entities From Nodes (Branches)	144
5.3.3	Inserting Entities Into a System. (Enter Nodes).....	146
5.3.4	Queue Nodes.....	146
5.3.5	Continue Nodes	147
5.3.6	Deleting Entities from a System (Delete Nodes).....	148
5.3.7	Service Activities	148
5.3.8	Assign Nodes.....	149
5.3.9	A Three Shovel Example	150
5.3.10	Accumulation (Collect Nodes).....	151
5.3.11	Data Recording (Stats Nodes)	151
5.3.12	Decision Nodes.....	152
5.3.13	A Two Shovel Example	154
5.3.14	Match Nodes.....	156
5.4	Resources and Switches.....	157
5.4.1	Wait Nodes.....	157
5.4.2	Free Nodes.....	158
5.4.3	Alter Nodes	158
5.4.4	On Nodes.....	158
5.4.5	Off Nodes	158
5.4.6	A Single Shovel Example	159
5.5	NUsim Reports.....	162
5.5.1	How Statistics are Maintained	164
5.5.2	How Statistics are Reported (Report Nodes)	165
5.6.	Program Structure	168
5.6.1	Network Definition.....	168
5.6.2	The 'Initialise' Relation.....	169
5.6.3	The 'Entities' Relation	169
5.6.4	The 'Resources' Relation.....	170

5.6.6	The 'Variables' Relation.....	171
5.6.6	The 'Seeds' Relation.....	171
5.6.7	'Results' Relations.....	171
5.6.8	Simulation Controls.....	172
5.7	Conclusions	175
Chapter 6	Haulage Equipment Selection and Scheduling Module	176
<hr/>		
6.1	Introduction	177
6.1.1	Selection and Scheduling Problem - General Description	177
6.1.2	Program Objectives	179
6.1.3	Program Flow	179
6.1.4	Quarry Example	181
6.2	The Haul Route Network.....	183
6.2.1	Network Structure.....	183
6.2.2	Graphical Formulation.....	188
6.2.3	Integration with a 3-dimensional Pit Model.....	189
6.2.4	Additional Graphics Features.....	189
6.3	Truck Selection by Linear Programming.....	193
6.3.1	Route Haul Times.....	193
6.3.2	Validation of the Haul-Time Calculation Routine.....	196
6.3.3	Linear Programming Formulation.....	196
6.3.4	Two Types of Truck	201
6.4	Simulation Network Formulation.....	204
6.4.1	Converting the Route Network into a Simulation Network ...	204
6.4.2	Shift Patterns.....	206
6.4.3	Dispatching Algorithms.....	207
6.4.4	Dynamic Allocation	209
6.5	Canadian Test Examples.....	211
6.5.1	Quintette - Summary of Operations	211
6.5.2	Coal Mountain - Summary of Operations	216
6.5.3	Quintette LP Input.....	218
6.5.4	Quintette LP Results	223
6.5.5	Quintette Simulation Input.....	230
6.5.6	Quintette Simulation Results	230

6.5.7	Coal Mountain LP Input.....	236
6.5.8	Coal Mountain LP Results	236
6.5.9	Coal Mountain Simulation Input.....	236
6.5.10	Coal Mountain Simulation Results	239
6.5.11	Test Example Conclusions.....	239
6.6	Conclusions	242
Chapter 7	Conclusions and Recommendations for Further Work	243
<hr/>		
7.1	Introduction	244
7.1.1	The Need for Rapid Modelling	244
7.1.2	Correspondence with Reality	245
7.2	Recommendations For further Work	246
7.2.1	Modular Simulation	246
7.2.2	Expert Systems	246
7.2.3	Further NUmine Development.....	247
7.3	Conclusions	249
REFERENCES	250
APPENDICES	259

LIST OF FIGURES

Figure No.	Page No.
1.1	The NUmine System..... 10
2.1	Rimpull-Speed-Gradeability Curve for Caterpillar 769C Off-highway truck (after Caterpillar, 1980) 30
2.2	Brake Performance Curve for Caterpillar 769C Off-highway truck (after Caterpillar, 1980) 33
2.3	Estimated Future Production (after Morgan & Peterson, 1968)..... 34
2.4	Elements in a Haul Cycle which may Require Observation..... 36
2.5	Dispatching Board (after Mueller, 1977)..... 43
2.6	Automated Truck Control System (after Aiken, 1980)..... 45
2.7	Production Rate and Productivity at Sishen Iron-ore Mine (after Rossouw (1986) 49
2.8	Effect of Dispatching Below Saturation (after Cross & Williamson, 1969) 51
2.9	Haulroad Simulation Network Diagram (after Tu & Hucka, 1985) 54
2.10	The Relative Value of Adding One Shovel Near Saturation (after Tu & Hucka, 1985)..... 55
2.11	DISPATCH Overall Communications (after Farrell, 1988)..... 60
2.12	Cycle of dispatch Transactions for a Truck Operator (after Farrell, 1988) 62
2.13	Vital Signs Monitoring (after White & Olsen, 1986) 64
2.14	AVL Micro-based System Components..... 68
2.15	A 6-Node Example Showing all Feasible Paths..... 72
2.16	The Same 6-Node Example 'Optimised' 72
3.1	Storage Files and Relations..... 85
3.2a	Two Relations Associated With Preliminary Truck Selection 87
3.2b	The Two Relations Added 87
3.2c	The Combined Relation Sieved 87
3.2d	The Sieved Relation Customised..... 87
3.3	Performance and Retarder Curves Efficiently Stored and Merged on their Common Fields..... 88
3.4	Event Handling using the Keybuffer 92
3.5	Selection of File Operations Using a Simple Menu..... 93
3.6	Pull-Down-Menu for a Haulage Network Design System..... 94
3.7	Window for Editing Borehole Coordinate Data 98
3.8	Definition of a Database Field using a Variable Window 99
3.9	Principle of Reverse Nesting as Applied to Help Files.....102

3.10	Excerpts from Two Help Files used in the Haulage Network Design Example.....	103
4.1	The 4 phases of NUmine Graphics.....	108
4.2	Labelled Borehole Positions.....	112
4.3	Cut Diagrams.....	112
4.4	Text Justification.....	113
4.5	Boreholes with Contours Overlaid.....	116
4.6	HP Viewing Area Definition.....	118
4.7a	NUmine Display Area Definition	120
4.7b	Isotropic Window Definition	120
4.8	Graphics Relations.....	123
4.9	Graphics Primitives - Pointer Structures	125
4.10	Adding, Editing and Decoding Graphical Information	127
4.11	Graphical Modelling of 3-dimensional Surfaces	128
4.12	A Picture's Pointer Structure	129
4.13a	Typed Input.....	131
4.13b	Tracing the Position of a Line's End Point	131
4.13c	X Coordinate Constraint.....	131
4.13d	Relative Y Coordinate Constraint.....	131
4.13e	Angular Constraint.....	131
4.13f	Constrained Distance from Previous Point	131
4.14a	Modular Constraint - Grid	132
4.14b	Tollerancing to a Existing Graphical Items.	132
4.15	'Boxing'	133
5.1	Compilation and Calculation of a NUsim Function.....	138
5.2	Sampling from a Cumulative Frequency Curve.....	140
5.3	NUsim Network Example	141
5.4	NUsim Network Example (Manually Defined)	142
5.5	The Event Calendar.....	144
5.6	Routing Entities Example	145
5.7	Three Shovel Example Network	149
5.8	Two Ore Shovels Assisted by a Single Dozer.....	155
5.9	Branching to Matches from Queues	156
5.10	Single Shovel Loading Coal and Waste (with Shift Pattern)	160
5.11	A Typical NUsim Output Report.....	167
5.12	NUsim Trace Report	173

6.1	Haulage Analysis Flowsheet	178
6.2	Pull-down menu options in the Haulage Analysis Program	180
6.3	Quarry Example - 'Production' Relation.....	181
6.4	Quarry Example - Infrastructure and Road Plan	182
6.5	Quarry Example - 'Nodes' Relation	184
6.6	Quarry Example - 'Segments' Relation	184
6.7a	Quarry Example - Conventional Methods Node List	185
6.7b	Quarry Example - Conventional Methods Segment Links	185
6.8	Quarry Example - 'Routes' Relation.....	186
6.9	Quarry Example - 'Paths' Relation	187
6.10	Quarry Example - Valid Routes and Sub-Routes	182
6.11a	Create Haul Route Flowsheet	191
6.11b	Split Path Flowsheet.....	192
6.12	Route Profile and Preliminary Speed Considerations for Dp1-Ld3.....	194
6.13	Speed versus Time for Dp1-Ld3.....	195
6.14	Interpolation of Velocities at Critical Points	195
6.15	Segment Times for 6 Different Trucks hauling material along test route.	198
6.16	Segment Times for 6 Different Trucks returning empty along test route.	199
6.17a	Comparison of Calculated times, Model times and Manufacturer Specified times for Hauling	200
6.17b	Comparison of Calculated times, Model times and Manufacturer Specified times for Returning Empty	200
6.18	A 'Trucks' Relation.....	203
6.19	Operating Schedule for a Two-Shift Day	206
6.20	Conditional Branching from Dispatching Point	208
6.21	Subsidiary Network for Dynamic Allocation.....	209
6.22	Quintette - Principal Haul Routes	212
6.23	Quintette - Waste Haulage	215
6.24	Coal Mountain - Principal Haul Routes	219
6.25	Quintette - Coal Haulage	222
6.26	Comparison of Haul and Return times for 'Fixed' routes	224
6.27	Relationship between allocated capacities and times for optimal route selection with various truck combinations.....	228
6.28	Selection of best solution by cost.....	229
6.29	Quintette - Simulated Production for fixed allocation (one Wabco to each load point)	231
6.30	Quintette - Simulated Production for various truck combinations using the maximise shovels policy.....	231
6.31	Quintette - Simulated Production for various truck combinations using the maximise trucks policy	233

6.32	Quintette - Simulated Production for various truck combinations using the 'behind schedule' policy.....	233
6.33	Quintette - Simulated Production for various truck combinations using the 'match factor' policy	234
6.34	Quintette - Simulated Production for various truck combinations using dynamic allocation with factor (F) = 1	234
6.35	Quintette - Simulated Production for 5 Wabcos and 7 Terexs using dynamic allocation over a range of factors (F)	235
6.36	Optimum F for 5 Wabcos and 7 Terexs at Quintette.....	235
6.37	Coal Mountain - All Possible Waste Haulage Paths	237
6.38	Coal Mountain - Diagrammatic Representation of LP Solution to Waste Haulage	238
6.39	Optimum Damping Factor using 14 Caterpillar 777's at Coal Mountain where Factor (F) = 1	240
6.39	Optimum F using 14 Caterpillar 777's at Coal Mountain where Damping Factor = 33 secs	240

LIST OF TABLES

Table No.	Page No.
2.1	Distributions which can be Represented Using the Weibull 40
2.2	Summary of Standard Dispatching Policies 50
2.3	Increases in Utilisation and Productivity at Newlands as Predicted Using MINPLN (after Chatterjee & Brake, 1981)..... 52
2.4	Increases in Utilisation and Productivity at an Open-pit Copper Mine as Predicted Using MINEVL (after Chatterjee & Brake, 1981)..... 52
3.1	Preferred devices for various input activities (after Reid, 1984)..... 82
5.1	Priority Rules For Waiting in Queues147
5.2	Queue Overflow Facilities147
5.3	Saved Entities151
5.4	Queue Selection Rules.....153
5.5	Activity Selection Rules.....153
5.6	Activity Statistics.....162
5.7	Queue Node and Wait Node Statistics.....162
5.8	Stats Nodes Statistics.....163
5.9	Resource Statistics.....163
5.10	Entity Statistics.....163
5.11	Simulation Controls174
6.1	Test Haul Route Characteristics.....197
6.2a	Quintette Loading Equipment.....214
6.2b	Quintette Hauling Equipment.....214
6.3	Quintette Material Characteristics217
6.4	Quintette Production.....217
6.5a	Coal Mountain Loading Equipment.....220
6.5b	Coal Mountain Hauling Equipment.....220
6.6	Coal Mountain Material Characteristics221
6.7	Coal Mountain Production.....221
6.8	Quinette - Wabco Potential in a fixed regime assuming perfect dispatching.....225
6.9	Quinette - Terex Potential in a fixed regime assuming perfect dispatching.....226
6.10a	Required Trucks and Corresponding Objective Functions when in a closed-out configuration227
6.10b	Required Trucks and Corresponding Objective Functions when in a closed-out configuration227

LIST OF PLATES

Plate No.		Page No.
3.1	Hewlett-Packard 9000 Technical Workstation and Peripherals	81
4.1	NUMINE Graph Plotting.....	115
4.2	Maximising Screen Use using Multiple Graphics Viewing Areas.....	115
6.1	Coal Mountain - Highlighted Route and Profile.....	190

Chapter 1

Introductory Chapter

Chapter 1

Introductory Chapter

1.1 Introduction

It is generally recognised that at least 50% of a mine's operational expenditure can be attributed to the haulage process. This has not been restrictive in the past but as the technology required to mine lower grade deposits at a profit has developed, the resulting increase in the size of operations has been matched by a proportional rise in transportation costs. This has taken them to an unacceptably high level. In fact, if the trend is allowed to continue at the same rate, the life potential of the same marginal deposits will be diminished considerably.

As a result, there is currently a concentrated worldwide effort to reduce haulage costs. In some cases, this has meant the implementation of new developments such as in-pit crushing and rapid conveying. However, in many medium to large open pits, electric shovels and diesel electric drive haulage trucks are still in common use. They provide flexibility, can operate in confined spaces and, most importantly, they are an established technology. Despite these advantages, truck fleet productivity has the lowest improvement rate of all the alternative transport mechanisms and were it not for the degree of difficulty involved in converting from one haulage mode to another, many more mines would have made the switch.

There have, however, been some significant advances in at least one aspect of discrete haulage - that of truck dispatching. This thesis deals with the subject of the analysis and optimisation of truck selection and dispatching systems. The following sections briefly describe the main areas of study involved.

1.1.2 Truck Dispatching

Truck dispatching is the allocation of trucks to shovels on a regular basis throughout the course of a shift in order to satisfy the particular needs of a system in a more efficient manner. These include:

- the reduction of empty miles travelled through optimal route selection;
- the reduction of truck and shovel idle times;
- an improved response to pit disturbances and breakdowns;
- improved blending control;
- a reduction of shift-start hold-ups;
- automatic handling of refueling, maintenance and tyre management.

The importance of dispatching is such that the impact of implementing such a system must be considered at the planning stage. This involves the construction of a number of computerised models of the proposed, or existing transportation network. Their objectives are to determine the optimal truck fleet, the way it should be organised and to generally help satisfy the above requirements.

In the past, the time and energy spent constructing these models has been excessive. Each one is completely independent, designed to fulfil its own set of goals which contribute to the overall picture. Attempts have been made at simplifying the process by employing a common database (Brake & Chatterjee, 1979) and by the use of specific programming languages, but the model building process is, in general, too long and too complicated.

1.1.3 The Modelling Process

Jolley (1970) gives an overview of the various ways in which a system can be modelled. These are described below:

- Experimentation with actual facilities

This method provides very few inaccuracies but may cause disruption to production and may prove costly to carry out.

❑ Experimentation with a physical model

This method is useful where scale has little impact on the operation or where results can be extrapolated easily. However, it is time consuming, often expensive and sometimes difficult to achieve.

❑ Mathematical models

These can be implemented at much less cost and there is no disruption to production but analysis is confined to problems which can be defined by sets of equations and which are therefore, by definition, predictable.

❑ Simulation

Simulation, which is an attempt at predicting the performance of a complex system by synthesising what is actually taking place, provides the engineer with the ability to analyse any process, even one which cannot be easily expressed mathematically. However, in general, simulation is also a time-consuming process.

1.1.4 Computer Aided Design

Computer Aided design is a self-explanatory term which can refer to any planning process using computers. Historically, it has been used solely to describe the graphical representation of objects on a computer screen. Today the role of drawing in CAD is changing with greater emphasis placed on the way it conveys its message and on the way it can be combined with a CAD database to form the core of a totally integrated design system. (Medland & Mullineux, 1988).

A complete CAD package is envisaged here as the ideal framework for the rapid development of haulage network models. The rationale behind this assumption can be summarised with three criteria in mind, namely flexibility, integration and ease of use. If these exist, the model building process will be simplified greatly.

1.1.5 Objectives

It would be impossible to fully understand the objectives of this thesis from the limited introductory information documented so far. However, they are described here in an attempt to clarify the proposed structure of the work. Their actual meaning should become more clear after further reading.

- The first objective is to demonstrate how relevant, selected modules of an integrated computer aided mine planning and design system can be used to simplify and shorten the process of analysing truck/shovel haulage systems.
- The second is to determine, using simulation, the extent to which 'optimum' solutions derived using mathematical (linear programming) models can be met in real time by the application of dispatching policies.

The first involves the semi-automatic creation of mathematical and simulatory models in order to derive and compare solutions to the haulage problem. The second is of crucial importance when attempting to minimise the number of solutions which need to be carried through to the final stages of analysis (those which involve simulation) again due to the time consideration.

1.1.6 Thesis Format

The format of the thesis attempts, as far as possible, to reflect the quantity of work carried out in each relevant area.

This chapter began by introducing some of the basic terminology associated with the problem under a series of sub-headings. The following section gives an overview of a computer aided design system of which this work forms part (The author has, in fact, been involved in the development of various aspects of the overall system). It then goes on to discuss simulation modelling in greater detail, analyses the various simulation languages which are available and looks at some of the systems which have been applied in the broader area of mine transportation. Chapter 1 ends by suggesting some of the attributes of a more efficient, alternative modelling environment.

A detailed breakdown of each chapter's format will be given in their respective introductions. A brief description of each follows:

- ❑ Chapter 2 reviews the work carried out thus far into truck/shovel control and dispatching techniques;
- ❑ Chapters 3,4 and 5 go into greater detail about specific aspects of the computer aided design system;
- ❑ Chapter 6 describes the way in which these have been utilised in the truck/shovel analysis problem then goes on to examine two operational 'case studies' with reference to the above objectives;
- ❑ Chapter 7 provides conclusions and makes recommendations about future work which may be carried out in the same area.

1.2 NUmine

Mine planning involves the development of a number of conceptual plans for the extraction of a mineral deposit which are then examined comparatively in order to determine the most financially viable. This sounds straight forward but, in fact, conventional project evaluation of this type involves:

- the assembly and manipulation of a considerable quantity of diverse and uncertain information.
- the generation of numerous geological, engineering and representative drawings.
- a substantial amount of numerical calculation - often repetitive.

These are all jobs which today's computers are capable of performing particularly well - a fact reflected in the quantity of planning packages which have been developed worldwide since the 1970's. However, most concentrate on one particular aspect of the design process to the exclusion of the others. Also obvious is that they have been specifically designed to run on just one of a wide range of computers, operating systems and languages.

As a direct consequence, a substantial amount of time and energy is spent modifying software and manipulating data to tie in with existing systems. In addition, when confronted with a new program, a mine planner, who may have very little computing experience, is confronted with a completely new operating environment. This reduces his productive capacity in the short term at least and may leave him unwilling to return to the program the next time.

1.2.1 NUmine Philosophy

In 1986, the University of Nottingham's department of Mining Engineering recognised that substantial benefit could be gained from a completely integrated planning system covering the full spectrum of mine design activities ranging from the input of exploration data to the financial evaluation of detailed, alternative solutions - A 'Total' system. The department could draw on a considerable background of computing experience with research up to this time having been carried out in a number of specific areas - in particular, surveying, rock mechanics and mine environmental engineering. One of the aims was to incorporate this work into the overall system.

In more general terms, the principle objectives were:

- ❑ to cover the complete range of planning activities associated with the underground or open-cast extraction of any mineable material;
- ❑ to ensure complete integration between each of the system components;
- ❑ to allow for multi-level development at different times and to different degrees of sophistication using a modular approach;
- ❑ to use more informative graphical rather than numerical representation wherever possible;
- ❑ to make the system robust and easy to use;
- ❑ to incorporate dynamic, interactive data manipulation;
- ❑ to pay particular attention to the uncertainty which is inherent to a large proportion of mine planning processes;
- ❑ to try to minimise the memory requirements of each application so that the system is able to run on the micro-computers for which it was designed; and
- ❑ to give the user ultimate control over the planning process - not the computer. In this way the planner retains a greater understanding of the work carried out and is able to contribute more to those elements which still require human intellect and experience.

Further important considerations were that a great deal of planning work is actually carried out while the mine is operational and, on an academic level, that the system should enable research into the application of new computing technology to various aspects of mine design.

1.2.2 NUmine Structure

The modular configuration underlying the whole system was envisaged by Ferguson (1985). It consists of a central database surrounded by a range of utility software units designed specifically for mine planning applications (Fig 1.1). These include the database management system, a fully integrated graphics package, and a user interface module for robust and user-friendly, interactive communication. Peripheral to all of these are the mining application programs themselves. In Figure 1.1, these have been grouped under broad headings which follow, in a clockwise direction, the usual planning cycle. In reality, the boundaries between some of the groups are difficult to define and the development of individual modules has followed no particular pattern.

Some of the areas being covered are listed below:

- Exploration and Survey: Borehole Analysis,
Automated Surveying,
Automatic Digitisation,
- Geological Modelling: Surface Modelling,
Solid Modelling,
- Reserve Estimation: Geostatistics,
Cut-Off Grade Analysis,
- Geotechnical Analysis: Slope Stability Assessment,
- Mine Design: Pit-Limit Algorithms,
Production Optimisation,
(based on all of the above)
- Production and
Equipment Scheduling: Truck/Shovel Analysis,
Dragline Selection,
Longwall Simulation,
Room and Pillar LHD simulation,
- Environmental Analysis: 3-D Perspective Visualisation,
Ventilation,
Methane Drainage,
- Financial Appraisal: NPV / DCFROR Calculations,
Risk Analysis.

A more detailed description of the core modules, in particular the user interface, follows in Chapter 3. Personal involvement with the development of the graphics system and its relevance to the analysis of mine transportation systems has necessitated a chapter for this alone (Chapter 4).

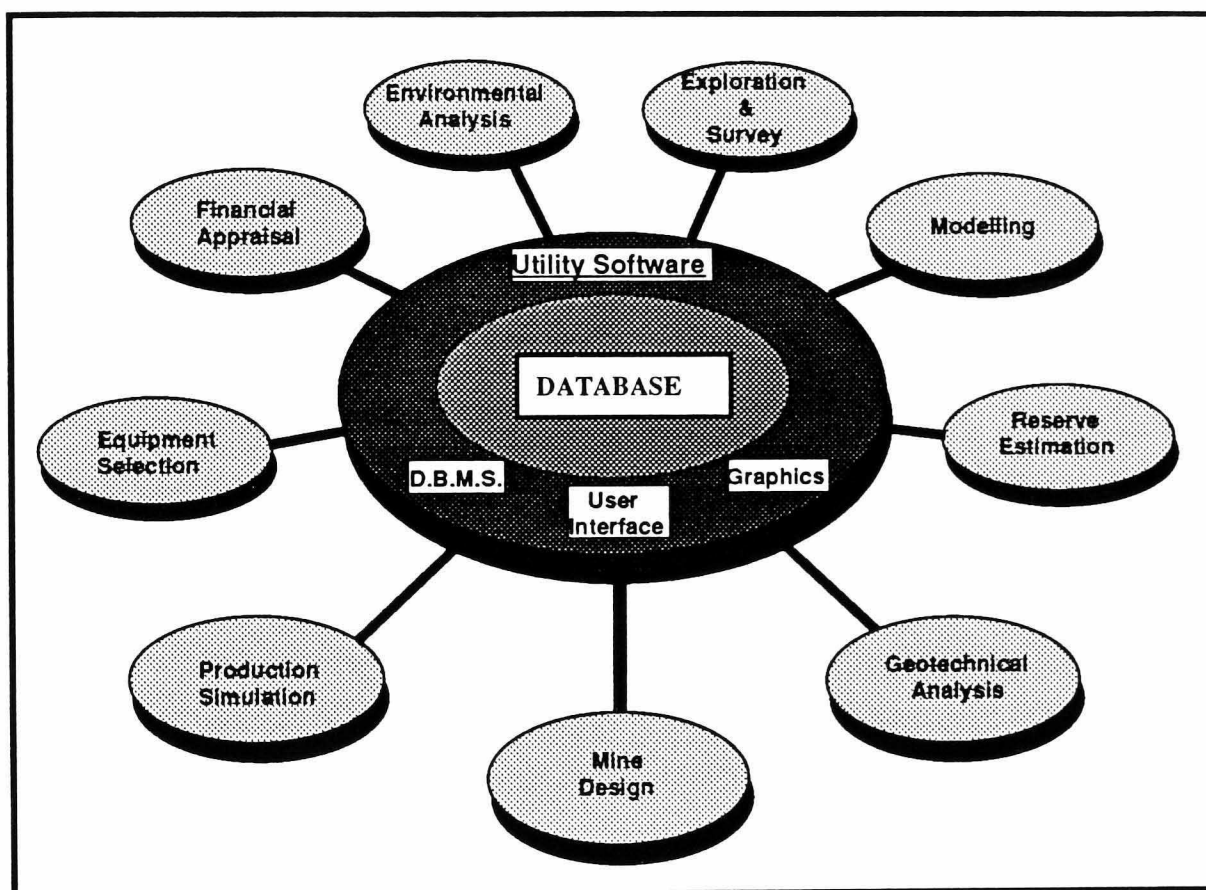


Figure 1.1 The NUmine System

1.3 Simulation Modelling

Simulation has been defined as the representation of the dynamic behaviour of a system by moving it from state to state in accordance with well defined operating rules (Pritsker, 1979).

The areas in which simulation has played a significant role for almost 30 years have been

- Air traffic control
- Manufacturing system design
- Financial forecasting
- Communication/Distribution/Information system design
- Transportation
 - traffic light timing
 - commercial distribution systems

The principle underlying each of these applications is that by simulating a number of intuitive solutions it is usually possible to determine which is the most feasible, economic or practical.

The main advantages of simulation are: its flexibility, the ease with which it can be applied to virtually any model; its insight, the way it provides the user with a means of visualising and comprehending the workings of a process; and its adaptability, a favourable comparison between the results of an existing system and those obtained using a simulation model of the system, however simple, guarantees that the model can be realistically adopted to examine a new but similar experimental process.

1.3.1 Stages in the Modelling Process

Pritsker and Pegden (1979) describe 9 stages in the modelling process but emphasise that it is first important to ensure that simulation is actually required and that no simpler, alternative method can be used equally well.

1. Problem Formulation

This includes specification of the types of design alternatives which are to be evaluated using the model as well as of the model itself.

2. Model Building

Defining the structure of the model is usually the simplest task but can be the most time-consuming.

3. Data Acquisition

A model is only as good as the data it uses. A considerable effort must be made to ensure that the collected data is representative of the process being modelled.

4. Model Translation

Preparation of the model for processing (programming)

5. Verification

Establishing that the program executes as it should

6. Validation

Ensuring that the desired accuracy of correspondence exists between model and process

7. Strategic and Technical Planning

The process of establishing an experimental solution

8. Analysis of results

Analysing simulation outputs to make inferences and recommendations.

9. Implementation and Documentation

1.3.2 Simulation Categorisation

Simulation modelling processes can be roughly subdivided into two categories - discrete and continuous.

❑ In discrete modelling, the state of a system remains constant between the occurrence of instantaneous events. The model can adopt either an event orientation where a calendar of events is maintained and the steps associated with each are implemented as they occur, an activity scanning orientation, where events are initiated or terminated according to prescribed conditions or a process orientation where sequences of events can be generalised into a single statement.

❑ In continuous modelling, the state of certain attributes within the system vary continuously. The model is constructed from state and derivative variables which are integrated at run-time by splitting each process into small discrete steps.

In reality, many problems contain aspects which are both discrete and continuous in nature. Several simulation packages offer both types in combination to facilitate this.

Another categorisation differentiates deterministic from probabilistic simulation. A model can only be described as deterministic if, for every value of input, the output can be determined in some reproducible fashion (Bobillier *et al*, 1976). In many cases, a certain amount of uncertainty will be associated with the outcome and probability factors need to be considered. The latter will become evident throughout the rest of this thesis.

1.3.2 Simulation Languages

The proven advantages of simulation as an analytical technique have resulted in the development of a wide variety of simulation languages which are now in use. In general, these provide for the automatic sequencing of events, the implicit definition of entities and the automatic accumulation and printing of statistics - facilities which allow the user to concentrate on model development without having to worry about the program's more 'clerical' tasks (Adam & Dogramaci, 1979 (a) & (b), Pritsker & Pegden, 1979). Brief descriptions of the main ones follow:-

GPSS

Type: process oriented discrete

Base: Fortran

The General Purpose Simulation System (GPSS) is probably the best known and most widely used of the simulation languages. Its main benefit is its simplicity. A model is constructed by combining a set of standard facilities, of which there are at least 40 types, into a block diagram which defines its logical structure. Transactions passing through a facility are processed one at a time to simulate a small part of the overall process.

Its simplicity is also one of its major disadvantages. Its limited computational ability and its rigidity makes the formulation of a complex model difficult (Bobillier *et al*, 1976). It is also integer based which means that frequently events occur simultaneously. These must be separated by giving each transaction a priority number.

GPSS-PC, a micro-based version with graphics facilities, is now available (Sturgul, 1987). NGPSS is a derivative with continuous capabilities.

Simula Type: process oriented discrete Base: Algol

Models created using Simula are put together as a collection of defined processes. During run time, each process can be in one of four states namely active, suspended, passive or terminated and these are altered as a means of simulating the events taking place in the real system. The results are obtained simply and inexpensively.

Simscript Type: discrete / combined Base: Algol, PL/1

Simscript models are written in a manner similar to English which makes them very easy to understand (Markowitz, 1979). Variable names and different entity types may be specified to enhance this desirable property. The language is available in 5 levels:-

- level I is a simple teaching language,
- level II contains statements which are similar in power to fortran,
- level III contains statements which are comparable to Algol/PL1,
- level IV provides a framework for modelling using entities, attributes and sets.
- level V allows for automatic time advance, event processing and accumulation and analysis of simulation generated data.

The modelling environment is truly general purpose. i.e. the number of entities or specific sub-processes may be input as parameters.

GASP IV Type: combined Base: Fortran, PL/1

The General Activity Simulation Program (GASP IV) is a derivative of GPSS which is able to handle both time and state events (Pritsker, 1979). The system is flexible - transactions may become grouped together in sets referred to as files and partial differential equations as well as logic, memory and generator

functions are available.

The language's main objective is to provide commonly used sub-programs but to force the correct organisational structure on to the user. In this way, subsets of the overall process can be run on their own and expanded later. In addition, names of variables can be made to correspond with the process being modelled and interactive decisions can be made by the user during a simulation run. However, studies carried out in the USA by the Aeronautical Research Laboratory (Chatterjee & Brake, 1981) showed that GASP was 4 times slower than Simula and took up 4 times as much memory.

Q-GERT

Type: network combined

Base: Fortran

The Graphical Evaluation and Review Technique (Q_GERT) is a GASP based language which attempts to model a process using a graphically formulated simulation network. There are only 10 node types but functions can be added or combined at single nodes. Resources can also be incorporated.

SLAM

Type: network combined

Base: Fortran

SLAM, a Simulation Language for Alternative Modelling, provides a unified modelling environment which combines the ease of network modelling obtainable using Q-GERT with the flexibility of GASP IV (Pritsker & Pegden, 1979). Discrete events can be modelled with an event or process orientation and activities may be scanned continuously.

6 specific interactions can take place between the various world views of SLAM:-

- entities flowing through the network can initiate a discrete event;
- events can alter the flow of entities in a network;
- entities flowing through the network can cause instantaneous changes to state variables;
- state variables reaching prescribed thresholds can initiate entity movement;
- events can cause instantaneous changes to state variables;
- state variables reaching prescribed thresholds can initiate events;

Continuous Languages

A standard for continuous languages has been developed by the continuous system simulation language committee (C.S.S.L). Current languages which can legally be referred to as CSSLs include : CSMP, the continuous system modelling program; ASCL, the advanced continuous simulation language; CSSL III and IV; and DARE_P, which was written entirely in ANSI Fortran IV and is therefore extremely machine dependent. All these languages are similar with the main differences being the format of their input and output operations (Korn & Wait, 1978).

Others include MIMIC which was specifically designed for CDC computers and Dynamo (Lehman, 1977) which is really only suitable for continuous models that can be represented mathematically.

Other Simulation Languages

- Simpl/1 A discrete simulation language based on PL/1
- SOL A symbolic language for general purpose simulation
- ASPOL (Simulation Process-Oriented Language)
- INS (An Integrated Network Simulator)
- BOSS (Burroughs Operational System Simulator)
- GPDS (General Purpose Discrete Simulator)
- GPS-K (General Purpose Simulation - K)
- AS an algol simulation language (Buxton, 1968)
- Flow Simulator
- SEAL (Simulation Evaluation and Analysis Language)
- SLANG A system aimed at simplifying the simulation process at the design stage (Buxton, 1968)
- COSMO A continuous modeller (Buxton, 1968)

1.4 Simulation in the Mining Industry

The mining industry has been relatively slow to recognise the importance of simulation as an analysis tool. Nevertheless, there have been a number of successful applications in specific areas of production and transportation. Most of the following models were developed from scratch, without the use of a simulation language. They illustrate some of the techniques which can be utilised and some of the benefits which can be achieved.

The simulation of truck/shovel systems is omitted from this section since Chapter 2 deals with the subject exclusively.

1.4.1 Production

Suboleski and Lucas (1968) recognised that it was becoming increasingly difficult to follow a structured plan in room and pillar operations due to the amount of interaction involved. A simulation package was developed to model waiting times and breakdowns using random functions. The model was found to be useful in many other areas namely evaluating mining methods, designing section layouts, evaluating new equipment, justifying capital expenditure and training management personnel.

Touwen and Joughin (1972) suggested that the application of mathematical methods such as linear programming and queueing theory had only been possible in a limited number of simple cases. They developed two procedural models, one for underground stoping operations and one for transportation, using discrete, event-orientated simulation. Items considered were the expected duration of operation, the probability and duration of a failure or mishap and the consequences of the interruption.

Facesim, developed by the Virginia Polytechnic Institute (Hanson & Selim, 1975) uses the event-oriented method to simulate both conventional and continuous coal mining activities. Hanson goes on to describe a model built at the Twin Cities Mining Research Centre aimed at improving the utilisation of current mining technology, investigating equipment schedules and examining the economic impact of implementation proposals.

Haycocks *et al* (1984) describe FRAPS a program similar to Facesim but more geared towards user-friendliness. Attention is paid to the quality of input and output facilities and to the size of field equipment. Two versions exist, one for

conventional room and pillar operations and one for continuous mining. LHDsim, a combined stochastic and deterministic simulation package designed to simulate face activities and production using load-haul-dump units, is also mentioned.

Katsabanis *et al* (1984) developed NTUsim, an LHD simulation package for use in Greek underground bauxite mines. The system is event oriented and deterministic but its main benefit is the ease with which entries can be modified. This is important in an area where tectonic irregularities frequently cause the layout to differ from the theoretical ideal.

1.4.2 Locomotives

In Germany, during the 1970's, there was plenty of scope for the application of simulation (Wilke, 1973). It was becoming common practice to link two or more mines together thereby increasing the overall transportation distance. A small program based on the fundamentals of GPSS was developed in an attempt to identify problem areas and hence minimise production loss and transportation costs. An additional advantage of the system was the ability to test out policies relating to the selection of train destinations - a problem analogous to truck dispatching. One general conclusion drawn was that the cost benefit of being able to update decisions en-route decreases as the sophistication of the initial decision policy increases.

A similar situation exists in China (Zhongzhou, 1984). Large depths and long haul distances combine to make the formulation of mathematical models very difficult. Two systems, one time-oriented the other event-oriented were implemented at different mines. The Algol 60 programs included route matrices for making junction destination decisions.

At Climax Molybdenum (Steiker, 1982), simulation was used to determine track layout, crusher specifications, train lengths and numbers and dispatch procedures. A generalised system was considered too cumbersome to be useful so the models were developed using GPSS. Three different track systems were tested and the means and standard deviations obtained were found to fall within 95% confidence limits. As expected, variations increased with the number of trains simulated.

At Impala (Wilson, 1984), simulation was preferred to queueing theory and linear programming as a means of analysing surface rail operations because it was easy to understand. Having spent a considerable time collecting data, GPSS was applied in deterministic mode due to the number of runs which would have otherwise been required for statistical accuracy. Several recommendations were

produced including the better training of drivers and the appointment of a train controller. The GPSS reporting facility proved extremely useful.

Discrete and combined simulation was successfully applied in the design of a computerised control system for rail haulage in a horizontally cut mine (Holzman & Haefner, 1978) and at Kiirumavaara iron ore mine in Kiruna (Oberg, 1980). Fitzgerald and Blair (1977) describe a deterministic model on a mini-computer for a single track iron ore railway operation and simulation has also been used to analyse the transport of materials by locomotive (Hancock & Lyons, 1984).

1.4.3 Conveyors

In 1968, Sanford and Manula (Talbot, 1977) were among the first to recognise the potential of continuous simulation in the analysis of conveyor systems. They developed a system known as Beltsim, which is mainly concerned with the detrimental effects of breakdowns. Typical examples are when there is a malfunction in a feed or discharge arrangement and during the replacement of idlers. The system derives an estimate for the material lost and helps the engineer consider the financial implications of installing surge bins etc.

British Coal Operations Research Executive have been using stand-alone simulation for the design of longwall conveyor systems for at least 20 years (Price & Szabo, 1970). Pressure on the transportation system due to the move towards fewer faces of longer length resulted in the development of Simbelt. Originally, three Fortran programs were available:

- ❑ Simbelt 1, to calculate the total bunkering requirement;
- ❑ Simbelt 2, to estimate the clearance potential of various belt/bunker configurations, and
- ❑ Simbelt 3, for the analysis of new roadways.

Later (Hancock & Lyons, 1984), Simbelt 4 replaced Simbelts 1 and 3 by being able to analyse face patterns and the conveyor network at any mine. Based on the results achieved, bunkers could be added using Simbelt 2 which now included feedback i.e. the ability to recognise that when a bunker is full, all the belts which feed it must stop.

Simbelt is now available on IBM PC micro-computers. The 'M' version uses colour graphics and gives the user interactive control over the simulation process.

Simbunk (Ryder & Szabo, 1976), was designed to test the behaviour of a conveyor system operating under central control. More specifically, the effectiveness of decision rules applied by the manual controller. Examples are: under what conditions shall a bunker be discharged and at what rate?; and should a belt be stopped to give priority to another district and, if so, when?. The considerable amount of input and output involved demanded the extensive use of help screens and an efficient means of visualising the current situation. By 1984 (Hancock & Lyons), colour coding had been introduced in what could now be considered an interactive graphical environment.

1.4.4 General Systems

The rapid advance of computing power has given rise to the development of a number of more general simulation systems throughout the mining fraternity. Ryder (1976), was concerned with the low rates of production being achieved on working faces in South Africa's deep level hard rock mines. The problems of congestion and overstrained surge capacities were worsened by the dynamic nature of the operations themselves. Transim was designed not only to model the locomotives which were central to the whole problem but also the boxhole pulling schedules, tip and bunker configurations and the transportation of men and materials. Shortest route selection and train capacity calculations were intrinsic to the event-oriented system but the 2 weeks of preliminary field observation and 4 man days of computing effort required to set up a single exercise were major drawbacks.

Pennsylvania State University's Underground Materials Handling Simulator is also a 'total' systems model in that any mining process can be simulated to any level of detail (Manula *et al*, 1975). It works on the process oriented basis that in order to carry out an activity, a particular set of resources must first be available. It is also evolutionary and can be added to by experience. Specific study areas have included the relationship of system design to accidents and hazards, and the design of operational control mechanisms.

1.4.5 Language Comparisons

There have been several attempts at assessing the relative merits of simulation languages as opposed to conventional languages in mining. We have seen how GPSS in particular has been used as a time-saving instrument on a number of occasions. Its fundamental simplicity has led to its general acceptance as the language to opt for and, in times of difficulty, there will usually be someone with a knowledge of GPSS to consult. But what about the practicalities?

Fytas *et al* (1985) used Transim II (Ryder, 1976) which is a Fortran based system and GPModel which is a shovel and truck simulator written in GPSS to analyse the same haulage network. They found GPSS to be readable and simple to learn and use. They found it reduced project time - a comparison of the length of code (Fortran 4500 lines, GPSS 870 lines) is indicative. It was also easy to debug and data recording and compilation was automatic.

However, compared with Fortran, it lacked flexibility, it produced unrelated output consisting of columns of figures from which the relevant information had to be extracted and its fixed memory requirement was high, being independent of the size of the model under examination. In addition, GPSS models have to be reprogrammed whenever a modification is required and sub-routines cannot be added in a form compatible with the block structure. On a specialist note, these make the modelling of dispatching policies very difficult.

These same conclusions can be applied, in varying degrees of magnitude, to conventional and simulation languages in general.

Though the disadvantages of simulation languages appear to outweigh their advantages, Sturgul (1987), reiterates the time factor and emphasises that Fortran is a sequential language whereas many events in a mining process occur simultaneously. Sturgul and Singhal (1988) attribute the reluctance of the mining industry to utilise simulation languages to the fact that they must be learnt and that they are expensive to install. They then advocate the use of GPSS PC for its graphical capabilities and its relatively low cost. They estimate that a model involving 2 shovels and 3 dumps could be set up in as little as half an hour.

One other general recommendation was that simulation languages should be used except in cases which involve a large number of sequential statements. Here, conventional languages may be more appropriate.

1.5 Conclusions

Whatever language is used, the development of simulation in the mining industry is currently limited by a number of general factors. First, it is usually the case that models have to be developed from scratch, without reference to any existing information which, if available, would make the task much easier. Second, both the input to and the output from the simulation model take a rigid numerical format. There is, in other words a distinct lack of graphical visualisation. Third, there has been little regard to the quality of human interaction which can be achieved in the development process.

While the last two factors apply less to the aforementioned specific applications, these are, by definition, the systems which take the least account of external factors in the form of existing data. Of these, Profitas (Vagenas, 1988), a system looking at the automisation of LHD vehicles, is one of the few to make use of graphical simulation within an integrated modelling framework. The system is menu-driven, sub-models can be run independently and good quality graphical output can be obtained automatically. These are some of the features which have been built into NUmine's haulage analysis system.

Before continuing, it is worth considering some of the dangers of using simulation. Its potential for example, may be negated by over-elaborate programming. At the same time, too much simplification in the form of sweeping assumptions can lead to incorrect results. The second danger area is validation. A model which is inconsistent with the rules of the actual system may 'work' but will produce spurious results. Validation is highly dependent on the purpose of the project and no scientific theory exists with which to guarantee the validity of the model. Instead, validation must be carried out by comparing the results obtained with observations from a similar real system.

Automated network generation ensures that the simplest techniques are used and allows specific problem areas to be analysed quickly in a standardised manner. It cannot guarantee absolute correspondance to an existing system but, if the flexibility of a language is also provided, alterations can be easily made.

Chapter 2

A Review of the Development of Truck/Shovel Simulation and Dispatching Systems

Chapter 2

A Review of the Development of Truck/Shovel Simulation and Dispatching Systems

2.1 Introduction

Haulage trucks were first introduced into surface mines in the early 1930's (Singhal & Fytas, 1985). They were generally small units (weighing approximately 15 tons) and remained so until further development of mechanical drive systems in the late 1950's at which time 35 to 50 ton trucks became available. During the 1960's rapid advances in the technology of high speed, 1600-2000 hp diesel engines combined with electric wheel transmission resulted in the introduction of 85-100 ton diesel electric trucks and the period culminated in the wide-spread industrial acceptance of the 170 ton (154 tonne) diesel electric which is utilised extensively today. Only the largest mines have been able to justify the implementation of trucks with a greater capacity than this. Elsewhere, the cost and the potential loss in flexibility have been inhibitive factors.

Whatever the scale of operations, it is essential that the correct decisions regarding the selection of equipment are made and that the best use is made of the available fleet. Wilke and Heck (1982) outline the stages in the selection process which constitute the evaluation of a truck/shovel solution to the haulage problem:

- determine the 'optimal' fleet size
- set up an efficient organisational system
- implement a reliable maintenance and servicing system.

In the equipment selection process, the evaluation of shovels comes first since truck performance is influenced more by the choice of shovels than vice-versa (Crawford, 1979). Because of the variables involved, estimation of truck productivity is probably the most complex portion and the one discussed in most detail throughout this chapter. However, in reality, the aim is to produce the lowest cost for the product which may mean re-evaluating the system from scratch, i.e. altering the blast design as well as the shovel selection decision.

Conventional fleet size determination, based on a combination of calculation and experience, fails to sufficiently account for the uncertainty involved in mining operations. Computer simulation overcomes this and has been applied extensively not only to find the best truck for a particular load-haul-dump configuration, but also to test out the probable effects of changes in the cycle or the introduction of a new system. Hence, a method for the critical examination of a mine's organisational structure became available. The main limitations to this were that the operation must be known in sufficient detail and that, due to the ever-changing nature of a mine's structure, this data must be kept up to date.

New control mechanisms were investigated with a view to keeping track of equipment status and location. Two-way radio communication allowed for the more efficient assignment of trucks to shovels in order to meet with specific shift quotas and pre-defined stripping ratios (Naplatanov *et al*, 1977). It also provided a means of dealing with changes in loading and hauling rates and, in scheduled maintenance or emergency breakdown situations, changes in machine availability. Once this technology had advanced to the stage where trucks could be reallocated to specific shovels at any point in time, the 'which shovel' decision became significant and investigations into the potential value of various dispatching policies accelerated.

This chapter deals with the conventional techniques used in equipment selection and shows how they can be used to predict cycle times (Section 2.2). Section 2.3 then describes how simulation together with the above information can and has been used to establish preliminary fleet sizes. In Section 2.4, truck monitoring and control mechanisms are introduced to show how, if necessary, dispatching decisions can be made and implemented. Section 2.5 looks at the way dispatching policies can be analysed using simulation.

Automatic dispatching systems have been installed at several mines worldwide. Section 2.6 describes these and Section 2.7 introduces some of the advanced techniques associated with them.

2.2 Selection of Haulage Equipment

The selection of haulage equipment to serve a set of loaders depends on a number of factors :-

- The capacity to handle projected quantities.

Selection decisions are based primarily on production requirements which may be lower to start with than the ultimate plan (Kullberg & Wright, 1968). Truck productivity depends, to a large extent, on the time it takes to complete a haulage cycle. Due to its complexity, estimation of the cycle time has been the subject of a large number of computer projects.

- The match with loading equipment.

The steady increase in the size of excavation equipment has resulted in a necessary increase in the size of haul units (Atkinson, 1982). As well as obvious physical considerations such as making sure that the shovel's loading height exceeds the height of the truck, the ratio of dipper capacity to truck capacity must be such that the truck can be filled in less than 7 cycles, thus avoiding excessive load times, and in a minimum of 3 to eliminate shock loading.

- The suitability to haul mine material.

This may be the most important factor in the selection of truck body configuration i.e. whether conventional rear dump trucks or tractor trailer units with rear, side or bottom dumping are to be used. However, other factors such as manoeuvrability and the loading/dumping configuration may also need to be considered.

- Physical conditions

Apart from their affect on cycle time, haul distances, road grades and rolling resistances may place a limitation on particular truck types, reducing their operating efficiency to an unacceptable level.

- Service and maintenance requirements.

Kullberg and Wright (1968) estimated that a machine availability of between 65 and 75 percent must be allowed for preventative maintenance i.e. servicing, refuelling, tyre changes and major component repairs. This figure gradually increases to between 75 and 85 percent during the first 2 years of a machine's life with a 2% reduction in availability every year thereafter. However this can only be achieved by increasing the size of the fleet with a proportional reduction in the mean size of the individual haul units (Crawford, 1979).

- ❑ Projected availability and utilisation of equipment.

When these values are taken into consideration, the calculated fleet size will probably be less than 80% of that required in reality.

- ❑ Capital and operating cost.

The net present value of cash flow must be calculated for each fleet proposal in order to arrive at a final decision. The calculation of operating expenditure is an area of technology which has lagged behind since it depends, to a large extent, on the interaction between the loading and hauling units. (Bohnet, 1984).

- ❑ Experience.

Valuable knowledge can be gained from equipment being used in similar circumstances elsewhere. When expansion of existing operations is to take place, an understanding of the current situation can be used to predict the variability of cycle times and the effects of interaction in any future proposals (Morgan & Peterson, 1968).

- ❑ Future developments

Haul distances, grades etc. will vary throughout the life of a mine and this must be taken into consideration at the fleet planning stage.

Singhal and Fytas (1985) give a more detailed breakdown of the factors to be considered in truck selection and the desirable specifications required for the selection process to take place.

2.2.1 Cycle Time Calculations

The number of trips that can be completed by a truck per hour is a function of the time taken for the round trip from loader to dump and back again. Estimation of the haulage cycle time based on truck performance and haul profile conditions are therefore fundamental to the planning process (Crawford, 1979).

A complete cycle can be broken down into 6 elements :-

1. Loading time
2. Time to travel loaded to the dump
3. Dumping Time
4. Time for return journey

5. Manoeuvring time at the loader
6. Delays

The non-travel elements are generally assumed constant for a given shovel-truck combination. Loading time, for example, can be calculated from the following equation :-

$$T_l = C_t / C_{sd} \times T_{ss} \quad \dots (2.1)$$

where:

- T_l = Load Time (secs),
- C_t = Truck Capacity (m^3),
- C_{sd} = Shovel Dipper Capacity (m^3),
- T_{ss} = Shovel Swing Time (secs).

The haul and return times, however are affected by a considerably larger number of variables and as such are the subject of numerous computer simulation studies (Atkinson, 1982).

O'Neil and Manula (1967) were among the first to appreciate the value of being able to realistically duplicate truck movement using standard or deterministic simulation and included a relatively simple subroutine for the calculation of haul times in their open-pit materials handling model (Section 2.3.2). While the methodology for manually calculating the performance of an automotive vehicle has long been known (Caterpillar, 1968), their recognition that if acceleration over a short period of time is assumed constant then velocity and speed relationships can be reduced to familiar algebraic equations has been re-used extensively since :-

From Newton's second law of motion, $F = ma$, the instantaneous acceleration of a vehicle

$$a_i = (F_2 - F_1) / m_t \quad \dots (2.2)$$

where:

- a_i = instantaneous acceleration (ms^{-2}),
- F_1 = Resistance to motion (kN),
- F_2 = Force exerted by truck (kN),
- m_t = mass of truck (Kg)

A breakdown of how F_1 and F_2 are obtained during the simulation process follows:

Resistance to Motion (F_1)

The 3 forces which act against the movement of a load are :-

Rolling Resistance

The force of the ground against the vehicle wheels.

Air Resistance

The force acting against motion by the surrounding atmosphere.

Grade Resistance

The retarding force of gravity which must be overcome when travelling uphill (or the assisting force, downhill).

Grade resistance, which is equivalent to $mg \sin\theta$ where θ is the gradient of the slope, and rolling resistance can both be expressed as a percentage of gross vehicle weight. Rolling resistance and air resistance, meanwhile, are speed related but at speeds less than 60 km/hr, variability in air resistance is negligible. Hence, for experimental simplicity, the two can be considered as a single property.

Taking 10 kgF/tonne of rolling resistance to be approximately equal to 1% of grade resistance, effective resistance per tonne,

$$F_1' = (R_{r+a} + R_g) \times 100\% \quad \dots (2.3)$$

where: F_1' = Resistance to motion (kN per tonne)
 R_{r+a} = Rolling + Air Resistance (% of GVW),
 R_g = Grade Resistance (% of GVW).

Force Exerted by the Truck (F_2)

The power available to overcome the effective resistance and accelerate the load F_2 can be obtained by interpolation from a digitised version of the manufacturer's performance chart of available rimpull vs speed (Fig. 2.1).

F_2 is equivalent to the usable rimpull which may be limited by the traction available. This is a function of the weight on the drive wheels:

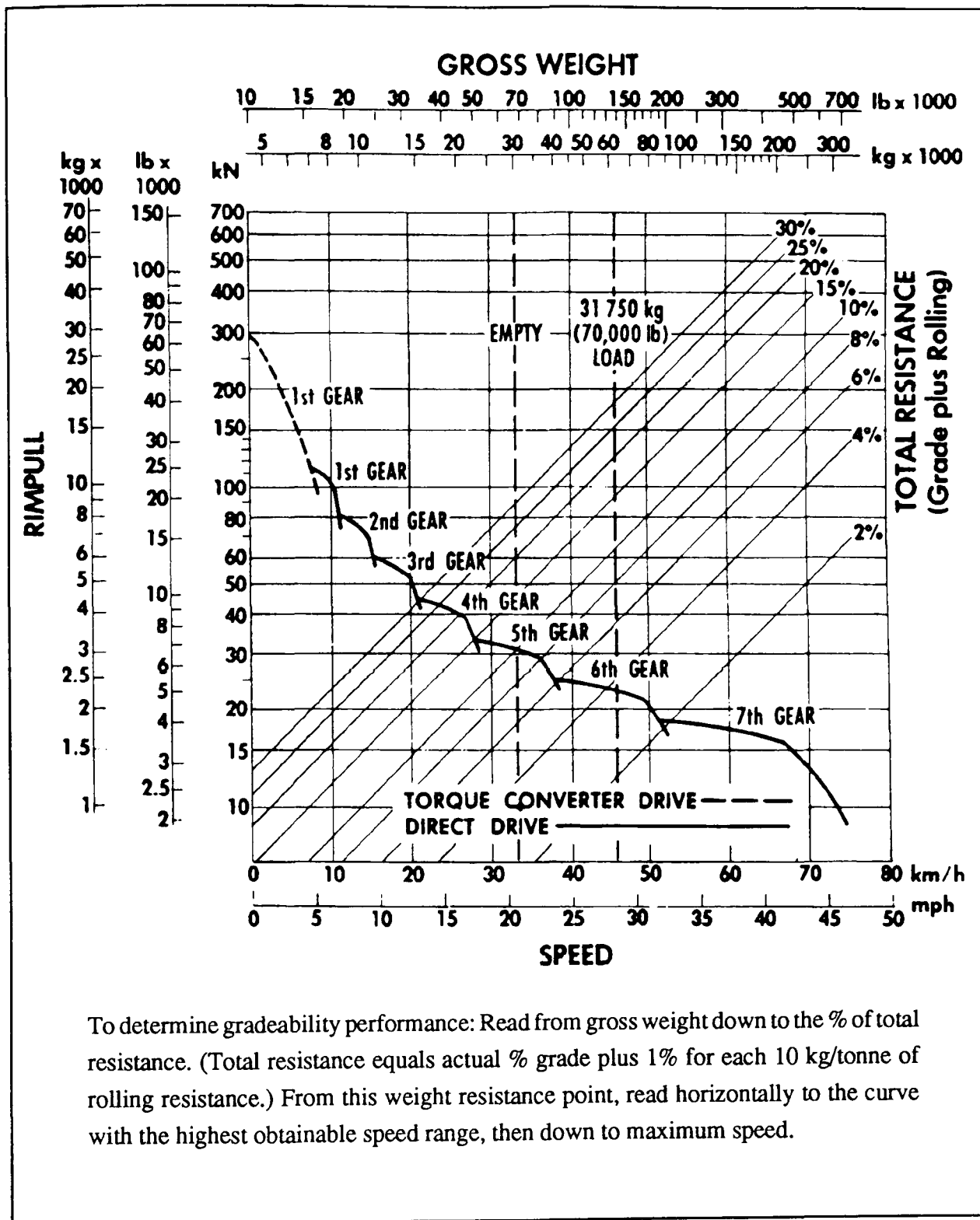


Figure 2.1

Rimpull-Speed-Gradeability for Caterpillar 769C Off-Highway Truck (after Caterpillar, 1980)

$$R_a = \mu \times W_w \quad \dots (2.4)$$

where: R_a = Available Rimpull (kN)
 μ = coefficient of traction
 W_w = weight on wheels (tonnes)

The maximum power in each gear is also limited by altitude.

Substituting the values obtained back into Equation 2.1:

$$a_i = F_2 / m - F_1' / 100 \times g \text{ (ms}^{-2}\text{)} \quad \dots (2.5)$$

from which the equations of motion:

$$v = u + at \quad \text{and} \quad s = ut + \frac{1}{2}at^2$$

are used to determine instantaneous changes in speed (may be negative if F_1 exceeds F_2) and distances covered (see Section 6.3.1)

2.2.2 Performance Simulation Packages

As well as his own deterministic model for shovel/truck fleet size determination, Srajer (1987) describes some of the range of performance simulation packages available today. Each are concerned with mimicking the performance of a given vehicle over different courses or comparing different vehicles over the same course. A course consists of a number of homogeneous road segments separated by identifiable changes in gradient, curvature or road condition. By continuously recalculating the rate of acceleration (as above) the programs model what happens as a vehicle moves through its range of gears until either the end or the maximum steady-state speed for a particular road segment has been reached. If the latter is achieved, the truck is assumed to continue at this speed until it arrives at the start of the next segment, whereupon the process is repeated.

Probably the best known package is Caterpillar's VEHSIM (Caterpillar, 1984, Singhal & Fytas, 1986) which receives data in 3 major groups :

- ❑ Machine data, which includes a digitised version of the rimpull-velocity curve, a shifting time between gears during which the machine can be expected to slow down due to motion resistance, and a mass correction factor to account for inertia in the vehicle's rotating parts;
- ❑ Course data, which describes both the haul and return routes in terms of distances, grades, rolling resistances and maximum velocities (see below);
- ❑ Simulation data, which is used to control the simulation process and produce performance reports at designated intervals.

Most of the other major manufacturers (including Wabco, Euclid, Terex, Kress, Komatsu and Unit Rig) and some mining companies have similar programs for calculating cycle times, though each may use a different set of heuristic values for the fixed time elements based on experience with their own equipment.

As pointed out by O'Neil and Manula (1967), the speeds obtained from the rimpull-speed-gradeability curve may exceed imposed limitations due to traffic, sharp curves and retarding on favourable grades. If this is true, the 'safe' speed for the segment, obtainable, in the case of downhill retarding, from digitised versions of manufacturers' brake performance curves (Fig. 2.2), must be achieved in advance. As is required when the truck approaches the end of either the haul or return course, a deceleration rate must be applied during the previous segment(s).

Many of the independent deterministic cycle time calculation programs form part of larger open pit modelling packages (Kim & Ibarra, 1981, Fytas & Calder, 1984). The stand-alone system developed by Boulton and Blair (1980) makes continuous use of feed-back and feed-forward loops to model the probable actions of the truck driver with regard to propulsion and brake control.

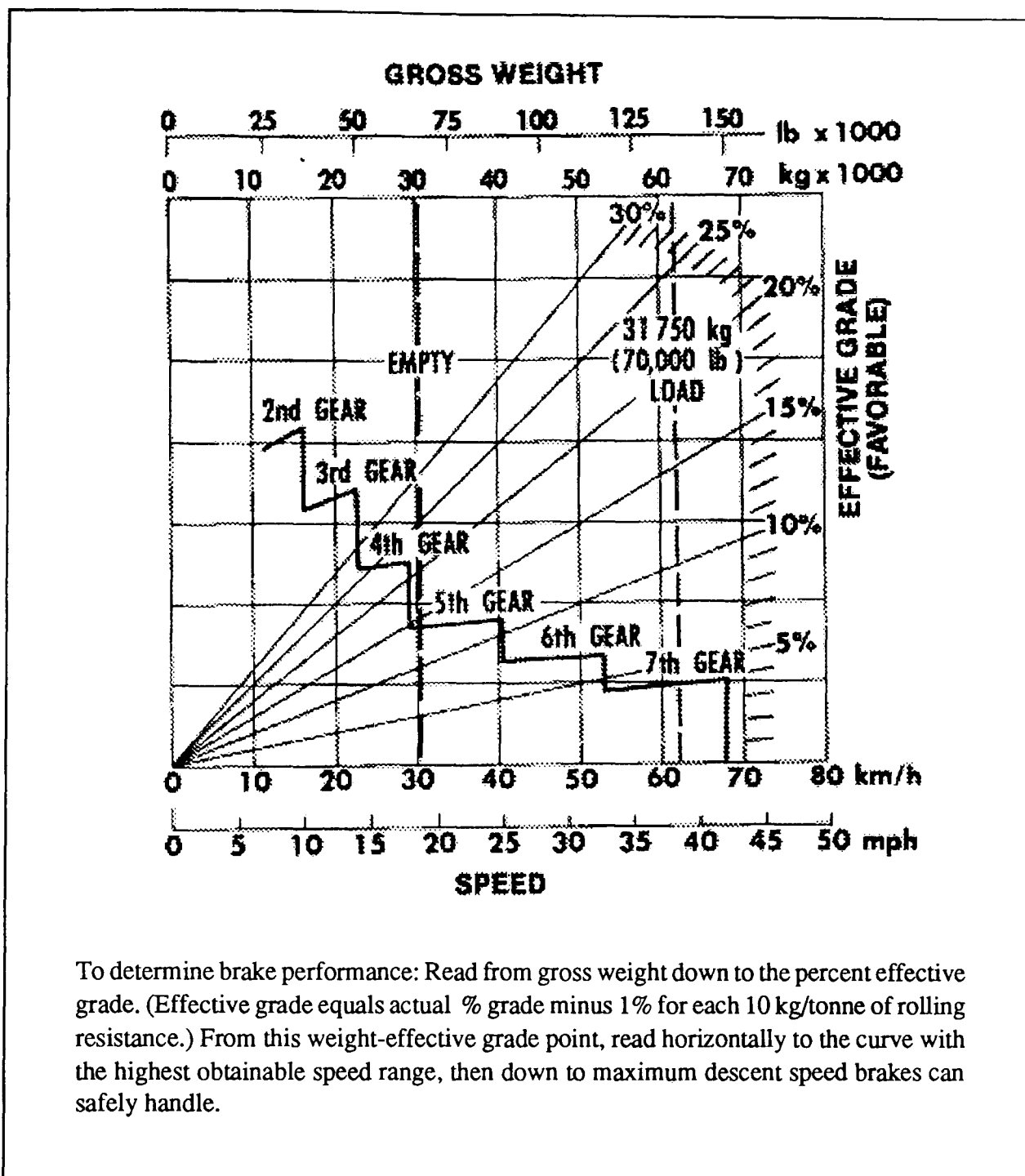


Figure 2.2

Brake Performance for Caterpillar 769C
Off-highway Truck (after Caterpillar, 1980)

2.2.3 Prediction of Productivity Variations

Cycle time calculations take no real account of the interaction between the various components of the fleet. Morgan and Peterson (1968) recognised how a model of current operations could be used to predict future equipment requirements with delays in mind. Their objectives were to predict the productivity of the present fleet under the new operating conditions and then, by correcting the potential production rate for the interaction between shovels and trucks, determine the most economic new fleet size.

The variation in predicted haul times was assumed to be distributed about the calculated value in proportion to the variation of current measured values about the current calculated value (Fig. 2.3). The effects of interaction, they suggest, could be corrected using average values for mismatch* and truck bunching. Bohnet (1984) goes a stage further in modelling the effects of mechanical availability and operating efficiency on shovel/truck interaction using binomial distributions.

Though these methods provide a quick analysis of truck combinations operating with a single loader, he recognises that more complex arrangements involving multiple haul routes, dispatching techniques and blending requirements can only be analysed accurately using stochastic simulation.

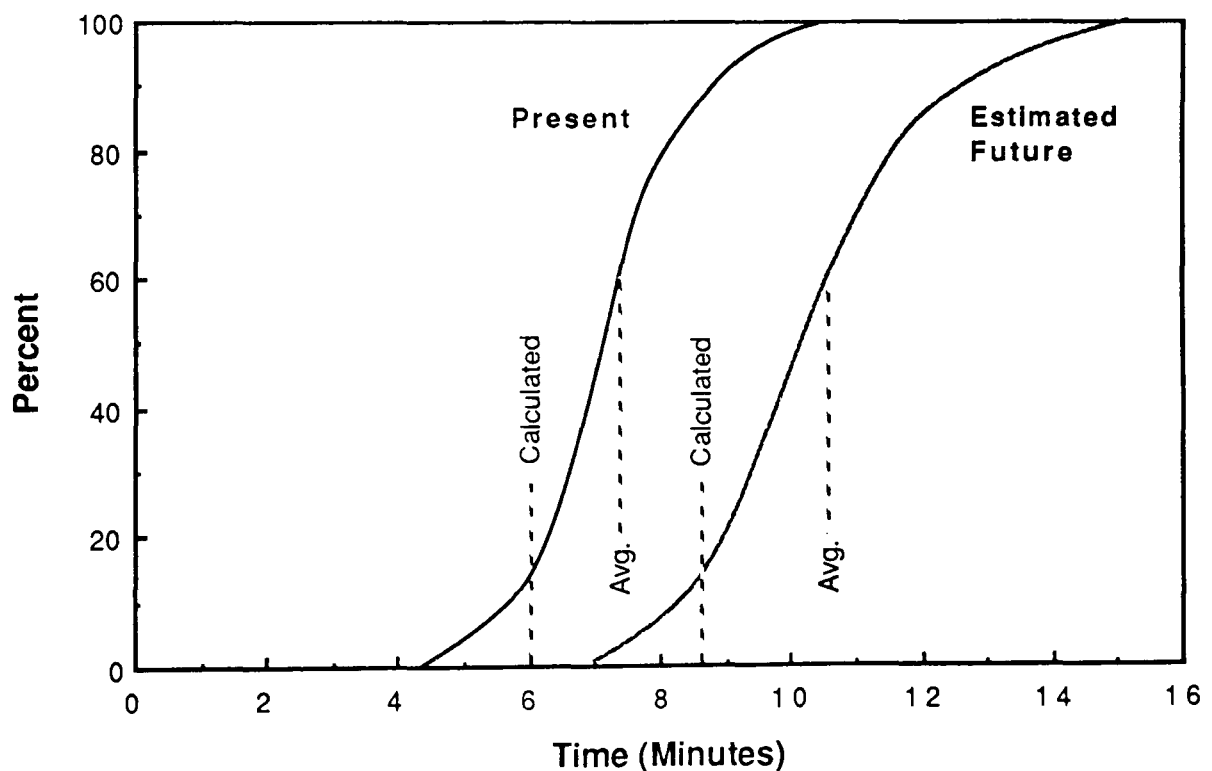


Figure 2.3 Estimated Future Production. (After Morgan & Peterson, 1968)

* Mismatch is the condition which exudes when loader capacity outweighs truck fleet capacity or vice-versa.

2.3 Simulation of Truck/Shovel Systems

Fleet size determination cannot be based entirely on the results of cycle time calculations. This is particularly so in complicated and marginal deposits and where there is a high likelihood that a combination of mixed sized trucks will form the optimum solution. There are several reasons:-

- ❑ The probabilistic nature of individual units negates corrected mathematical models of the type suggested by Morgan and Peterson (1968) and Bohnet (1984).
- ❑ No account is taken of the interaction between different items of equipment and queuing. As a result, very little is learnt about idle times.
- ❑ There is no way of evaluating situations where multiple mine faces link to multiple mine destinations
- ❑ The assessment of technical changes such as the introduction of trolley-assistance is often a laborious process.
- ❑ No account is taken of particular problem areas throughout the day such as at the start or end of a shift and when absenteeism results in operator deficiency.
- ❑ No account is taken of breakdowns.

Simulation allows the user to quickly determine the areas of operation which are most likely to affect productivity and to evaluate haulage schemes with regard to the location and scheduling of fleet items. For example, it is often unclear whether the introduction of additional trucks will increase productivity or simply congest the mine further (Bauer & Calder, 1972).

2.3.1 The Model Building Process

Having determined that the analysis of a truck-shovel mining operation is too complex to handle mathematically, the next task is to set about building a simulation model for the purpose. Tu and Hucka (1985) describe the main steps in the model building process under 3 main headings:

1. Collection of data

In order to ensure that the model will reflect reality, the input data it receives must be accurate. Data collection involves clear definition of the individual elements of the truck and shovel work cycle. For most existing systems, these are identical to the elements listed in Section 2.2.1. The time taken to complete each element can then be established either

- by observation,
- from established production reports,
- by calculation e.g. a deterministic model of truck haul times.

In reality, element times and load weights are distributed about a mean value and vary from one mining operation to another. This is reflected in the different distribution parameters utilised by the various models described in the next section. To examine the validity of any assumed distributions, tests such as the Kolmogorov-Smirnov goodness of fit test is carried out (see also Kim & Ibarra (1981)).

Figure 2.4 gives a breakdown of the elements which may require observation. For improved accuracy, wait times should be incorporated into the simulation process by modelling the queues which develop if a load or dump position is unavailable at the time a truck arrives and from which they are usually served on a first come first served basis (Bauer & Calder, 1970).

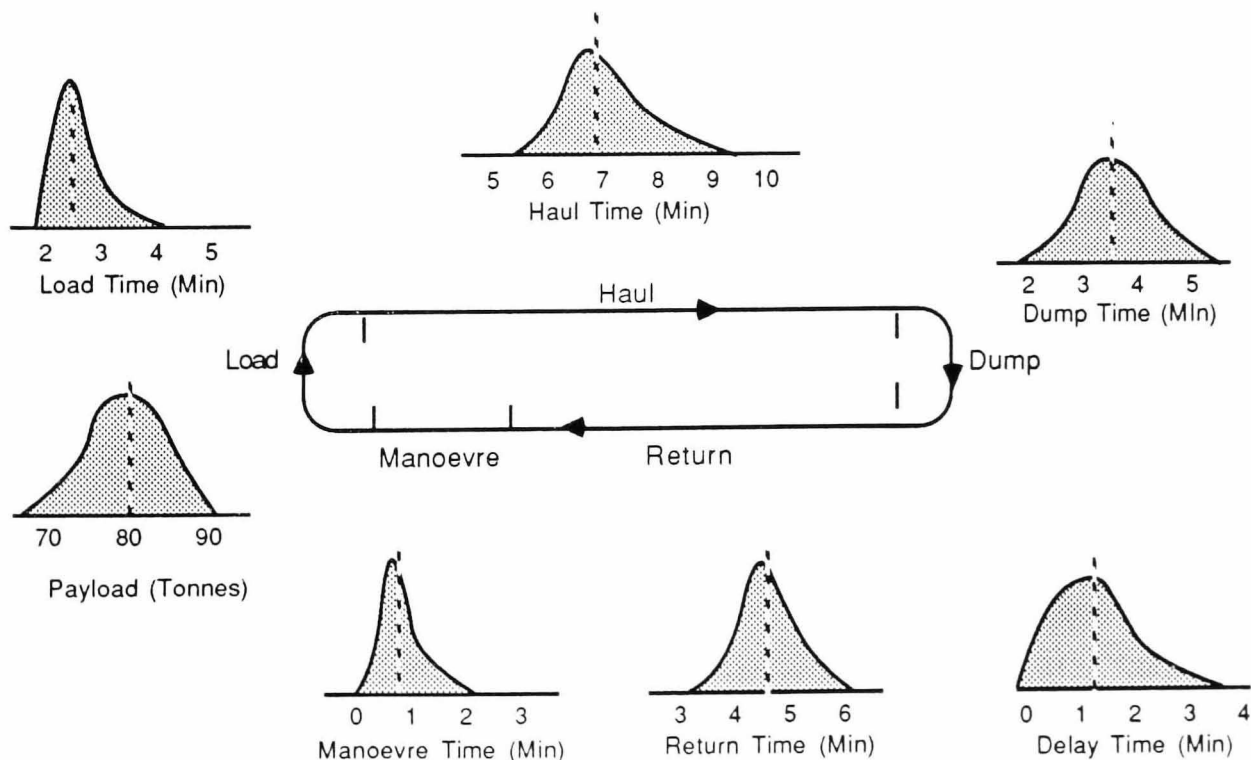


Figure 2.4 Elements in the Haul Cycle which may Require Observation

2. Definition of the model structure (Programming)

This includes definition of the haulroad network, truck and shovel assignments and other data such as shift times. The programming process brings together the model structure in combination with the probability distributions described above.

3. Model Validation

Generally involves comparing the results of a number of simulation runs with data collected from the operation being studied. Each run should be started with a different set of random number generating seeds (Lizotte *et al*, 1985).

2.3.2 Truck/Shovel Simulation Models

A description follows of some of the open-pit haulage simulation packages which have been developed over the last 20 years. While the overall model structure tends to be similar throughout, the conclusions drawn reflect the depth of knowledge which had been reached at the time.

O'Neil and Manula (1967) developed a system in an IBM FORTRAN based language called DAFT. In this system, a log of shift operating time was kept for each haulage cycle element, the unit with the minimum value being selected for movement. Output information included the number of loads and the total ore and waste hauled by each truck from each shovel together with shovel idle times, truck wait times and cyclic queuing delays. In addition, the speed and location of each vehicle could be obtained every second.

Tests carried out on a prototype mine produced preliminary fleet sizes and fixed assignment alternatives which were sufficient to handle the mine's required production capacity. It was also demonstrated how simulation could be used to examine the effects of variations in haul profiles, rolling resistances and load weights.

Cross and Williamson (1969) experimented with a refined mathematical model developed jointly by the University of Arizona and CompuTech Research Ltd. They decided, at this early stage, that it was not necessary to simulate actual truck performance and that mean times from a simulation carried out beforehand or actual data would be sufficient. Hence, random numbers from a normal distribution were generated about the expected value.

To justify his comparison of problem solving methods, Jolley (1970) compared monte-carlo simulation with mathematical analysis for truck/shovel selection and concluded that it is in the mid-range of trucks per shovel (i.e. the match point) where the largest discrepancy lies. When fewer trucks are considered, mathematical and simulation analyses produce similar results since few queues develop while in the upper range, saturation causes the results to converge.

Bauer and Calder(1972) took refuelling, shift breaks and right-of-way considerations into account and established that if simulation is based on reality, it can be used, with confidence, for major decision-making. In addition, the recognition that dumping points may be subject to extensive queues caused by blockage, prompted the suggestion that the selection of a visible dump destination according to its current status could be carried out on a trip by trip basis.

Together with Cross and Williamson (above), Bauer and Calder recognised that dispatching could be a way of improving truck productivity. However, it was only in the late '70s, after the technology required to control equipment in surface mines had been developed (Section 2.4), that a great deal of further investigation took place. The simulation models mentioned below form part of more generalised planning packages aimed, amongst other things, at the investigation of dispatching policies (Section 2.5).

Hauk (1979) describes what is essentially a deterministic model but which slightly randomises the time between events for each vehicle. Travel times are simulated by taking 90% of the value prespecified from time studies and randomising from a Poisson distribution with a mean equal to the remaining 10%.

Brake and Chatterjee (1979 + Chatterjee & Brake, 1981) used SIMULA which they found well structured, readable, efficient, easy to debug and which, like most other simulation languages, reduced the programming load considerably. They also pointed to the advantages of an efficient storage facility particularly in large mines where, due to the sheer volume of information, data could be easily corrupted or lost. The model consists of two modules: one used for planning prior to production; the other for evaluating existing operations. In the first module, all the events are log-normally distributed about a calculated mean. In the second, random values are taken from spline functions fitted to frequency curves of observed data. Additional features include the ability to obtain the disposition of all available equipment at any time, to model breakdowns and subsequently reallocate trucks easily and to model the effects of the introduction of trolley-assistance.

Kim and Ibarra (1981) describe the use of a Mine Operations Analysis Model developed by Kim and Dixon using the GASP IV discrete simulation language. Load, dump times and load weights were observed, travel times were calculated and normal distributions used.

The model developed by Wilke and Heck (1982) does not differ greatly from that described above by Kim and Ibarra but highlights the special problems of truck haulage under difficult conditions. Breakdowns are modelled stochastically by establishing the number per shift, determining their length and randomly distributing them throughout the day.

Fytas and Calder (1984 + Singhal & Fytas, 1986) use PITSIM II, an interactive, discrete, open-pit modelling system developed at Queen's University. Loading and dumping are assumed to be distributed log-normally and haulage cycle times are calculated deterministically. Alternatively, empirical formulae may be used.

The model developed for simulation of the operations at Lac d'Amiante (Lizotte *et al*, 1985) makes extensive use of the Weibull distribution primarily because it can be adopted to a variety of shapes. Specific definition of the distribution requires 3 parameters: one defining shape, one scale and one location. The advantages are 3-fold:

- ❑ As well as the standard distributions listed in Table 2.1, the Weibull can be adapted to represent the skewed distributions which occur in reality.
- ❑ γ represents a minimum value for the distribution which will always be exceeded.
- ❑ A random variable can be easily obtained from the following equation:

$$x = \gamma + \beta(-\ln(r))^{1/\alpha} \quad \dots (2.6)$$

where α = shape parameter,
 β = scale parameter,
 γ = location parameter.

Other systems have been described by Manula and Ramani (1977), Bonner and Moore (1977) and Nenonen (1982). Tu and Hucka's model (1985) is among those described in greater detail in Section 2.4.

α	Distribution Represented
1	Exponential
1.5	Log-Normal
3.6	Normal

Table 2.1 Distributions which can be Represented Using the Weibull

2.3.3 Problems with Truck/Shovel Simulation

O'Neil and Manula (1967) recognised two very important limitations of the simulation process which must be taken into account:

- ❑ Trucks are assumed to operate near the limit of their abilities. Simulated productivity therefore tends to be optimistic and an allowance of, say, 10% is generally required. In working mines, this figure can be determined fairly accurately by comparing simulation results with what happens in reality.
- ❑ Even in working mines, it is difficult to find a sufficient number of production shifts performed under identical conditions to test the model rigorously.

Chatterjee and Brake (1981) question the usefulness of over-complex simulation models because of the difficulties involved in collecting and incorporating a large quantity of data in the detail required.

2.4 Truck Monitoring and Control

Prior to the mid 1970's and, in the majority of open-cast operations, right up to the present day, it has been standard practice to allocate a set of trucks to each operating loader at the start of a shift based on the production requirements for the day. It is clear that, due to the variable nature of mining operations, this fixed allocation technique has many shortcomings compared to a system which reassigns each truck on a trip by trip basis. However, it was recognised that these could only be overcome with the implementation of an efficient monitoring and control system.

2.4.1 The Need for Greater Control

The inherent disadvantages of an uncontrolled fixed allocation policy showed up differently at different mines throughout the world. At Palabora, for example, the fleet distribution calculated to meet with production requirements seldom produced a round number of trucks per shovel (Crosson *et al*, 1977). Marginal trucks were allocated according to shovel loading priorities based on the day's activities, with the result that high priority shovels were consistently overtrucked and low priority ones undertrucked. Without an efficient means of doing so, the controllers were reluctant to redirect traffic even when breakdowns or substantial changes in digging rate occurred.

At Pima (Mueller, 1977), a very large open-pit mine owned by Cyprus Mines Corp., radios were used to reassign trucks in emergency situations and, after investigations using queuing theory and computer simulation, to occasionally reassign trucks after each trip. However, they found it difficult to minimise truck delays at shovels and shovel wait times. It also proved impossible to follow up drivers to ensure that tasks had been completed as required.

Despite setting up a controller's office with good visibility of the mine, the manually recorded shift reports at Bougainville Copper Ltd (Hempenstall & Hill, 1980) still contained too many errors for the foreman to effectively supervise his fleet. At Lake Jeannine (Baron, 1977), attempts to enhance the controller's visual interpretation of the current situation and the use of radio communications meant that he spent most of his time recording and relaying information and very little improving fleet performance.

2.4.2 Manual Control Systems

The above were all examples of mines which used manual control - a system which relies heavily on the use of radio transmitted data with occasional reallocation of trucks to shovels in emergency situations. In more advanced systems, trucks are reassigned each time they pass a control point by displaying, in rotation, cards with shovel identification codes printed on them (Rossouw, 1986) or by flashing up instructions, derived at by some other assignment policy, on an illuminated board.

To solve their problem of fleet management, the engineers at Pima (Mueller, 1977) constructed a simple dispatching board designed to monitor the movements of a truck based on its predicted position in the haul cycle (Fig. 2.5). Transparent, plexiglass slides, numbered to indicate clock time, were advanced up the board at 5 minute intervals. When a driver radioed in, he would be directed to a new shovel based on the period elapsed since the previous allocation. At the same time, a plastic marker, colour coded to indicate its relative speed, would be located at a position some way down the board indicative of the number of 5 minute periods normally required for the truck to complete the cycle. This position was immediately obtainable using an identically coloured label. If the driver failed to radio in by the time the slides had advanced his truck's marker well beyond the 'on time' row, investigations would follow. In addition to the 10-15% gain in productivity resulting from tighter monitoring and control, this simple device acted as a useful tool in the dispatching decision process since, by looking at the board, it was immediately apparent which shovel had been waiting for an allocation longest.

2.4.3 Semi-Automated Control Systems

Initial investigations into computerised control began in a German limestone quarry in 1967 (Naplatanov *et al*, 1977). Until the late 1970's, a major limitation to the automisation process was cost. The introduction of 4-16K mini-computers at this time, however, made feasibility studies inevitable.

In semi-automated systems, a digital computer is used to monitor the status and location of the trucks which make up the haulage fleet. It may also be used in the dispatching process to select trucks according to an optimal assignment policy (Section 2.5). However, the final decision rests with the dispatcher who relays information manually either by radio or visual means.

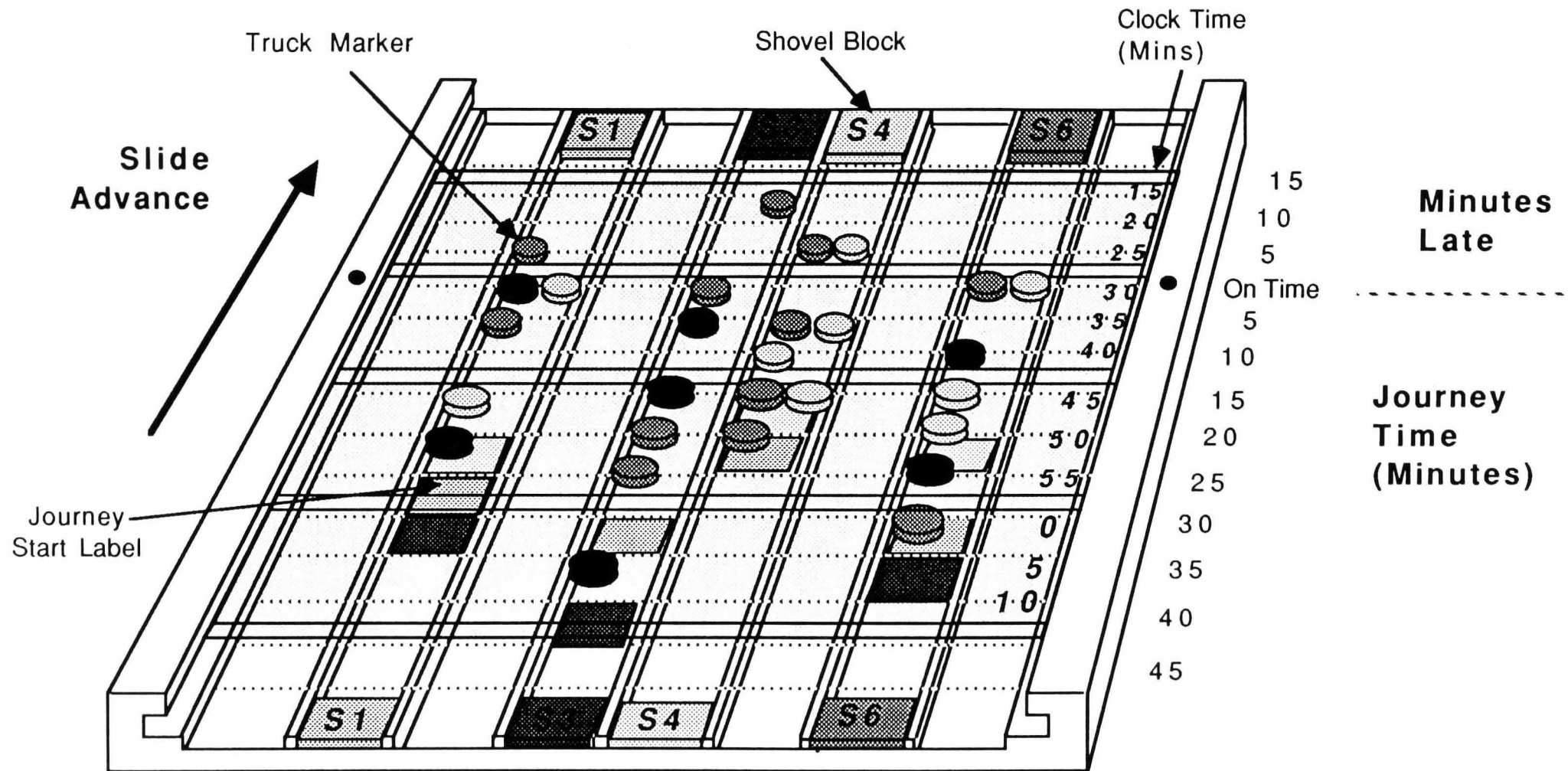


Figure 2.5 Dispatching Board (After Mueller, 1977)

At Lake Jeannine (Baron, 1977), a 16K mini-computer with an interface linking the administration and the truck control office was found to improve the decision making process. Initial attempts at automatic shovel and truck identification were beset with difficulties.

Meyer (1979) describes such a system applied to an open pit copper mine in Tucson, Arizona. Trucks show up as rectangles which are automatically assigned to shovel columns on a graphics screen. The rectangles are modified when the computed allocations are firmly accepted by the dispatcher.

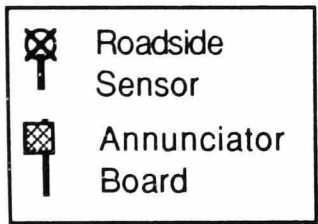
At Bougainville (Hempenstall & Hill, 1980), a monitoring and report generation system (MPS) consisting of two screens, one detailing shovel locations and truck allocations, the other drivers and equipment, was installed and was found to give the foreman greater control. However, a fixed allocation policy was continued, the system's main benefits being realised when breakdowns etc. occurred.

Lizotte *et al* (1985 & Lizotte *et al* (1987)), concerned about the cost of establishing full control in small to medium-sized operations, designed a similar system for semi-automated truck dispatching at the Black Lake asbestos mine. A Hewlett Packard 9845C computer, specifically chosen for its graphics capabilities, was used to display a 'dispatching board' type representation of the current pit situation. Again, a fixed policy was retained with reassignments being made every time a breakdown or shovel move occurred.

2.4.4 Automated Control Systems

The fundamental problem with both manual and semi-automatic control systems is that they rely, to a large extent, on man's time and ability. In fully automated systems, information from either the controller or the truck is radio-linked to an on-board truck transmitting device or roadside beacons and the human element is more or less eliminated (Fig. 2.6).

There are three main types of beacon in use (Aiken, 1980). Selection of the correct beacon type depends on the circumstances which prevail at the mine.



1. Beacon detects that a dispatch is required
2. Shovel displayed on annunciator panel
3. Second beacon signals cancellation of the display

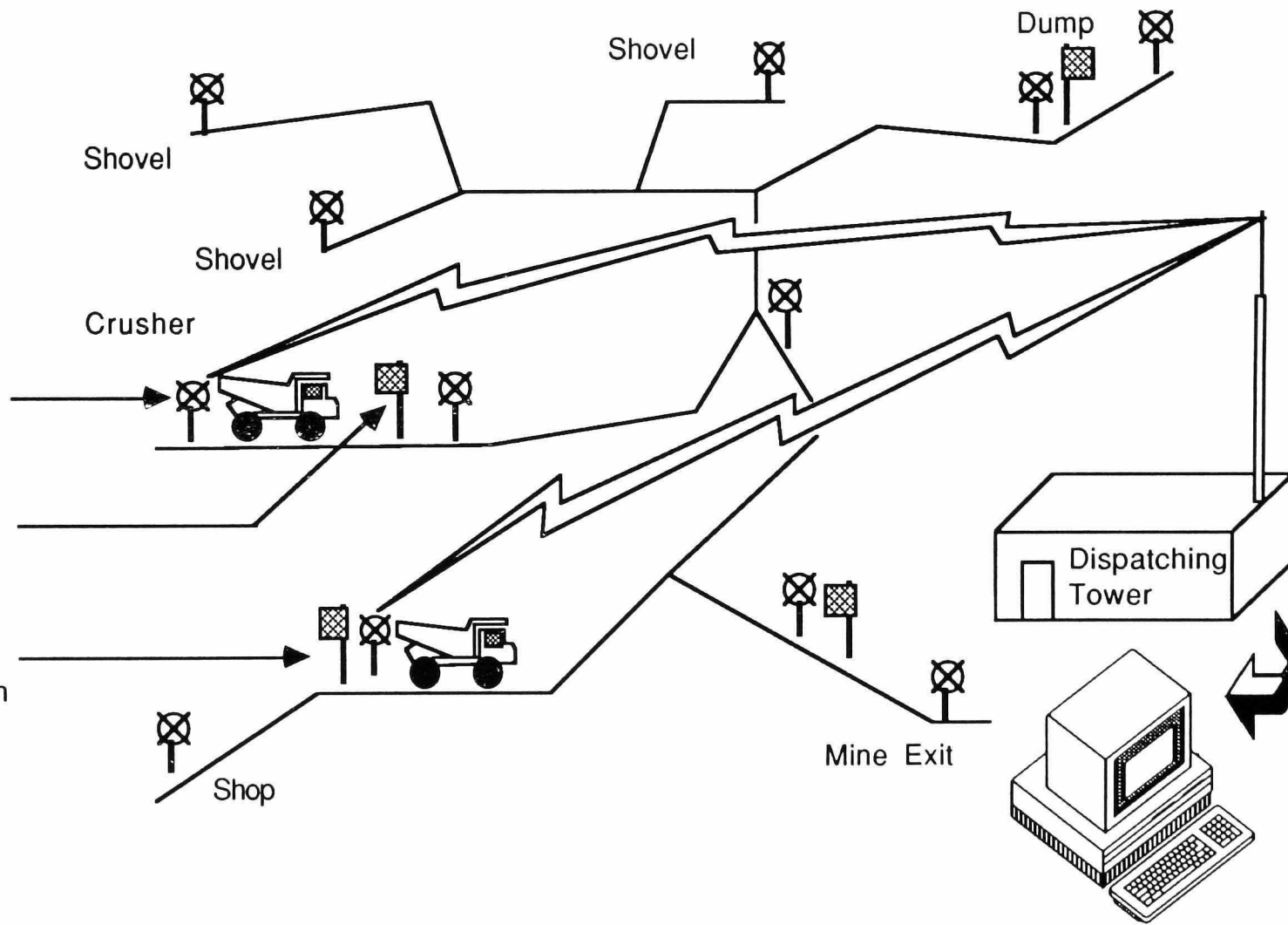


Figure 2.6 Automated Truck Control System (after Aiken, 1980)

❑ Optical System.

This system uses a bank of light-sensitive photoelectric cells which are mounted vertically so that they become activated by a beam emitted from a passing truck. The signal is transmitted to a central computer which automatically selects a shovel on a time priority basis and flashes its identification code onto a display board. Optical systems may be subject to stray light beams - a problem accentuated when the road in the vicinity of the sensor becomes worn to such an extent that vertical misalignment takes place. If something or someone stands between a truck and the sensor blocking the activating beam, no allocation message is received.

❑ Buried Loop System.

Here, an electric loop, powered by a low wattage source, is buried at each of the mine's control points. As the truck passes over a loop, it receives a signal which is encoded on board and transmitted to the central control. Dispatching instructions are relayed to the driver by radio or annunciator as above.

❑ Microwave Systems.

Using a microwave system, vehicles can be tracked along their entire route from roadside sensors located throughout the mine. Information concerning the truck number, size, material carried, etc., are transmitted to the control computer either by cable or radio waves from the beacon. As an alternative, sensing and transmitting may be carried out from the truck. This avoids the possibility of two adjacent trucks having to be dealt with simultaneously by a single beacon.

Investigations into the possible installation of an automatic truck control system at the large Palabora open-pit copper mine began in 1970 (Crosson *et al*, 1977). Time studies carried out during 1971 showed that truck delays at shovels accounted for over 12% of total truck time. Since the mine was undertrucked by design (on the principle that having a single shovel standing was less costly than having a number of waiting trucks), it became apparent that these delays were avoidable. The requirements of the dispatching system, then, were 3-fold:-

- ❑ Avoid excessive waiting time at the shovels,
- ❑ Maintain production levels at pre-define tonnage and grade specifications,
- ❑ Maintain waste removal in accordance with the stripping ratio.

A dispatching algorithm was formulated (Section 2.5.2) based on the above requirements with the additional premises that:-

- ❑ In the long term, the total tonnes hauled per km will be constant so there is no point in considering distance when selecting shovels;
- ❑ Ore shovels should be given priority over waste shovels.

The calculations were to be carried on an 8K digital computer housed in a small building next to the pit lookout.

After examining the various methods of registering truck movements and identifying their capacity, it was decided to use an inductive loop system whereby a signal, emitted from the trucks is picked up by receivers located at each pit entry. When a truck passed within 8m of a sensor, the shovel with the greatest instantaneous priority value was flashed up on an annunciator positioned some way ahead.

The reduction in truck delay times brought about by installation of the above system meant that the current production requirements could be fulfilled with fewer trucks. This resulted in the cancellation of the planned purchase of a 90 tonne truck - an immediate saving worth six times the cost of the total system. Additional benefits included :-

- ❑ Reduction in human error brought about by continuous recording and automatic report generation.
- ❑ More accurate pit planning.
- ❑ Less monotony for the drivers who had previously spent their whole shift travelling between two destinations.

At Medet (Naplatanov *et al*, 1977), it was realised that the improved truck utilisation and blending brought about by computerised control could lead to substantial cost savings. This deduction was given extra significance at a mine where the low grade copper deposits made, what was a very large operation, particularly vulnerable to price fluctuations.

The control procedure depended on the results of tests carried out after each blast to determine copper content. Based on these, the program split loading stations into high or low yield sites and truck allocations were made according to the ratio of ore (waste) produced over the ore (waste) production quota for each. If the number of trucks allocated to a shovel exceeded a pre-determined critical value, the truck in question was withheld while the operator decided either to increase the critical value and, hence, the likelihood of queuing, or choose another destination

based on the time elapsed since the last allocation and the travel time from shovel to dump. Physical control was carried out by radio transmission from the trucks to roadside beacons. A second monitoring unit, downstream of the annunciator address boards, caused cancellation of the shovel allocation number displayed.

After installation, Medet recorded a production increase of between 7 and 10 percent. More difficult to quantify but equally important, they noticed a significant reduction in maintenance and replacement costs. In addition, concentrate production increased due to improved flotation of the properly blended product.

Haulage costs at Sishen iron-ore mine (Rossouw, 1985) represent only 20% of the total operations. Nevertheless, rapid increases in production and scale during the period between 1973 and 1982 made the fixed allocation policy currently employed unsuitable. Simulation studies, backed up by careful analysis of the noticeable faults in the existing manual system, showed that a 7% increase in production could be achieved.

The hardware and the control procedures adopted were based, to a large extent, on the system used at Palabora (above) with a number of refinements in the light of recent developments in the field. These were aimed at effecting improvements in the following areas:

- Simplicity
- Acceptance by Operations Personnel
- Reliability
- Ease of Maintenance
- Flexibility with respect to frequent changes in equipment location.

Implementation of the system meant that the truck fleet was efficiently utilised and controlled and personnel were set free to devote their time to direct supervision. A production increase of at least 10% was actually realised (Fig 2.7).

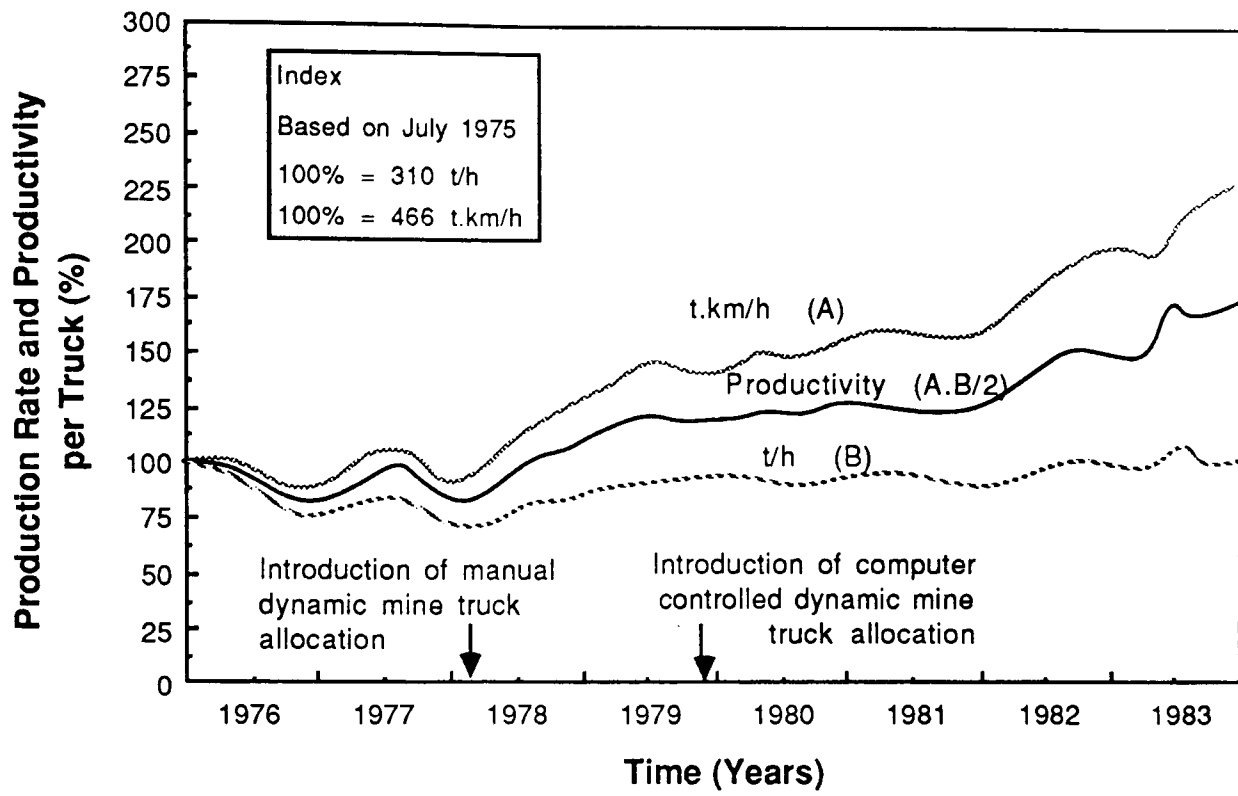


Figure 2.7 Production Rate and Productivity at Sishen Iron-ore mine
(After Rossouw, 1986)

2.5 Analysis of Dispatching Systems by Simulation.

Dispatching policies range from simple heuristic rules through to sophisticated mathematical procedures. Depending on their degree of complexity, the general description of dispatching policies given in published papers are not sufficient for rigorous analysis. However, computer simulation can be used to obtain a 'feel' for those which are described in even the briefest detail with regard to the likely effect on the mine environment. More detailed studies can be carried out to assess the feasibility of introducing a dispatching system and whether the cost benefits of increased efficiency would offset its installation and operating costs.

Policy	Criteria	Tested by:
Maximise Shovels	Assign to the shovel which has been waiting longest or is likely to be idle next..	Cross & Williamson (1969) Kim & Ibarra (1982)
Maximise Trucks	Assign to the shovel at which the truck is most likely to be loaded first.	Brake & Chatterjee (1979)
Blending Schedule	Assign to the shovel which is furthest behind schedule.	Wilke & Heck (1982)
Shovel Coverage	Spread allocations evenly amongst all the shovels according to the current match factor.	Lizotte <i>et al</i> (1987)

Table 2.2 Summary of Standard Dispatching Policies

2.5.1 Standard Dispatching Policies

Apart from fixed allocation, the most commonly used truck dispatching criteria are given in Table 2.2

Cross and Williamson (1969) carried out preliminary investigations into the first of these and results indicated that the increase in loading time which could be achieved with dispatching would mean that fewer trucks would need to be allocated to each shovel. In fact, simulation tests based on data collected at Pima showed that a fleet reduction of five 100 tonne trucks could be achieved and that this would compensate for the current 'production limiting' down-times due to repair, refuelling etc. Figure 2.8 shows that the application of a dispatching system is most effective below saturation and lessens as saturation is approached.

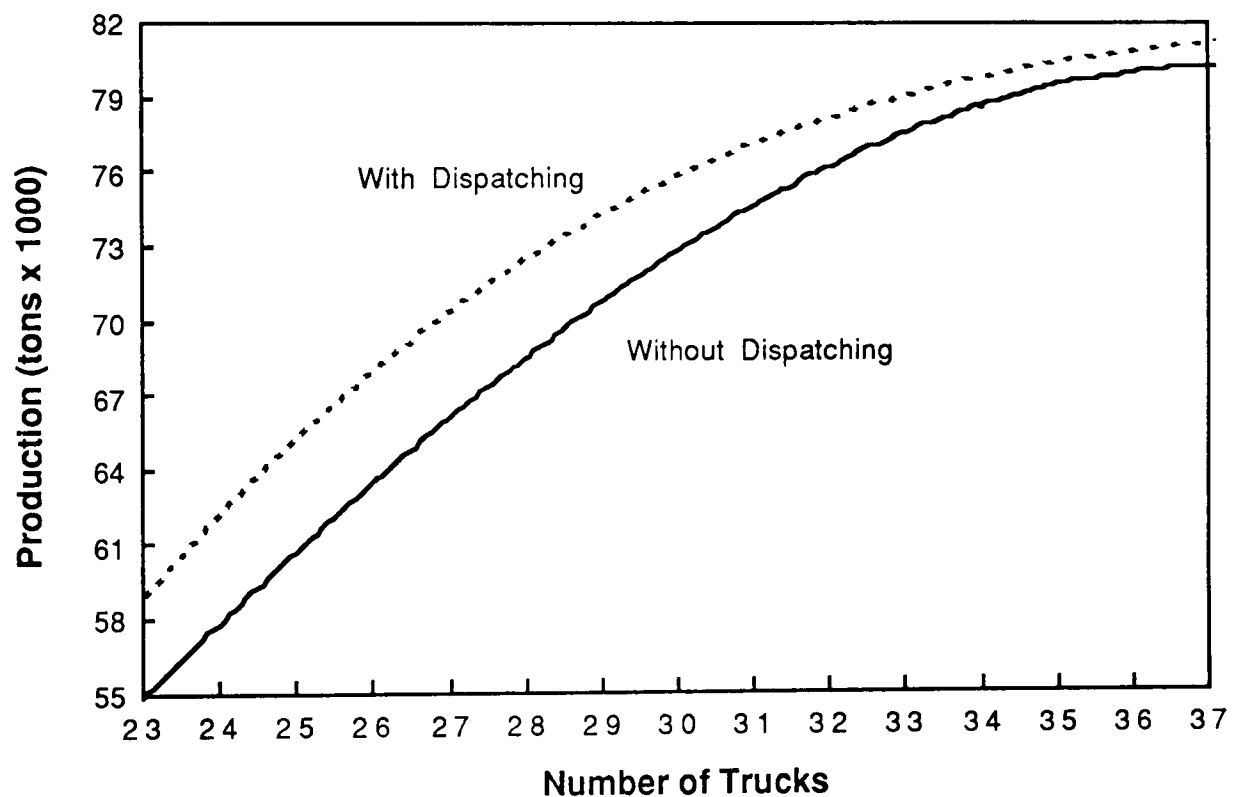


Figure 2.8 Effect of Dispatching Below Saturation.
(after Cross & Williamson, 1969)

Brake and Chatterjee (1979) compared 3 dispatching options using their SIMULA model namely:

1. Fixed Allocation;
2. Minimise queuing but fixed to dumps;
3. Minimise queuing;

Though some benefits of dispatching in terms of productivity and utilisation were evident, the limitations of the minimise queuing algorithm adopted singularly also showed through in what was a very simple test.

More extensive tests were later carried out on data collected at Newlands strip coal mine (Chatterjee & Brake, 1981). The MINPLN module predicted production and utilisation increases for both trucks and shovels of 3% and 4% respectively (Table 2.3). The relatively small improvements were due to the size of the fleet (12 trucks, 2 shovels) which provided a very limited choice anyway. The 9 shovels and 33 trucks in operation at a large open pit copper mine produced more encouraging MINEVL simulation results predicting that an 8.3% increase in productivity could be brought about by the introduction of dispatching (Table 2.4). Further, it was concluded that a greater improvement could be obtained over long haul distances than over short ones.

	Fixed Allocation	Dispatched Allocation	Increase (%)
Truck Utilisation	73.0	76.0	4.11
Shovel Utilisation	92.0	96.0	4.35
Truck Tonnage	6222	6424	3.25
Shovel Tonnage	37330	38541	3.24

Table 2.3 Increases in Utilisation and Productivity at Newlands as Predicted Using MINPLN. (after Chatterjee & Brake, 1981)

	Fixed Allocation	Dispatched Allocation	Increase (%)
Truck Utilisation	89.6	92.5	3.24
Shovel Utilisation	63.0	61.1	3.11
Truck Tonnage	7506	8132	8.34
Shovel Tonnage	27606	31030	12.40

Table 2.4 Increases in Utilisation and Productivity at an Open-pit Copper Mine as Predicted Using MINEVL. (after Chatterjee & Brake, 1981)

Kim and Dixon's model (Kim & Ibarra, 1981) was used to test out a minimise shovel waiting algorithm in which an empty truck is assigned either to the next available shovel or to the one which will have been waiting longest when the truck arrives. The Open Pit Mine Simulation model (OPMS) was first used in non-dispatch mode and adjusted until it conformed with the actual operations at La Caridad Mine. The same corrections were then applied to the model in dispatched mode which predicted a 14% improvement in productivity together with reductions of up to 30% in shovel and truck waiting times. A comparison of the results obtained for 6 different sub-area configurations showed that in dispatching regions which consist of a number of mixed long and short haul cycles there is likely to be a greater degree of improvement than where all the possible routes are of a similar length.

Wilke and Heck (1982) recognised the importance of meeting certain ore blending requirements as well as maximising fleet utilisation. Here, trucks are initially allocated to shovels according to the production requirements for the shift. Thereafter, empty trucks are assigned to those shovels which are most behind schedule taking into account trucks which are already on their way. For example, if the production target for a particular shovel is 5000 tonnes per day, at 11.30, it has loaded 1900 tonnes and there are two 50 tonne trucks on their way, its relative arrears will be $5000 - 2000/5000 = 60\%$. The shovel with the largest relative arrears is selected.

Tu and Hucka (1985) developed a model with which to test out the first three policies. They used the SLAM simulation language (Pritsker & Pegden, 1979) representing each truck and shovel as an entity moving round a discrete event network (Fig. 2.9). Shovels were also modelled as resources liable to seizure at random intervals by simulated breakdowns and face moves. After validation, a series of tests were carried out on the maximise shovels policy with 9 simulation runs being considered adequate for each alternative. The results reinforced some of the arguments put forward by Cross and Williamson and also highlighted that, near saturation, the increase in production obtainable by adding one operating shovel far exceeds the expected gain if dispatching was introduced (Fig. 2.10).

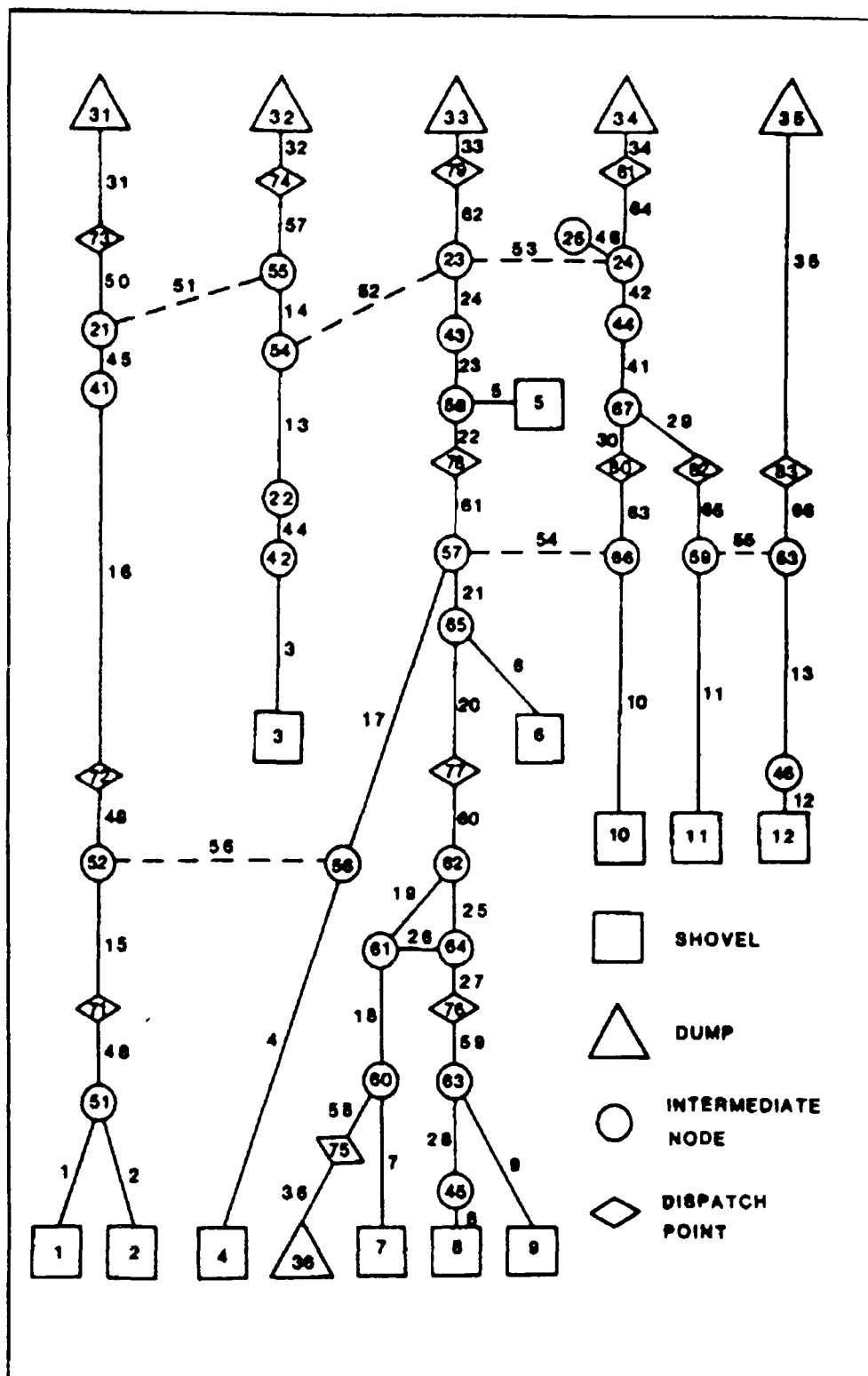


Figure 2.9 Haulroad Simulation Network Diagram (after Tu & Hucka, 1985)

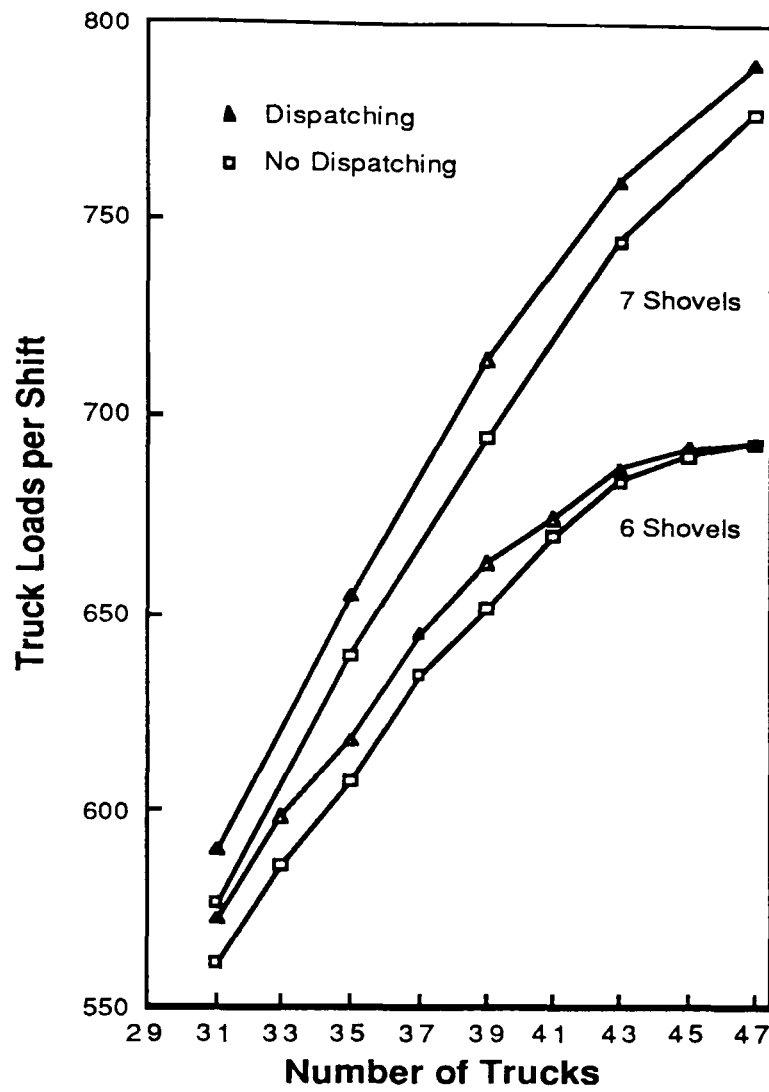


Figure 2.10 The Relative Value of Adding One Shovel Near Saturation (after Tu & Hucka, 1985)

To justify the usefulness of and assess potential improvements to their compact, semi-automated system as applied at the relatively small Black Lake mine, Lizotte *et al* (1987) tested the maximise shovel, maximise trucks and shovel coverage policies on their Weibull based simulation model (Section 2.3.2). The shovel coverage rule is based on the match factor - a dimensionless number originally devised by the Caterpillar Tractor Co to indicate the apparent balance between loaders and haulers in a particular situation:

$$MF = \frac{N_t \times T_s}{N_s \times T_t} \quad \dots (2.7)$$

- where:
- MF = Match Factor (< 1 indicates under-trucking, > 1 indicates over-trucking),
 - N_t = Number of trucks,
 - N_s = Number of loaders,
 - T_t = Truck Cycle Time (secs),
 - T_s = Loader Cycle Time (secs).

Here, current MF values are calculated for each sub-system within the pit. These are compared to the desired coverage based on production requirements and the shovel with the lowest ratio is selected.

Tests once again highlighted the relative benefits of introducing a dispatching system based on the first two policies but the levels of production achieved using the shovel coverage rule fell short even of those obtained using fixed allocation. However, with a global desired match factor of 1, the difference in production between the various operating shovels was shown to be negligible - invaluable at a mine where grade control is of primary importance.

2.5.2 Non-Standard Dispatching Policies

As mentioned above, a number of heuristic policies also exist. The dispatching algorithm employed at Palabora (Crosson *et al*, 1977) and, more recently, at Sishen (Rossouw, 1986), for example, is a modified version of the 'behind schedule' policy. Trucks are dispatched to the shovel with the highest current priority value. The priority of that shovel is then reduced reflecting the fact that it is closer to reaching its production objectives:-

$$P' = P - F(L/R) \quad \dots (2.8)$$

where: P' = Reduced Priority Figure,
P = Previous Priority Figure,
F = Factor (Determined Practically),
L = Truck payload,
R = Loading Rate of Shovel.

At Sishen, the priority value of each shovel is increased by dP every 5 seconds

$$\text{where: } dP = \frac{RT \times 5 \text{ (secs)}}{SA \times 60 \text{ (secs)}} \quad \dots (2.9)$$

T = Total truck capacity in area (t/h),
S = Total Shovel Capacity in area (t/h)
A = Average cycle time (min).

The maximum priority figure, and that which is allocated to each shovel after a major production stand-still, is 30000.

Hauck (1979) proposes a dynamic mechanism as an alternative dispatching technique. He recognises two methods of allocating trucks to shovels: First, they can be apportioned among shovels according to the ratio of haul time to the sum of all haul times; Second, they can be apportioned with cost minimisation the primary objective. The latter applies the cost coefficients associated with equipment being left idle to a linear programming model. To allow for the variations between predicted event-times and those which occur in reality the control system uses feedback to continuously monitor a truck's progress at a number of key points throughout the mine. Continuous reallocation is possible with a truck only being committed to a shovel within the last 100m of its journey.

2.5.3 Summary of Dispatching Policies

In the above examples, simulation has been used to assess the relative merits of various dispatching policies. Now that a number of real time dispatching systems using different policies are up and running at mines throughout the world, most of the conclusions arrived at have been proven to be accurate and generally applicable. It has been shown that selection of the best dispatching policy is a site specific problem - especially where grade considerations exist. However, generalisations can be made about those standard policies which take no account of blending requirements. The maximise trucks policy, for example, tends to favour those shovels which are nearest the dispatching point while the maximise shovels policy tends to even out shovel utilisation at the expense of production.

A general conclusion which can be drawn is that dispatching can be made to reduce the variability of truck haulage systems by evenly distributing productivity and cumulative delay time between all the items of equipment used. Since this variability increases proportionally with the mean travel distance into the pit and the fleet size, the larger the scale of the operations, the greater the potential savings.

2.6 Automated Truck Dispatching

Initial attempts at automated truck dispatching during the late 1970's met with limited success due to problems with reliability and implementation. However, taking into account more recent technological advances, Arnold and White (1983) suggested that there were over 350 mines worldwide which could cost justify such a system.

At this time, 10 systems were available ranging from semi-automated through to fully computer controlled. Half of these had been developed internally for use at specific mine sites (Section 2.4), another 7 had been developed or were currently under development by computing companies. The typical cost of implementation and support ranged from \$600,000 to \$3,600,000 depending on the scale of mining operations. This could usually be justified on a potential capital cost savings basis. For example, the ability to achieve current production rates with a smaller fleet of trucks.

The objectives of any computerised dispatching system are to maximise mine production with available equipment while responding to regular changes in the pit configuration and providing full reporting capabilities. Features include:-

- Optimal route selection;
- Reduction of truck/shovel idle time;
- Blending control;
- Positive destination control;
- Report and accounting facilities;
- Automatic handling of refueling and tyre management;
- Improved response to pit disturbances;

Programs are designed so that the large amount of relevant information is collected and presented to the dispatcher in a concise and logical manner. The system designed by Pincock, Allen and Halt (1982 a & b), for example, controls displays and provides statistical reports for up to 30 loading units, 120 trucks of mixed carrying capacity, 30 service locations, 30 parking areas and 5 major material groups. Simulation can be added to model the flow of trucks at 30 times the speed of normal operations.

Sassos (1984) describes the Automatic Truck Dispatch system (ATD) developed by Ebasco services inc. Particular attention is drawn, here, to automatic monitoring as the most cost-effective method of reducing waste motion and idle times.

2.6.1 DISPATCH

Reflecting its level of implementation and success, the best known and most well-documented system is Modular Mining's DISPATCH.

Dispatch at Tyrone

In 1978 (Himebaugh, 1980), simulation studies at the Phelps Dodge owned Tyrone copper mine showed that the introduction of a computerised dispatching system could be extremely beneficial. The company entered into a joint venture with Modular Mining Systems to develop the general purpose DISPATCH system and, in 1980, Version 1.0 was implemented at the mine.

The product's primary consideration was the provision of an optimal truck flow pattern based on continuously changing mine parameters. This was achieved by applying the SIMPLEX method of linear programming (Section 2.7) to 'synergise' truck movements whenever major changes to the fleet configuration, such as breakdowns, moves, delays or changes in cycle times, occurred or, by default, every 20 minutes. Normal allocation of trucks to shovels in order to minimise waiting times at shovels etc. was carried out in the interim using a dynamic programming technique.

Hardware

Version 1.0 was designed to run on the D.E.C. VAX 11/780 computer located in Tyrone's general office. This was linked, via an underground cable, to the master control panel in the mine's central dispatching tower 3.2 km away. Microprocessor based field control panels were mounted in all trucks and shovels and at each crushing location. These contained 2 kilobytes (kB) of erasable programmable read only memory (EPROM) for program storage and 128 bytes of random access memory (RAM) for temporary transaction storage. Field messages, 8 bytes in length, were transmitted to the central computer via UHF FM digital radio for continuous evaluation (Fig. 2.11).

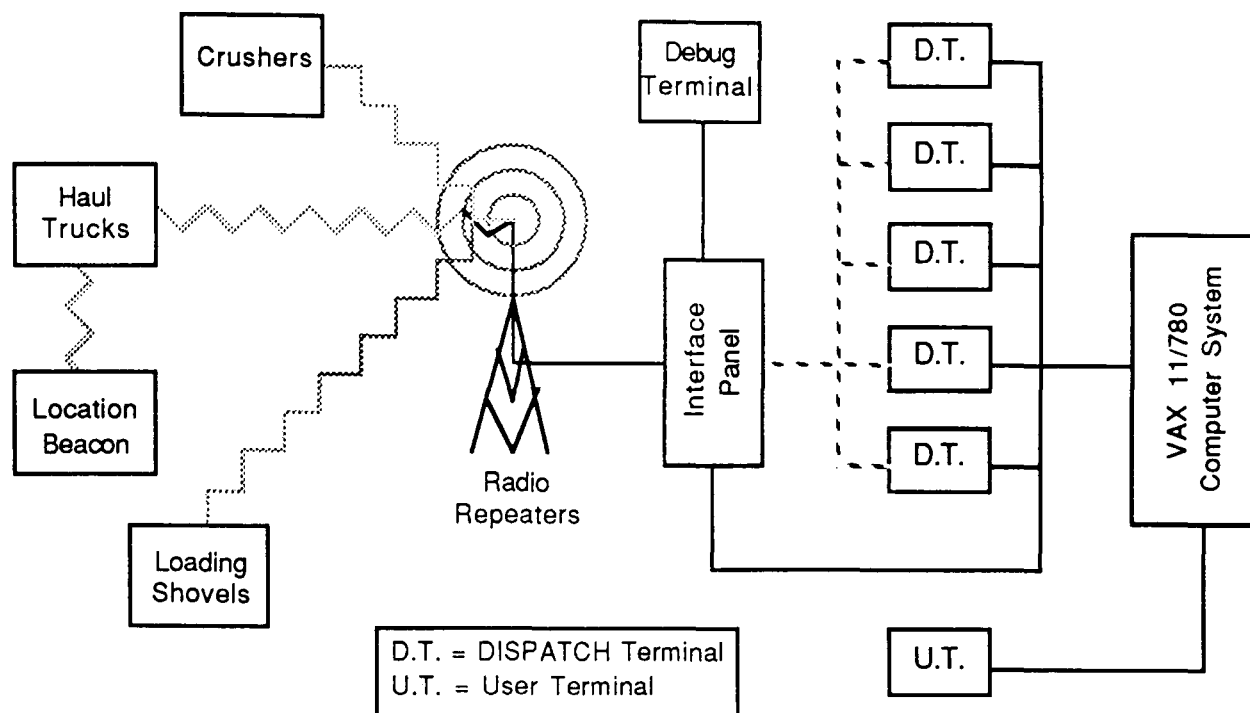


Figure 2.11 DISPATCH overall communications (after Farrell, 1988)

Mounted on the master panel were 2 CRT displays and a printer (Clevenger, 1983). The first of the terminals displayed pertinent pit information for the dispatcher's use including :-

- truck assignments;
- changes in material type;
- truck and shovel breakdowns and
- status changes.

The second was used for utility reporting e.g.

- equipment locations & availability,
- fuel reports,
- truck reports,
- shovel reports,

and input functions such as special assignments and field error corrections. The printer recorded a hard copy of the DISPATCH log which is kept for manual dispatching in the event of a system malfunction.

The total system required 200 kB of operating memory with a peak requirement of 300 kB. The capacity of the computer itself was initially 512 kB (later updated to 3.75 MB). As a consequence little program degradation was experienced by other VAX users.

Software

DISPATCH was originally programmed in VAX II FORTRAN - a superset of ANSI '78 FORTRAN. The programming load was divided into two main sub-systems sharing a single database: one to handle real time operations such as field transactions and dispatching; the other geared towards logging transactions as they occur and report generation. The first, being time critical, was given priority over the second and a prompt response time was considered vital. All input to the system was interactive with a built in ability to issue clear diagnostics and respond with examples if help was required.

Field panels were programmed in MOTOROLA 6802 assembly language.

System Operations

The sequence of operations required in the normal running of DISPATCH are described by Himebaugh (1980) and White *et al* (1982). A summary is given :-

At the start of every shift, the foreman enters shift set-up information based on the pit's production requirements. Operator and equipment assignments are also made taking into account any status changes such as the unavailability of load areas or dump points.

The dispatcher, who is responsible for the general running of pit operations, reviews the shift set-up and enters any necessary corrections. Thereafter, the computer takes over and all transactions are displayed as they occur in real time. The dispatcher is warned of any events which may require his intervention, such as a truck arriving at a shovel other than the one to which it was assigned, by an audio signal.

After logging in at the start of a shift, truck drivers are required to inform DISPATCH when they have reached certain points in the haul cycle (Fig. 2.12). Whenever a load or dump has been completed, the computed destination is displayed on the truck panel and the driver confirms that the message has been received by pressing the 'OK' button. If no confirmation is made in the first 5 seconds, an audio alarm is activated to alert the driver. Other details that must be relayed from the truck are gallons of fuel received, breakdowns and stoppages.

The shovel operator informs DISPATCH of changes in material type and activates the dump destination message to the truck by signalling when a load has been completed. Breakdowns and delays are also relayed. When a shovel indicates that it is ready to move, truck allocation to it is temporarily suspended. However,

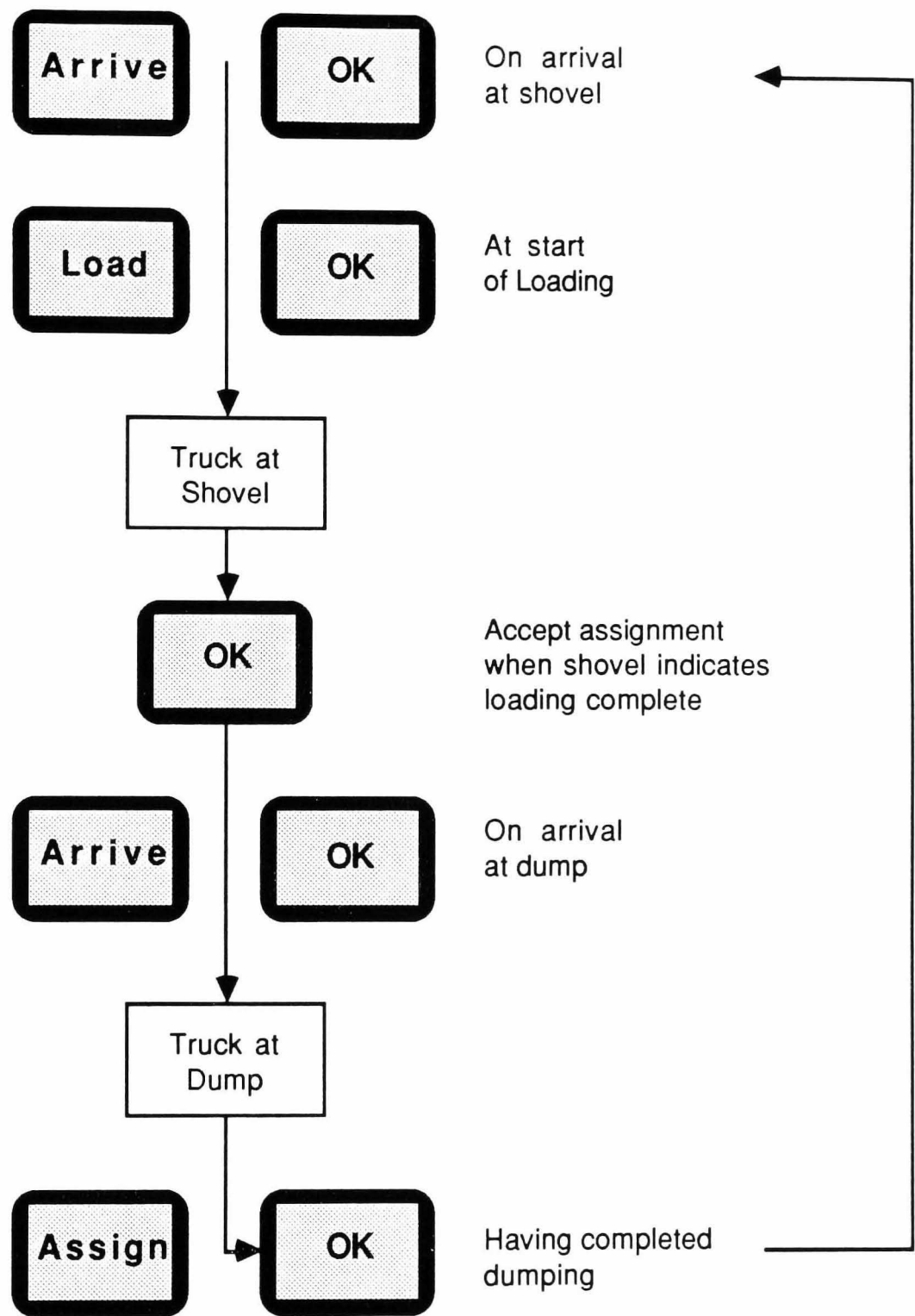


Figure 2.12 Cycle of DISPATCH Transactions for a Truck Operator (after Farrell, 1988)

the move is only allowed to take place after the last truck dispatched to the shovel has been loaded at the original location.

The control panel at the primary crusher monitors the arrival of loads, informs the system which stockpile is currently being supplemented and notifies of any crusher blockages (Batchelor, 1987).

System Developments

Analysis of the DISPATCH system versus the old manual radio system employed at Tyrone showed production increases varying between 3 and 22 percent with a mean of 11 percent. However, in addition to the changes in software organisation which had occurred in Version 2.0 (White *et al*, 1982) there was clearly room for improvement in a number of other, more general areas which were now aggravated by the problems of maintaining slightly differing systems at a number of mine sites. Version 3.0 (White & Zoschke, 1987) was to include the use of:

- alphanumeric displays for all the operators;
- PASCAL for remote computer programming;
- the radio link to tune remote computers;
- structured 'C' for all high level code;
- simulation for debugging, training, forecasting and planning;
- customised report generation;
- multiple languages without reprogramming;
- location monitoring using roadside beacons;
- vital signs monitoring.

Vital signs monitoring involves the strategic placement of sensors on a machine so as to measure such attributes as speed, temperature, pressure, voltage etc. If any current value exceeds a critical or dangerous level both dispatcher and driver are automatically alerted and appropriate preventive action can be taken (Fig. 2.13). The information can also be stored in a database giving the maintenance crew an invaluable record of machine performance (Arnold & White, 1983).

Implementation of Version 3.0 at Tyrone, brought about a further 3% increase in production with the additional benefit of 'positive relief' - a system whereby 'change of shift' driver replacement is carried out at computer designated tie-down locations hence reducing truck idle time.

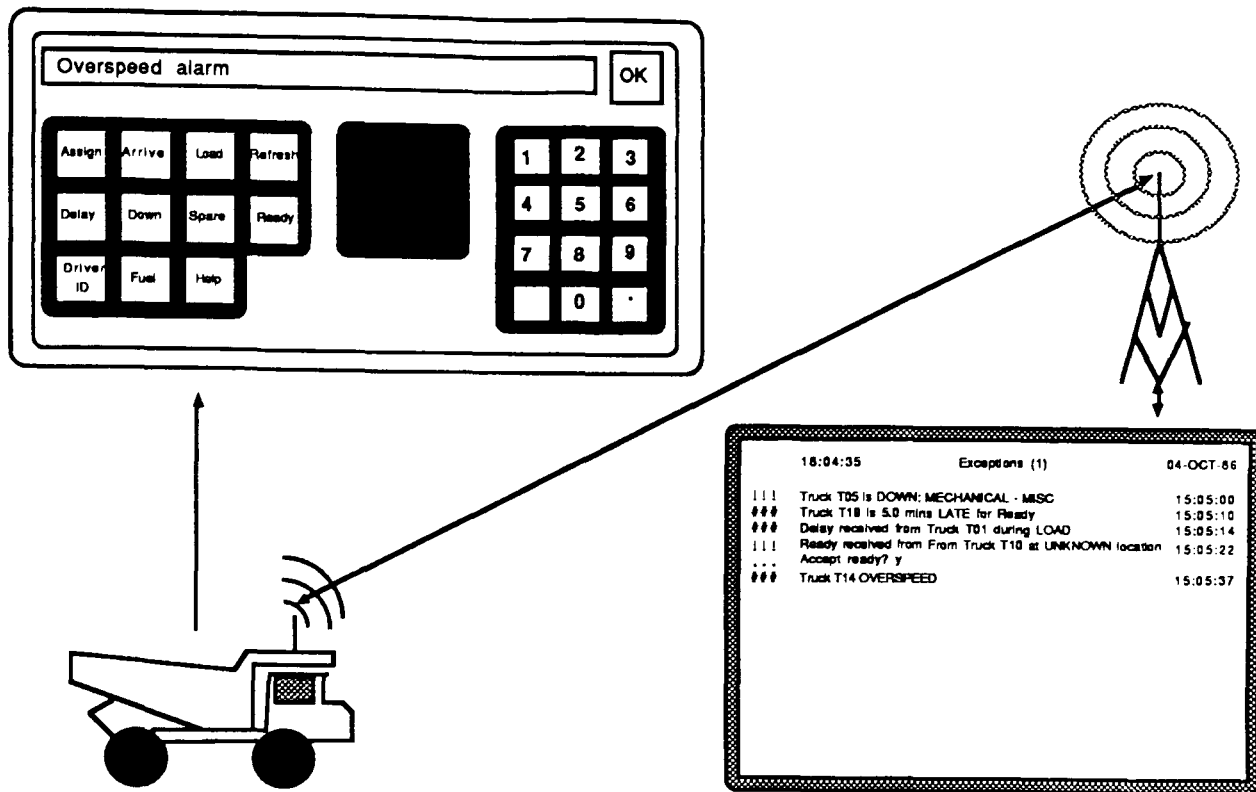


Figure 2.13 Vital Signs Monitoring (after White & Olson, 1986)

In January, 1983 (Farrell, 1988) the DISPATCH system was brought on line at a new mine owned by Quintette Coal Ltd. The planners had anticipated the difficulties of dispatching trucks in such a geographically remote multi-seam operation with a very harsh climate. Precautionary back-up, in the form of a second VAX 11/780 with 8 MB of storage capacity was to be installed and extensive use was to be made of VHF solar powered transmitting beacons to overcome the problem of fog.

Since the automated system had been introduced at start up, its benefits could only be evaluated by running the mine under a fixed allocation policy for a few days. Results showed that normal operations were being run at a 10% greater rate of production and that the application of blending control parameters to the dispatching algorithm had ensured a high quality coal product.

There were a number of problems with the Quintette system :-

- A regular temperature inversion layer and the hilly terrain caused problems with radio transmission.

- The field control panels were a major drain on the truck batteries.

A worsening signals was a good indication that there were problems with the vehicle's electrical system.

- ❑ Cheap power converters caused jamming of the radio frequencies.
- ❑ The VAX, designed to run in a controlled environment, crashed frequently due to dust.
- ❑ Rebooting one VAX in cluster with the other caused the dispatching procedure to be halted for 2 minutes.
- ❑ Separate global memory locations were required for each computer and the shared database could not be updated until one or other of the files had been closed. There was, therefore, a high likelihood of database corruption and crashes.
- ❑ Since the system had been introduced at start up, little time was available for operator training.
- ❑ Poor communications with the data centre 14km away.

Most of the hardware and software faults had been ironed out by the time Version 3.2 was introduced in May, 1984.

The MPS system used at Bougainville Copper Ltd. (Section 2.4.3) worked as planned but system changes were needed as operations evolved (White & Zoschke, 1987). The DISPATCH system introduced here included full colour graphics and again, because of its geographical remoteness, two interconnected VAX 11/780's. Teething problems during the first 6 months lead to erratic and not always favourable performance figures but within 9 months, and due largely to the fact that 95% of the source code was now common to each installation, DISPATCH was outperforming the old MPS system by 13%.

In 1983, the staff at Palabora (Batchelor, 1987) realised that there was a need for far greater control than could be achieved with the existing system (Section 2.4.4). It was, for example, not possible to determine shovel loading and crusher congestion rates nor was it possible to identify trucks individually. Detailed simulation studies predicted a potential increase in productivity of between 4.7 and 8.6 percent and the installation of DISPATCH was made economically justifiable by the ever increasing cost of fuel.

The Palabora system was to be run on a 4 MB computer with 4 operator terminals and was to make extensive use of location beacons for the reassignment of trucks already en route when shovel breakdowns or moves made it necessary. Palabora specific features included :-

- ❑ monitoring of the crusher digestion rate for blending control purposes.
- ❑ shift changes at allocated holding locations to prevent disruption of those trucks still running.
- ❑ automatic assignment of trucks to the tyre bay for retorquing after the first, third and sixth load.
- ❑ training carried out on 16k micro linked to the system.
- ❑ regular updates of the source code via a telephone link to the United States.

As a result of the system's introduction, 4 trucks were immediately placed on stand-by. This represented a 4.8% reduction in costs with no loss of production. Tests showed that truck utilisation had, in fact, increased by 7%. Batchelor concludes that further increases could be brought about by the introduction of the performance evaluation procedure (PEP). This compares current with historical performance data in order to determine those operating areas which are currently limiting production. Reports are produced hourly.

Further installations have been carried out at Carol Lake (iron), Chino (copper), Empire Iron, Bong (iron), Mt. Newman (iron), Cananea (copper) and Cerrejon (coal). Recent development work has been aimed at software enhancements which allow the user to write reports and change software logic without programming and to monitor vital signs using histogram displays.

2.6.2 Micro-Based Systems

The ever reducing costs and ever increasing capacities of micro-computers, has made them a viable alternative for the (semi-) automation of dispatching systems particularly in small to medium-sized mines.

Sadler (1988) describes a production monitor and control system (PMCS) designed to run on 3 IBM PC's (or equivalents) and claims a potential cost saving of between 10 and 12 percent due to real time decision support and lower equipment and manpower requirements. He suggests that the greatest concern is the time and effort required to keep mine data up to date and at the degree of accuracy required for the dispatching algorithm to act effectively. As a result, attention is paid to the accuracy of information received from the field and to the way it is presented. The 'acknowledger' sub-program checks field messages and, if unexpected data is received, asks for reconfirmation or corrections before time-stamping and sending

them on to the monitor sub-program. Graphical representation is carried out on a high-resolution colour monitor.

Krakiwsky *et al* (1988) describe an automatic vehicle location system (AVL) which is to include a 3-D digital terrain model of the mine's topography. The premise is that if the real time location of a truck is accurately known, it can be tied in with the terrain model and reallocated at any time hence improving productivity. The system hardware will be as illustrated (Fig 2.14) with a 512K RAM PC compatible complete with graphics board to be used as the central dispatching unit.

Research has been carried out into the optimum method of locating trucks. The roadside sensors used in most dispatching systems are widely dispersed and leave large 'blind' stretches. One alternative is dead reckoning where the information received from sensors is used to determine the position of a truck relative to its starting point. Another is global position sensing (GPS), which makes use of satellites to find the position of mobile receivers relative to a stationary one situated at the dispatch centre. Though an accuracy of ± 2 m is claimed, the limited constellation of available satellites means that GPS must be used in conjunction with dead reckoning. In any case, the accuracy that can be achieved using digital aneroid altimeters, doppler speed logs and directional gyros as used in aviaional auto-pilot systems combined with map-matching is adequate.

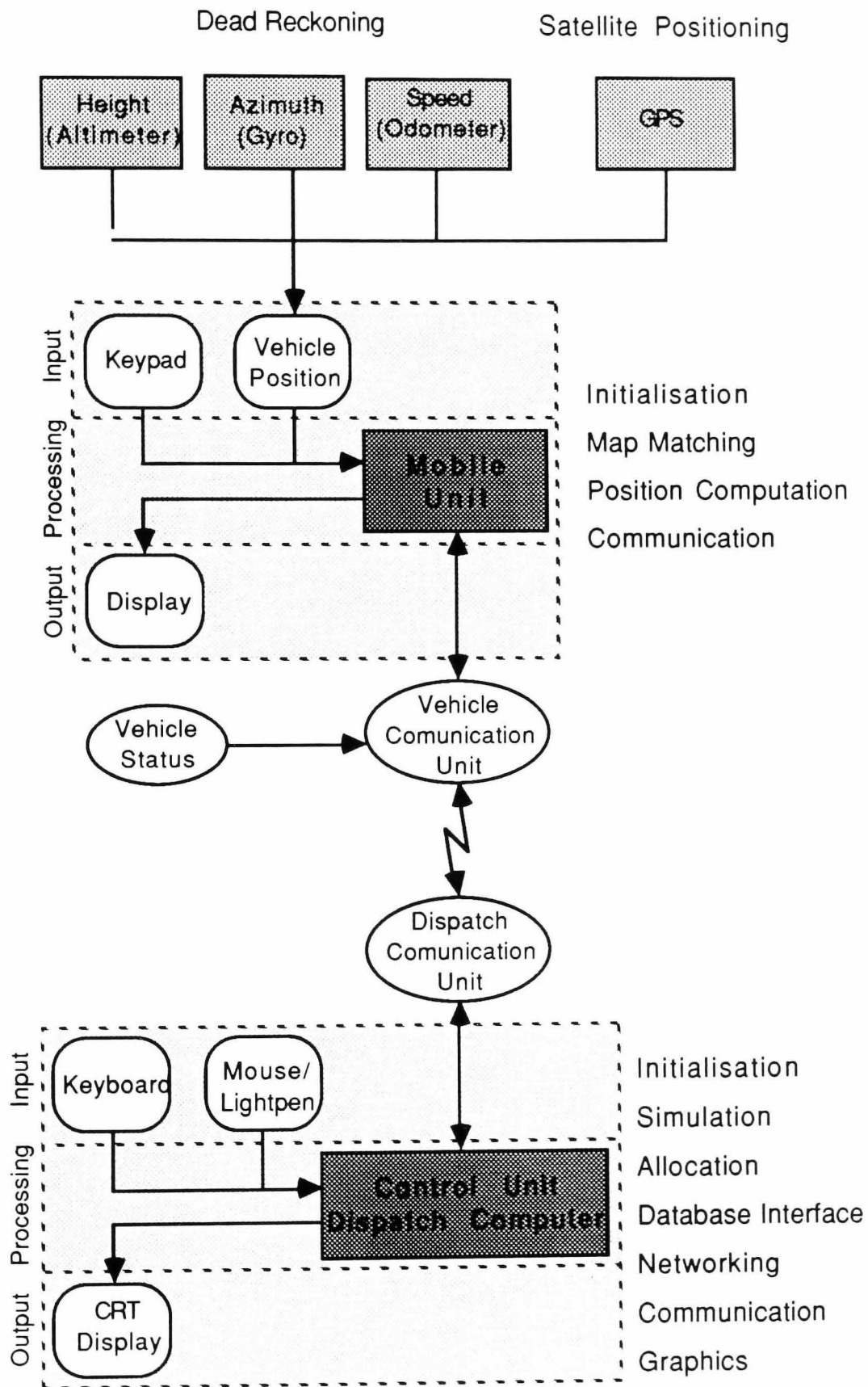


Figure 2.14

AVL Micro-based System Components

2.6.3 Summary of Automated Dispatching

Sassos (1984) summarizes the main benefits of an automated dispatching system as follows :-

- Increased mine production;
- Greater utilisation of equipment;
- Increased level of control;
- Dynamic control of ore quality;
- Reduction of delays at shift change and lunch periods;
- Ability to choose best production schedule;
- Accurate, up-to-date reporting;
- Safety in treacherous conditions;
- Improved training and forecasting through simulation;
- Management by exception :

By handling all routine allocations automatically, the dispatcher is free to spend time dealing with unusual events such as breakdowns or shovel moves.

2.7 Advanced Dispatching Techniques

It has been shown, over the last two chapters, how operations research techniques, in particular simulation, can be used to analyse mine haulage systems prior to their implementation. Unfortunately, mines do not stand still. Their layouts and fleet availabilities change frequently and operating objectives must be updated to ensure that the grade of material produced continues to fall between the narrow limits demanded. Dispatching trucks, then, is a complex optimisation problem which requires continuous monitoring of route selection and equipment status.

Linear Programming (LP) is the best way to 'optimise' a system while ensuring the efficient distribution of available resources to their various activities. However, truck/shovel route selection is a dynamic process and must be carried out based on the 'current' pit situation. Dispatching policies must, then, be applied in the short term to attempt to enforce the LP solution with optimisation of shovel and truck utilisation as a second main objective.

2.7.1 Linear Programming

Mining engineers are often faced with the problem of selecting one from a large number of alternative solutions. The decision must be based on the overall aim of the project (the objective function) and each alternative will be subject to a number of limiting factors (constraints). If these are defineable, linear programming can be used to assist in the decision making process.

White and Olson (1986) and, Bonates and Lizotte (1988) have formulated linear programming algorithms for the determination of individual shovel digging rates while ensuring that grade requirements are met. The former attempts to minimise costs based on the costs of haulage and plant, the quality of the material produced and the amount of rehandle envisaged for each solution. The latter attempts to maximise overall production using the following objective function:

$$\text{Maximise: } Z = \sum_{i=1}^n x_i + \sum_{j=1}^m x_j \quad \dots(2.9)$$

where: i = shovel in ore index
 j = shovel in waste index
 n = number of shovels digging ore

m = number of shovels digging waste
 x_i = production at shovel i (to be determined)
 x_j = production at shovel j (to be determined)

subject to constraints on grade and the total hauling capacity.

Possible refinements to the basic model include the addition of a relative priority factor (P) for ore production:

$$\text{Maximise: } Z = P \sum_{i=1}^n x_i + \sum_{j=1}^m x_j \quad \dots (2.10)$$

or a waste/ore ratio (R) to ensure that adequate stripping is maintained:

$$R \sum_{i=1}^n x_i - \sum_{j=1}^m x_j = 0 \quad \dots (2.11)$$

Further description of the shovel production LP model is beyond the scope of this project.

Having determined optimal shovel production rates, the next objective is to minimise the haulage requirements needed to meet them. White *et al* (1982) generated an LP formulation aimed at minimising the number of trucks required for shovel coverage since this amounts to the same thing.

For example, consider a very simple hypothetical mine with 3 shovels: one digging ore; one digging waste and one digging leach material. The material excavated from each loading point must be hauled to a crusher, a waste dump and a leach dump respectively but, after dumping, a truck can return to any shovel. Hence, the mine consists of 6 nodes, 3 of which (the shovels) are rate limiting and there are 12 feasible paths between them (Fig. 2.15). If a fixed dispatching policy was employed, the time taken for a truck to complete an ore/leach haul cycle would be 24 minutes. With a 3 minute loading time and one truck queuing, $24/3 + 1 = 9$ trucks would be required for each material type. Similarly, 7 trucks would be required for the haulage of waste making 25 in total.

In Figure 2.16, the example has been 'optimised'. Due to the proximity of the ore shovel to the leach dump and of the leach shovel to the crusher, trucks take a 'figure of eight' path round this area. Meanwhile, a fixed allocation is retained for waste haulage because of its rather isolated location. In this case, the time for a

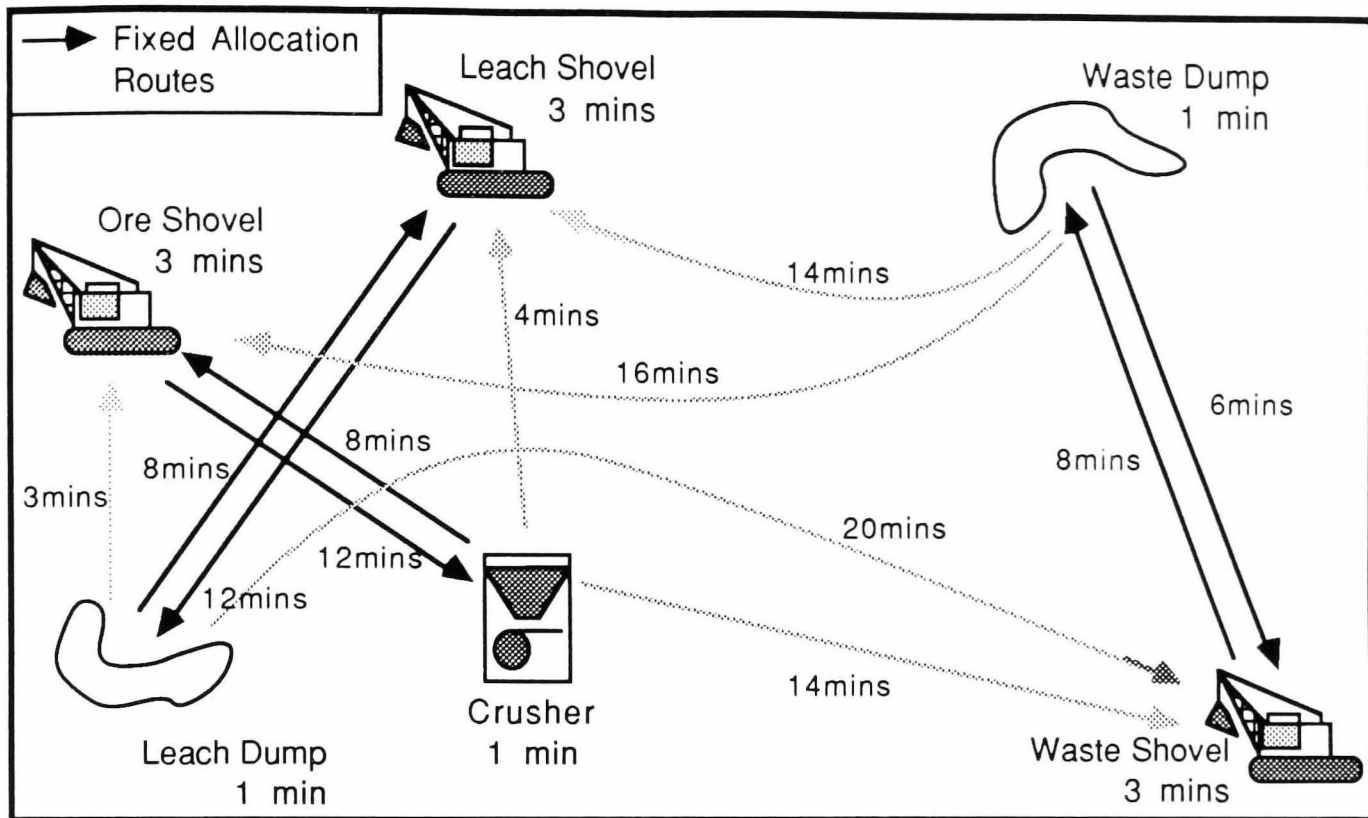


Figure 2.15 A 6-node Example Showing all Feasible Paths

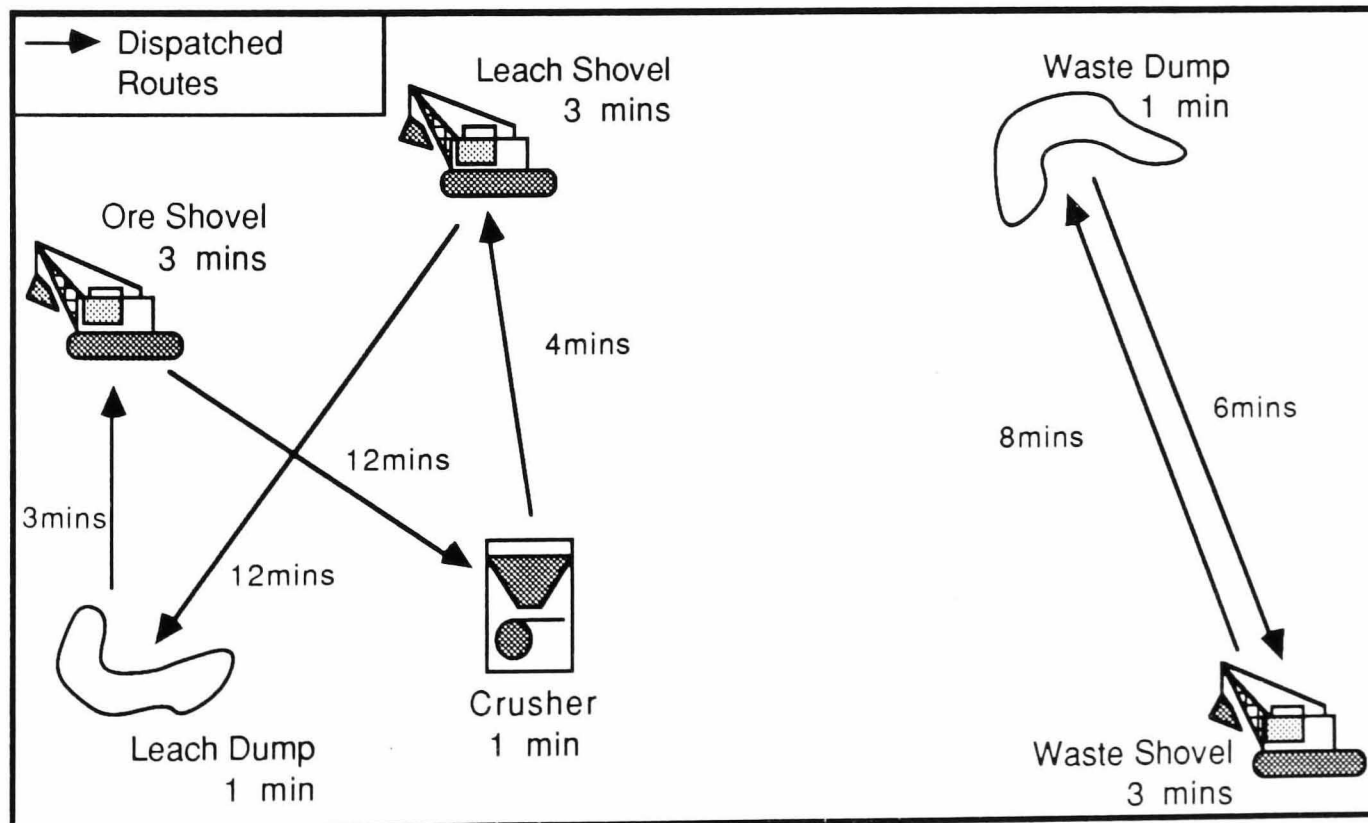


Figure 2.16 The Same 6-node Example 'Optimised'

complete circuit of the ore and leach shovels is 39 minutes which, with one truck queuing at each shovel, would mean that 15 would be required to serve this area. The total number required would be 22 - a significant saving of three trucks.

Mathematically, this problem is formulated as follows :-

$$\text{Minimise: } N_t = \sum_{i=1}^{N_p} P_i \times T_i + \sum_{j=1}^{N_d} P_j \times T_j + N_l \quad \dots (2.12)$$

- where: N_t = number of trucks (to be minimised)
 i = path index
 N_p = number of feasible paths
 P_i = mean rate over path i (trucks/min)
 T_i = mean travel time over path i (min)
 j = dumping point index
 N_d = number of dumping points
 P_j = mean input rate to dump j (trucks/min)
 T_j = mean dumping time at j (min)
 N_l = number of rate limiting loading points

Subject to the constraints of continuity at each node k :

$$0 = \sum_{k=1}^{N_{pi}} P_k - \sum_{k=1}^{N_{po}} P_{k'} \quad \dots (2.13)$$

- where: k = node index
 N_{pi} = number of feasible input paths at node k
 N_{po} = number of feasible output paths at node k
 P_k = mean rate over input path k (trucks/min)
 $P_{k'}$ = mean rate over output path k (trucks/min)

of limiting rates at each shovel:

$$R_l = \sum_{l=1}^{N_{pl}} P_l \quad \dots (2.14)$$

- where: l = loading point index
 R_l = Limiting rate at loading point

and of non-negativity:

$$P_i \geq 0 \quad \dots (2.15)$$

In general, rates of loading will exceed rates of dumping considerably. If a dumping point becomes rate limiting, such as when a crusher is slow to consume material, the mine's overall production will decline. In the event of a shovel excavating a number of different types of material at the same time, the shovel must be treated as a number of individual units each digging one material type.

2.7.2 Dynamic Programming

White and Olson (1986) demonstrate how the above LP model is adapted for use by the current DISPATCH system. Flow rates are considered in terms of m³/hr rather than trucks/hr to allow for different truck sizes, limiting flow rates are assumed to exist at stockpiles as well as at shovels and the model is tied in to the pre-determined blending requirement at each dumping point by the additional constraint :

$$P_j = Q_i \quad \dots (2.16)$$

where Q_i = Required input to plant (m³/hr) as determined from the 'minimise costs' LP model.

However, this formulation merely calculates required flow rates - the problem of assigning trucks to shovels in order to enforce the LP solution remains.

Bellman's Principle of Optimality states that:

'Whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision'

When dispatching trucks, the allocation decision must be made instantaneously based on the LP optimised paths but regardless of the previous truck assignment. A dynamic programming algorithm can be formulated for this purpose:

The scalar objective function, S, for assigning the jth truck to the ith path is:

$$S = \sum_{j=1}^{N_{tk}} \text{Min}\{S_j; H_{imax} - H_{ij}\} \quad \dots (2.17)$$

- where:
- N_{tk} = number of trucks expected to request an assignment in the near future,
 - H_{imax} = maximum haulage allocation required by the ith path
= $P_i \times T_i$,
 - P_i = path flow rate (m^3/hr),
 - T_i = travel time for path i (hrs),
 - H_{ij} = Remaining haulage allocation for the ith path at the time the jth truck would be assigned to it (A_j)
= $H_{i0} - P_i \times (A_j - L_i)$,
 - A_j = Expected assignment time of jth truck (hrs),
 - L_i = time last truck was allocated to ith path (hrs),
 - H_{i0} = Haulage allocation at time L_i

For the assignment of the jth truck to increase the total haulage allocation function S, $(H_{imax} - H_{ij})$ must be maximised. Hence, for each truck:

$$H_{imax} - H_{ij} > 0 \quad \dots(2.18)$$

$$\Rightarrow P_i \times T_i - H_{i0} + P_i \times (A_j - L_i) > 0 \quad \dots(2.19)$$

rearranging gives the expected assignment time of the jth truck:

$$A_j > L_i + H_{i0}/P_i - T_i \quad \dots(2.20)$$

A truck allocated to the ith path before the time represented by the right side of Equation 2.20 contributes nothing to the total haulage function but one which arrives just afterwards will be processed soonest. DISPATCH, therefore, continuously scans the list of paths to find the one which will require an allocation soonest

$$(\text{Minimum } X_i \text{ where: } X_i = L_i + H_{i0}/P_i - T_i) \quad \dots (2.21)$$

and selects the truck which is expected to become available closest to (but after) that time. If the truck is in the same region as the selected path, it gets allocated to it, increasing its H_{i0} , otherwise the next path is tested.

When an unallocated truck becomes available, the path with the current minimum objective function in the region, X_k , is normally selected. In order for a

truck to be assigned to a shovel outside its current region, the path, X_i , must also overcome an arbitrary cross region damping factor D i.e.

$$X_i + D < X_k \quad \dots (2.22)$$

for all paths, k , to shovels inside the region.

A small value of D provides for a more uniform coverage of shovels; a large value tends to minimise distances travelled at the expense of uniform coverage.

2.7.3 Summary of Advanced Dispatching Techniques

Linear programming techniques are well understood and have been applied extensively in those mines which have opted for comprehensive (and costly) systems such as DISPATCH. A major problem area encountered, and the reason why certain mines have found it difficult to justify the implementation of an automated system, has been the costs involved in creating and continuously updating a valid model of the operations (Sadler, 1988). If a degree of uncertainty about the accuracy of input data exists, the validity of any linear programming decisions decreases.

Bonatez and Lizotte (1988) point out the limitations of trying to optimise a linear objective function with linear constraints when dealing with a problem which is not, strictly, a linear one and certainly not a steady-state one. These enforce the fact that LP models can only be used as a guide on which heuristic or dynamic programming policies are based. One further limitation has been the speed of handling complex real-time calculations on the computer to the extent that formulations have had to be simplified at the expense of accuracy and, usually, effectiveness.

Assuming that today's computers are able to deal with this degree of complexity, the principle aim must be to ensure that the dispatching policy chosen is the one most likely to optimise production at a particular mine while meeting predetermined, and presumably correctly optimised blending requirements. The dynamic programming approach gives the dispatcher sufficient flexibility to achieve this within the confines of a particular configuration - the question remains, How?

2.8 Conclusions

The selection of trucks to serve a set of loaders in an open-pit mine is a complex decision conventionally based on route haul times which, if not easily obtainable from real data, must be calculated using deterministic simulation. This gives, however, a simplified view of the mine which takes no account of the probabilistic nature of the problem and the resulting interaction, bunching and queuing of individual units. Stochastic simulation of alternative proposals was recognised as the most accurate way of examining these effects.

While these changes to the planning process were becoming accepted, some larger mines started looking at ways of improving the limited degree of control which could be maintained over their existing truck fleets using manual radio communication. Rapid advances in computer technology during the late 1970's resulted in the introduction of (semi-) automated systems with the principle aim of reallocating trucks in the event of shovel moves or breakdowns. They also facilitated accurate and up to the minute report generation. The reallocation of trucks to shovels on a 'trip by trip' basis was now a relatively small step to take and tests on various dispatching policies carried out using probabilistic simulation showed that potential increases in production of up to 20% were possible. Several successful installations world-wide have created further interest and dispatching is now an accepted proven technology.

Problem areas include the customisation of the dispatching system to a particular mine's configuration and requirements, and the amount of time and money involved in formulating and keeping up to date an accurate model of the operations for effective planning and decision making. A fundamental requirement here, is the availability of an efficient database system able to store and provide easy access to all the diverse information involved.

Due to cost considerations, small to medium-sized mines have found that semi-automated systems are an adequate alternative to full scale implementation - the dispatcher is, in any case, more able to deal with a small truck fleet than with a large one. At the larger mines however, every effort is currently being made to reduce the inefficient 'human' factor. Investigations into on-board condition monitoring, payload control and record keeping, inertially guided driverless trucks and satellite based location monitoring continue.

Chapter 3

NUmine - A Computer Aided Mine Design and Planning System

Chapter 3

NUmine - A Computer Aided Mine Design and Planning System

3.1 Introduction

Chapter 1 includes a brief overview of NUmine. This chapter describes it in greater detail with particular attention to the hardware and utility software which is central to the system (Section 3.2). Personal involvement with the development of the user interface (Section 3.4) is the main reason why it has been discussed in greater detail than the database management system (Section 3.3). The graphics package follows in Chapter 4.

It will become apparent how much emphasis has been placed on robustness and user-friendliness. These not only reduce the reluctance to utilise the system of potential users who are not computer literate, but also speed up the 'hands-on' operating time of those who are. Another important feature is the flexibility which can be achieved when manipulating data to suit particular applications. This is fundamental to the ultimate objective of complete integration.

3.2 The NUmine Environment

NUmine is a micro-computer based system. This is because of the significant cost advantage which micro's enjoy over mainframe and mini-computers and because of the degree of interactive control which can be achieved during a program run. Throughout its development, emphasis has been placed on the optimisation of memory utilisation to ensure that the substantial requirements of such a large project remain within the limits of the micro computer. Fortunately, the rapid increases in the power to size ratio of the micro which are now taking place will allow development to continue indefinitely and at an ever increasing rate.

3.2.1 Hardware

The NUmine system has been developed on Hewlett Packard 9000 Technical Workstations (Plate 3.1). These consist of:

- a micro-computer in the series 200 or 300 range with 2 megabytes of internal memory;
- a hard disk unit complete with a built-in 3.5 inch flexible disk drive;
- a monochrome alphanumeric monitor;
- a combined alphanumeric and numeric keyboard; and
- a mouse - a small hand held device used for graphical location and manual selection purposes.

The computer contains a graphics card which converts graphical information into signals acceptable to whichever display device is currently being used, buffers it to compensate for differences in speed of operation and generates line segments and text characters. Though graphical output may be generated on the alphanumeric screen, NUmine requires a full colour monitor in order to make effective use of its graphics module. The two monitors used at the development stage have been a Cotron Sword and a Microvitec HL. Additional vital peripherals are:

- an alphanumeric printer (Epson FX 105);
- a digitising tablet (Benson 6301);
- and a means of transferring graphical images onto paper.

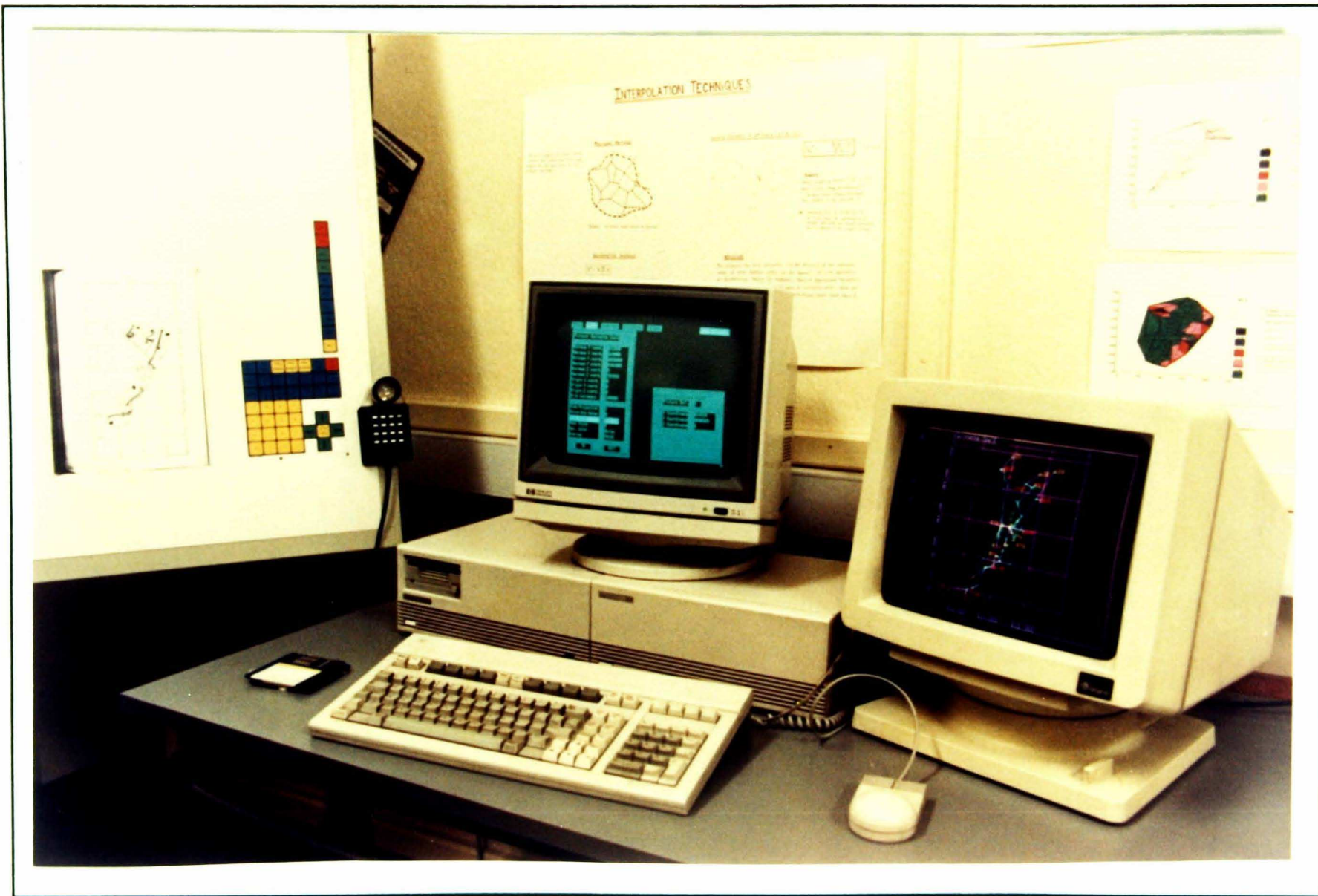


Plate 3.1

Hewlett-Packard 9000 Technical Workstation and Peripherals

The HP_7475A 6-pen plotter is ideally suited for the latter since it is not only made by the same manufacturer but is also the industry standard small plotter.

The Benson 6301 digitiser board consists of a fine mesh of horizontal and vertical conductive wires. A sine current is fed through each line and the resulting magnetic field generates a signal which is picked up by a sensor as it moves over the board's plastic surface (Benson, 1984). When a key on the sensor is pressed, the current position of the sensor is calculated and the coordinates are transferred to the computer for translation into a predetermined local coordinate system. The local coordinate system must be established before any graphical input can commence. This is achieved by locating at least three points on the board whose coordinates are known exactly. The graphics module then formulates a conversion matrix which takes into account any misalignment of the points relative to each other. One additional facility supported by the Benson digitiser is its ability to send out coordinates at frequent intervals without a key being pressed. This allows the current position of the sensor to be echoed on the graphics screen.

Table 3.1 summarises the preferred input devices for the full range of input activities associated with a computer aided design system. It is immediately apparent that the HP 9000 workstation and its peripherals provide an ideal platform for this type of application.

Activity	Preferred Devices (in order with equal preferences separated by '/s').
Selecting	mouse, joystick / tracker ball, light pen / touch screen / soft keys, function keys.
Positioning	mouse, joystick / tracker ball, light pen / touch screen, cursor keys.
Numeric Input	numeric key pad, alphanumeric key pad, digitising tablet
Text Input	alphanumeric key pad.
Drawing	digitising tablet, mouse, light pen.
Digitising	digitising tablet

Table 3.1 Preferred devices for various input activities.(after Reid, 1984)

3.2.2 Software

One of the main factors behind the decision to use the Hewlett Packard machines was that they supported an excellent PASCAL development system. There were a number of very strong reasons why this was considered important:

- ❑ PASCAL is a high level language which bears a close resemblance to written English. It is therefore easy to learn (it was originally designed to teach the concepts of computer programming) and easy to read (routines and variables can be named according to their specific purpose which enhances the ability of even someone looking at a written program for the first time to understand what is going on).

- ❑ It is a well structured language which forces the programmer to think hard about the best way to achieve his objectives. In the long run, these 'good habits' speed up development and usually result in the generation of compact and efficient algorithms, particularly when recursion is involved.

- ❑ PASCAL can be transported very easily between computers because of its standardised syntax.

In order to run any programs developed using HP Pascal, they must first be translated into a binary code format which can be recognised by the computer's MC68000 processing unit. This process is known as compilation and greatly speeds up run-time operations compared to those systems which interpret the source text during program execution. The resulting code file is stored on disk and can be executed automatically without further compilation.

NUMINE's inherent modularity was made possible by the HP Pascal Version 3.0 operating system which allows compiled program units to be placed in a library. Routines exported from any module residing in the library can be utilised by any other module in the system thereby eliminating unnecessary duplication of the many identical tasks which occur in the planning process. The linear programming and random function generating modules are two examples of utility software which are used extensively in the analysis of discrete transportation systems but are equally available to the financial appraisal programs. In a similar manner, complete applications can be accessed from within others. A good example is the NUMINE screen editor (Section 3.4.9).

3.3 NUmine Database

In computing terms, a record contains a number of descriptive elements about a particular item. These may be of any type (integer, real, string etc.) arranged in any order. The most advanced form of data storage on the HP 9000 series machines allows for the downloading and retrieval of records and this filing mechanism facilitates compacted storage of related data. However, it lacks the flexibility required whenever the same element of data needs to be utilised by a number of different applications or as part of two or more independent record frameworks. These are situations which arise frequently in planning in general and they cause problems which are compounded further in mine design. Here uncertainty, and the continuously varying nature of available information makes the use of 'static' data files unacceptable.

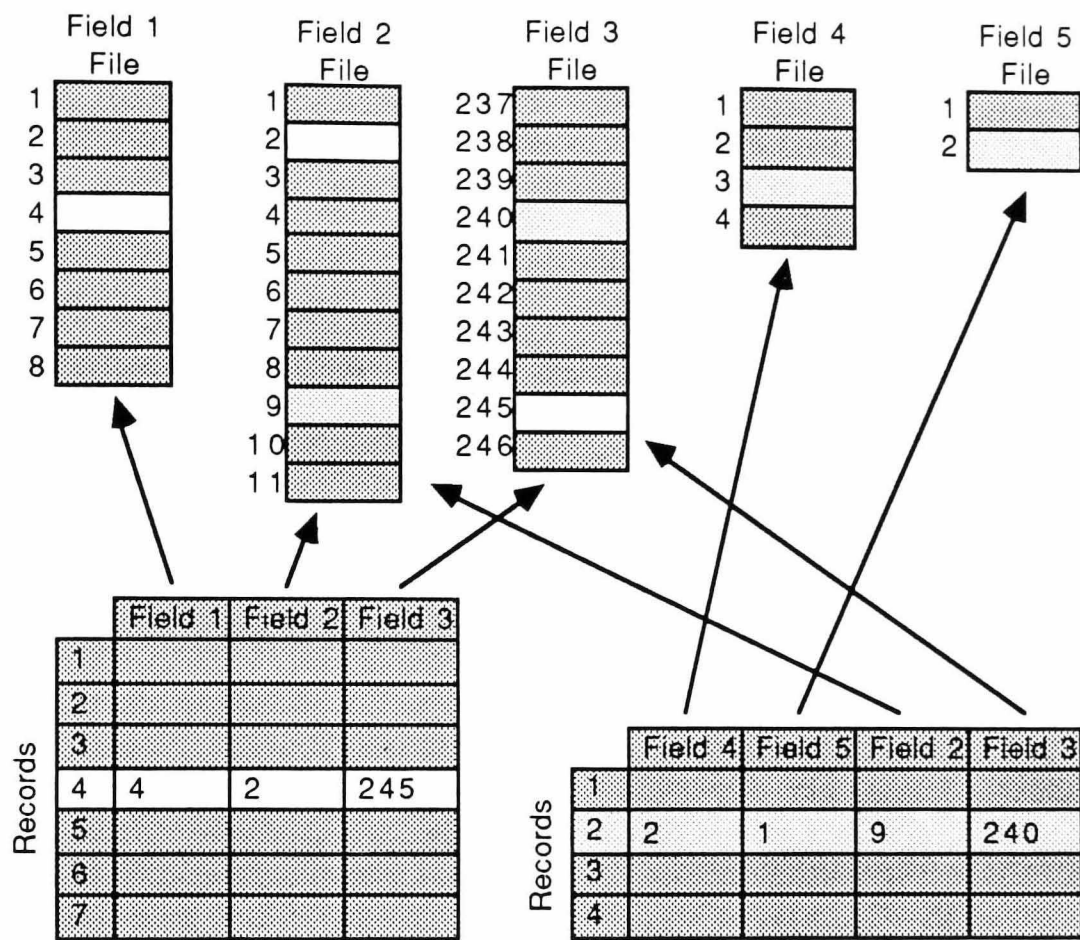
A database is an organised storage facility. Data items are still stored in files but access to them is achieved from a higher level through a series of structuring operations which make the computer's standard file system invisible to the user. This is known as the database management system (D.B.M.S.). Data manipulation becomes more systematic, the data structures are easier to maintain and reference, and standardisation allows the same information to be accessed by more than one user simultaneously. Brake and Chaterjee (1979) describe how a 'dynamic' database system, which enables the user to effect quick and frequent changes to the size and contents of stored data files, is indispensable in the haulage planning process and reduces a great deal of the programming effort.

3.3.1 Relational Database Management System (R.D.B.M.S.)

A relational database is one in which all descriptive elements of a particular type are maintained in the same storage file but independently from the elements which make up the rest of the record. The relationships between the various items are preserved in a separate file which may be altered without corrupting the data itself. This allows the same information to form a constituent part of a number of different data structures thus reducing unnecessary duplication. These structures are known as relations.

As illustrated in Fig 3.1, all records in a relation have the same format and consist of a fixed number of predefined fields. Each field is linked to one of the aforementioned storage files and when individual data elements are entered or edited, the integer returned in the relation is an index to its position in the file rather than its actual value.

Storage Files



Relations

Figure 3.1 Storage Files and Relations

If necessary, the same storage file can be accessed by more than one field in the same record (providing all the fields have a different name). This is achieved through defining an intermediate domain which links to the file indirectly. Other examples of prudent domain usage are:

- when a frequently used field element always takes its value from a definable set (e.g. loader types, colours etc.);
- when a field element can always be broken down into a fixed number of sub-elements (e.g. dates - 20/02/64 etc.).

Data elements can be added or edited either

- manually - through a tabular data interface which is displayed on the screen (Section 3.4.9), or
- directly - through a series of routines which can be called at program level.

3.3.2 Preliminary Truck Selection Example

The advantages of a relational system are best illustrated by way of example. A brief description of the type of database manipulation which would initiate the truck selection process further described in Chapter 5, follows.

The database in question contains information about a large quantity of haulage equipment. The data is primarily subdivided by equipment type since each will have its own set of descriptive elements. For example, bottom dump trucks will require exclusive information regarding width and ground to subframe clearance. Preliminary dumping investigations have shown that only rear and side dump trucks with a payload of between 90 and 120 tonnes would be acceptable.

The first task would be to append the relation containing the side dump trucks to that which contains the rear dump trucks (Figs 3.2a & b). Any fields exclusive to just one of the two relations would be added to the record definition of the other to create a new relation consisting of N fields where:

$$N = n_r + n_s - n_{(r \cap s)} \quad \dots (3.1)$$

n_r = number of fields in the rear dump relation

n_s = number of fields in the side dump relation

$n_{(r \cap s)}$ = number of fields which are in both relations

In a relational system, individual records can be selected according to the values stored in a particular field. This facility would be applied at the next stage to 'sieve' out those truck models whose payload falls outside the two specified limits (Fig. 3.2c). After similarly eliminating the remaining trucks which have any other obvious, limiting factors, the relation can be reduced to include only those fields required by the application (Fig 3.2d). However, before further investigation, the rimpull and braking characteristics of each of the remaining models are required.

Since some truck models have more gears than others, the most efficient way to store this information is by using one record for each gear in each truck (Fig.3.3). The relations used, all have the same format but, due to the sheer quantity of information, are subdivided by manufacturer as well as type. This time only those records which refer to the remaining trucks are combined in the two new relations - 'performance' and 'retarding'. Had the information been required in a single relation, the two relations could have been merged together on their common fields - the ones which refer to truck name and gear.

Rear Dump Trucks					
	Model	Manuf	G.V.W	Payload	Tip Ht.
1				68	
2				94	
58				154	

Side Dump Trucks						
	Model	Manuf	G.V.W	Payload	Width	Clearance
1				77		
2				98		
73				170		

Figure 3.2a
Two Relations Associated with Preliminary Truck Selection

Rear & Side							
	Model	Manuf	G.V.W	Tip Ht.	Payload	Width	Clearance
1					68		
2					94		
53					154		
54					77		
55					98		
131					170		

Figure 3.2b
The Two Relations Added

Acceptable Trucks							
	Model	Manuf	G.V.W	Tip Ht.	Payload	Width	Clearance
2					94		
55					77		
104					118		

Figure 3.2c
The Combined Relation Sieved

Trucks				
	Model	Manuf	G.V.W	Payload
2				94
55				77
104				118

Figure 3.2d
The Sieved Relation Customised

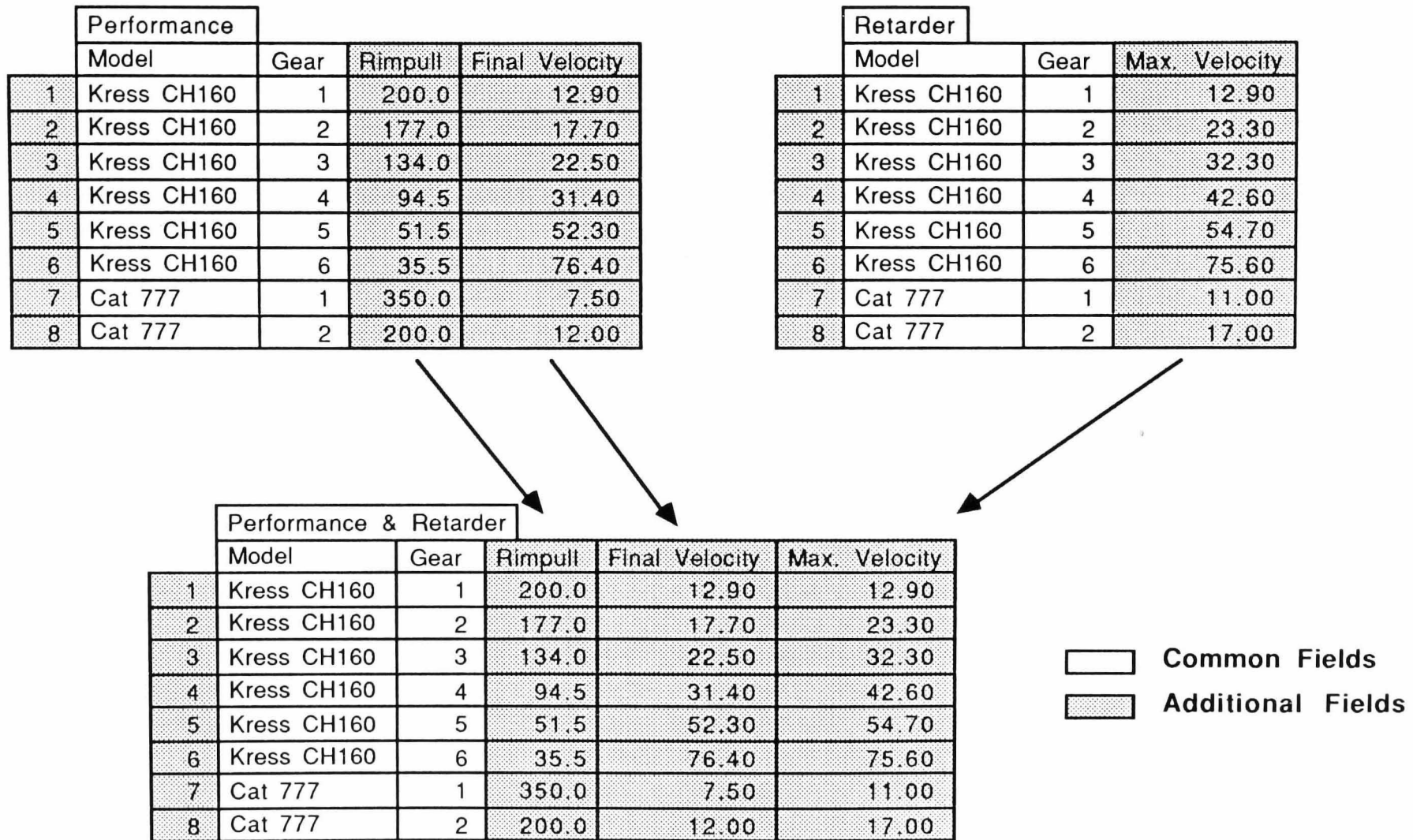


Figure 3.3 Performance and Retarder Curves Efficiently Stored and Merged on their Common Fields

These examples illustrate how the relational system allows data to be customised for a particular application. This is important when dealing with information which may be made available in any format and from a number of different sources.

3.3.3 Project Organisation

Seldom can all the information required in a planning application be stored in the same relation. More often, a number are required and, for reasons of speed and efficiency, these must be accessible simultaneously. In fact, the relations often exist in a number of independent database units. Projects form the third and outermost level of the NUmine database file organisation structure. They contain the names and host databases of the collection of relations which are required by an application and can usually be altered directly from within the application itself.

The concept of projects will be illustrated throughout Chapters 5 and 6.

3.4 User Interface

The NUmine user-interface is the mechanism by which the user is given the ability to control the flow of the program, to gain quick access to the various components of a project and, on a lower level, to enter, edit and check the validity of data. Put simply, it provides a means of communication between the user and the NUmine system.

The factors affecting the design of an efficient interface range from the user's ability to perceive, interpret and memorise shapes displayed on the screen through to the limitations of the various devices available for him to logically act upon them (Table 3.1). Reid (1984), describes the techniques applied to simplify the comprehension of displayed information, such as the correct layout and highlighting characteristics to be used in particular situations, and the ways in which information can be coded to suggest meaning or importance. For example, under normal circumstances, the user's eyes should fall naturally on the next item of interest but when an error occurs, it should be brought to his immediate attention by using such things as the sounding of an audible alarm, or the display of a flashing message. Reid also states that filling over 25% of the screen reduces the ability of the user to recognise and locate information.

The NUselect module applies some of these principles to provide a robust, friendly and helpful user-interface.

3.4.1 Display of Information

The first requirement of an efficient user-interface is its ability to display information quickly and clearly. The standard PASCAL write commands, including those which use ASCII characters to control the cursor position and the current highlight characteristics, were considered inadequate for the purposes of the NUmine system. A set of routines were needed which could provide rapid interchange between screen views and prevent corruption of the display format by attempting to show unwieldy pieces of information such as strings which overrun the screen width.

It was found that the storage and retrieval of a bit-map image of the screen contents and the ability to automatically reset the current cursor position gave the programmer a greater degree of control over what was displayed. Also, these

operations could be executed far more rapidly than the large quantity of PASCAL write statements required to achieve the same objective.

To avoid the possible scrolling effects caused by trying to write long variable strings, a method was introduced by which individual items are displayed in a box of known length. The portion of the string shown can never exceed the length of the box so by setting its start position at a safe distance from the left of the screen, the item is prevented from encroaching onto the next line.

3.4.2 Event Handling

One of the main features of the NUmine system is its use of the event driven programming concept. At the core of each application is an event-handling facility capable of responding to each keystroke, mouse movement and timer interrupt which occurs while NUmine is in operation. It does so by converting the event into a single character code which is appended to the H.P. 9000's keybuffer until it is ready to be dealt with. When the event-getting procedure is called, the next character is extracted from the keybuffer, checked for validity and passed back to the parent routine (Fig. 3.4).

This not only eradicated the screen echo of typed characters normally associated with conventional PASCAL read statements, but also, the use of characters made it possible to define the set of responses which actually mean something at a particular stage in the program. Hence, misplaced events, such as someone inadvertently pressing the reset key, were eliminated from the system.

A pure event-driven program is one in which the user is given complete control via an internal event loop around which everything else is built (Borland Int. 1986). The NUmine system was inhibited in this regard, by the HP Pascal operating system which has its own internal handling routines. The above method, which makes extensive use of the machine's keybuffer, does, however, go some way towards the desirable concept of modeless programming where, by minimising the number of points in a program where events are polled, the number of operations available at any one time is increased.

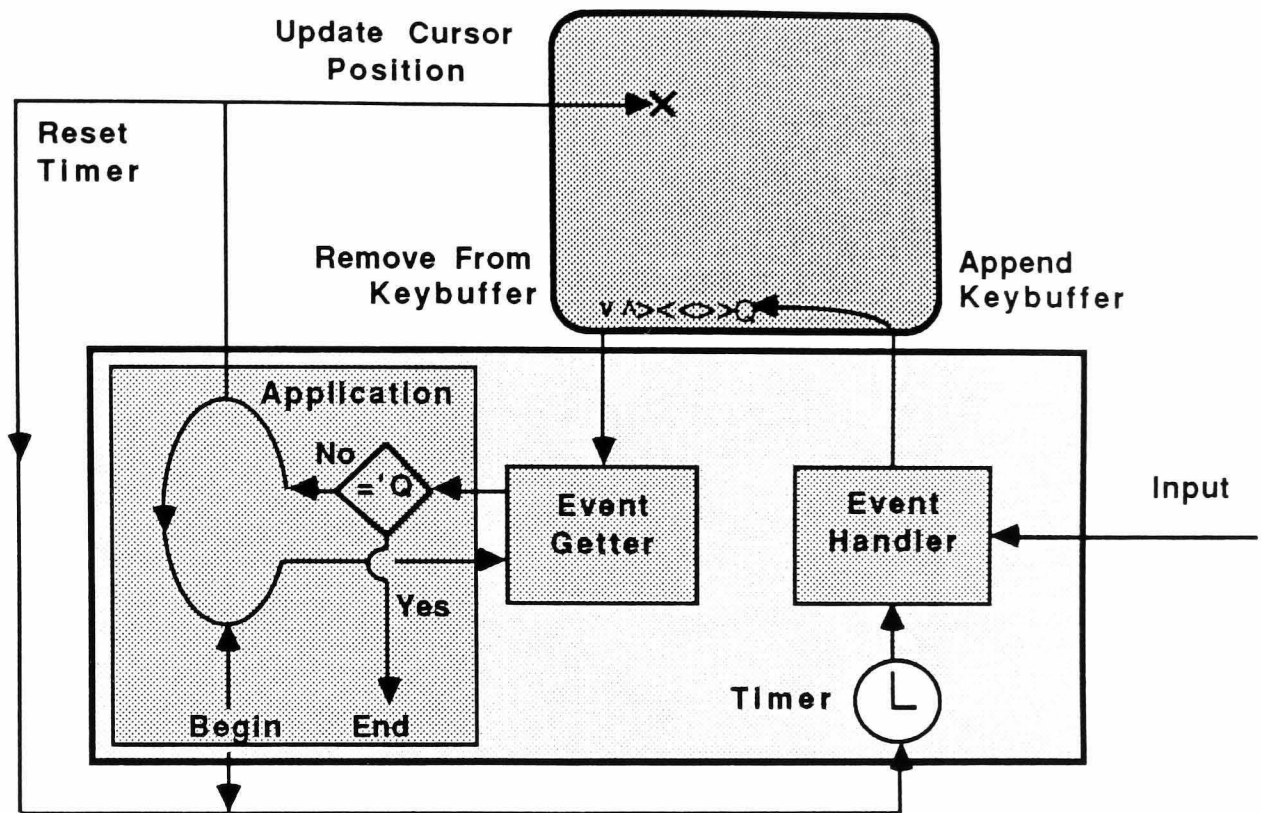


Figure 3.4 Event Handling using the Keybuffer

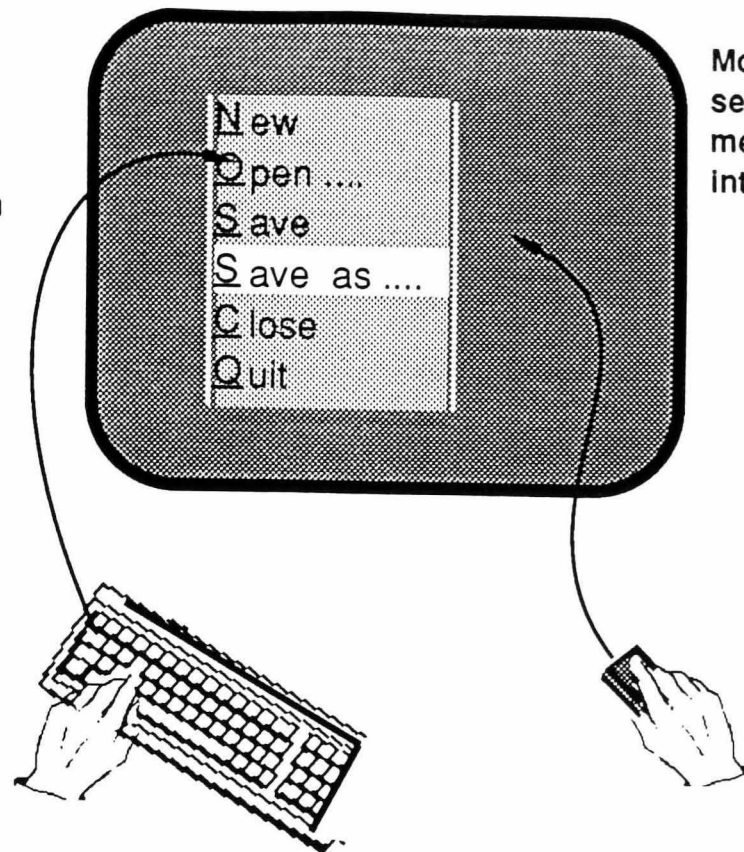
3.4.3 Feature Selection

The techniques used for feature selection can be split into two main categories (Reid, 1984): Linear command specification, which requires the user to input fully typed or abbreviated commands in order to activate a particular feature; and selection by location which involves the use of menus.

The event handling facility previously described would, if used as a feature selection technique, be classified as an advanced form of the linear mechanism. (This would be clearly illustrated if the single 'Quit-or-not' decision node shown in Figure 3.4 was extended to respond to a larger number of characters by routing off in different directions (see Section 4.4.4). However, unless the user is wholly familiar with the system, particularly as the number of available options increases, the linear approach becomes a problem of memory.

In NUmine, menus are applied whenever the user is required to select from a finite number of definable options. They are an efficient way of controlling the flow through a program and accessing specific data items

Keyboard used to select the option by typing the character which is underlined



Mouse used to select from the menu interactively

Figure 3.5 Selection of File Operations Using a Simple Menu

The reasons for this are three-fold. First, they present all the possible alternatives in a manner which avoids confusion in the user's mind; With the help of a mouse attached to the keyboard of the machine they provide an ergonomically efficient selection method and they avoid the possibility of selecting an invalid or unrecognisable option. The only disadvantage of menu systems are that, for an experienced user, feature selection is slowed down by having to physically navigate to the required destination. This problem has been partially overcome by providing an optional single character command facility for selection from the menu currently displayed. Figure 3.5 shows an example of the use of a simple menu for selection from a set of file operations.

If there are a large number of alternatives, the options should be divided into a number of related groups under appropriate headings (presenting an average person with a menu consisting of 10 or more options restricts his ability to analyse and respond rapidly). In this case, an array of menus is used to create a pull-down menu in which only those options grouped under the single currently activated heading are revealed for selection (Fig. 3.6).

The above examples show how a pull-down menu can be used to control the flow through a program and how the similarity between the standard requirements of different applications, has made it possible to adopt a standard

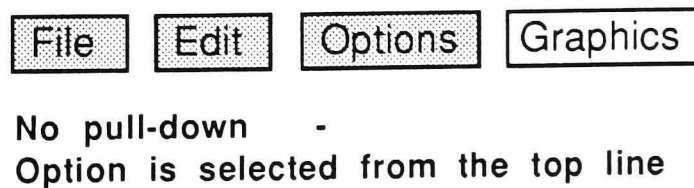
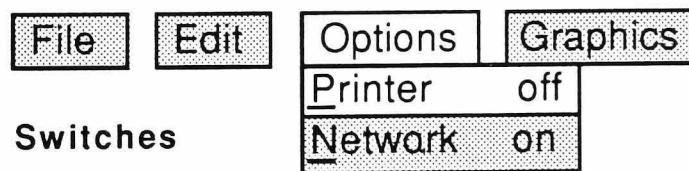
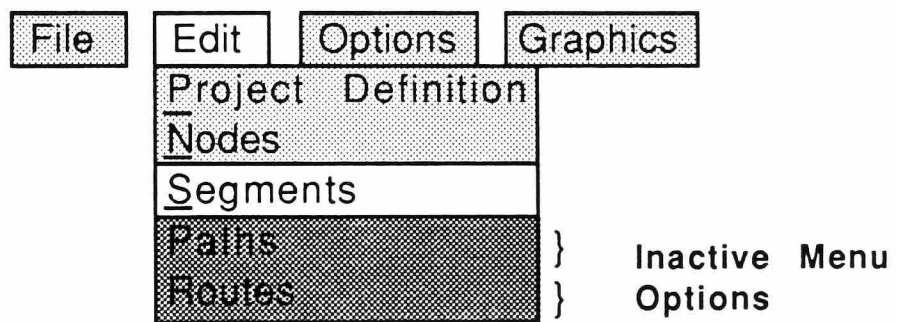
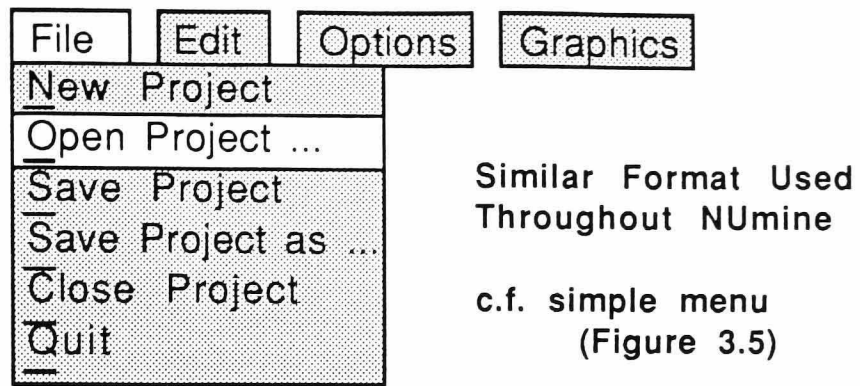


Figure 3.6 Pull-down menu for a haulage network design system.

format which will ease user familiarisation with the total system. Pull-down menus can also be used to select from a variable quantity of data items which have no particular group characteristics such as the relations in a database or the different methods of extraction that can be used in a mine (see Section 3.4.6).

3.4.4 Error Handling

Throughout NUmine, there are instances where mistakes made by the user can be detected and acted upon before the application is allowed to continue. Typical examples are during the editing of data items and in the order of execution of the program's various components. When these occur, processing is suspended while a flashing message is displayed explaining the nature of the error and, in some instances, one or more suggested courses of action. However, no matter how many checks are built into the system, there is still the possibility that a run-time error may occur causing inadvertent program termination. This eventuality does not automatically destroy the contents of the stack but does break the links between the various items of information stored and the local variables which access them.

Included in HP Pascal are two language extensions (Try and Recover) which prevent such an occurrence and the resulting invalidation of all the work carried out up to the time the error took place. By inserting these just inside the main control loop, access to the program's main pull-down menu is retained. The NUmine error-recovery routine which is used in association with these commands is also a significant aid to debugging.

When a run-time error occurs, a message is displayed on the top line of the screen indicating the nature of the error and the name of the routine or module in which the program failed (The latter is obtained from two dynamic arrays of string variables which are updated each time the program enters or leaves a major procedure or function). A list of the two 'recovery' arrays, one which contains routine names, the other modules, indicates the path taken leading up to the error. This information, combined with a list of the preceding line numbers (obtained using the HP debugging system) is sufficient for the programmer to account for and correct the error.

3.4.5 Data Entry and Editing

The above display, event handling and error checking facilities are major contributory factors to the robust nature of NUmine's data entry and editing system. With the exception of the one-byte data types, char and boolean, which can be altered with a single key-stroke, the system is based on the manipulation of characters in a string representing the specific item. The following algorithm illustrates this:

```
Initial status:   str-val is a string which represents the data item,
                  c is the character which initiated the routine,
                  If c = edit-item key (F6) then str-val is currently displayed in box
                  else nothing currently displayed in box,
                  outset is set of characters which will cause editing to terminate.
                  e.g. [SEL{mouse #1}, CR{mouse #2}, US{Up-space}, LF{Line-feed}]

begin
repeat
if c not in outset then begin
    if c = restore-char. (F7) then restore str-val to initial value.
    if c = null-char. (F8) then str-val = ' '.
    if c = delete-char. (DL) then delete character preceding the cursor position.
    if c = back-space (BS) then move cursor back one (scrolling if necessary).
    if c = forward-space (FS) then move cursor forward one (scrolling if necessary).
    if none of these then insert c into str-val at the current cursor position.
    wait for the next event which produces one of the above character codes .
    set c equal to the character code produced by the event.
end
until c is in outset
end
```

When a number is entered or edited in this way, the string obtained must be checked to confirm that it is recognisable as either an integer or real value before it can be converted back into the specific data type. Otherwise, the user is asked to correct it.

3.4.6 Windows

Any application can be broken down into a number of separate tasks. These are either directly invoked by the user or follow on as a consequence of one or more other tasks. Many require some form of user-computer interaction and often data entry or editing is involved. By subdividing the data so that it is presented to the user in chunks which roughly correspond to the tasks being carried out, the chances of losing track are reduced. Rather like the linear command specification for feature selection, trying to deal with an individual data item exclusively from the information related to it, requires experience and an in-depth knowledge of both the problem at hand and the way in which the computer is attempting to solve it.

The NUmine windowing environment allows for the interactive editing of a number of related data items on a single card or window. In general, the card's format is defined at program level by assigning characteristics to the fixed and variable parts of each window field (Fig. 3.7). The user is able to scroll around the card until the required field is highlighted whereupon its current value can be entered or edited as previously described. The facility is easy to use, robust (error checking is built in to each numeric field) and flexible (any data-type can be represented), hence, for consistency, all data manipulation is handled in this way - even when all that is required is a single data item. To facilitate this last objective, a number of functions were created which prompt the user for and then return single items from an automatically defined card.

By placing the call to enter a window inside a repeat loop, the structure of the window can be altered. This is important since there is a limit to the amount of information which can be both physically displayed in a window and taken in by the user at any one time. It also allows the system to cope with situations analogous to variable record types where the value of one field affects the structure of the rest of the record.

An additional advantage here is that the information can be imported from a secondary procedure or function which appears to be called from within the windowing operation. In Figure 3.8, this principle has been combined with an automatically created pull-down menu to greatly speed up the entry of a string and to force its value to belong to a set of excavation methods which, in turn, means that the string can be more efficiently stored as an integer referring to its position in the set.

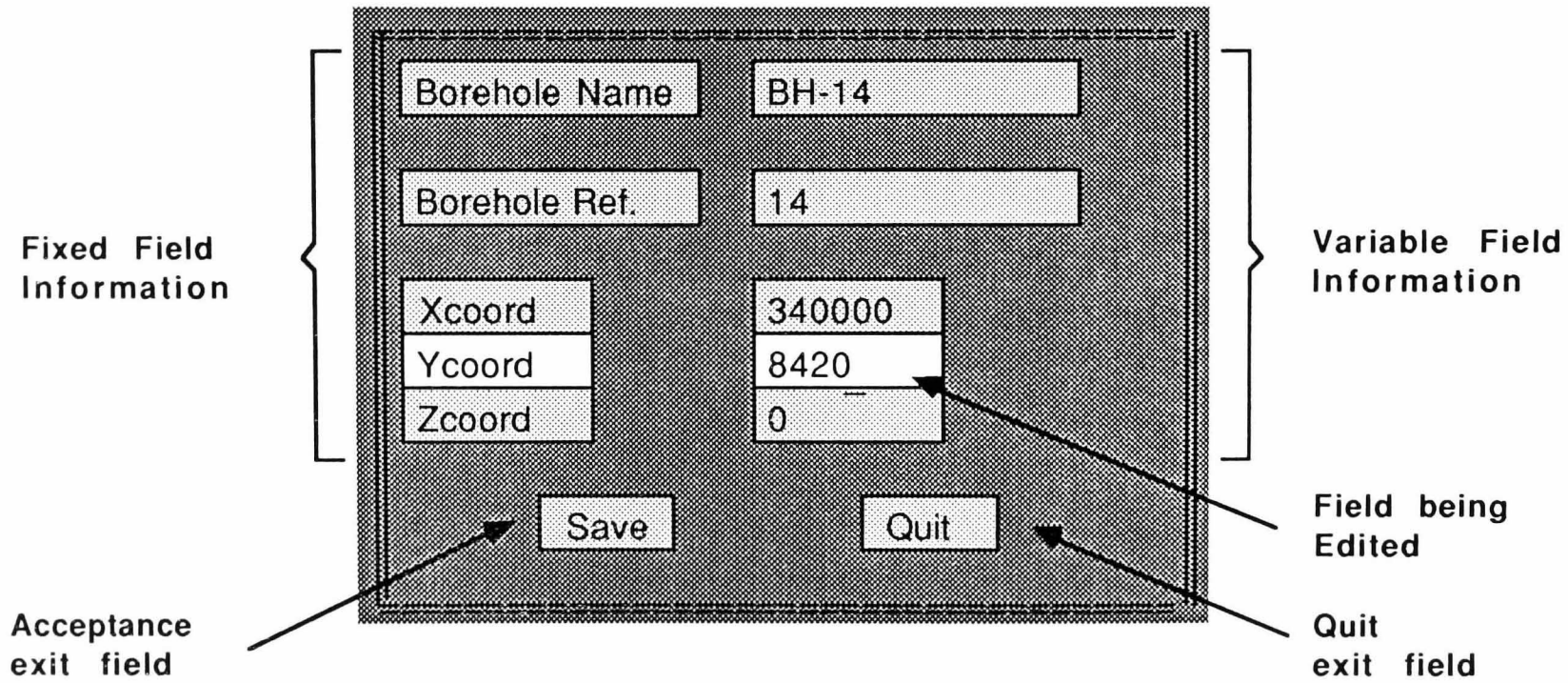


Figure 3.7 Window for Editing Borehole Coordinate Data

Item Name	Excavation Method
Field Type	i
Limit Data	TRUE
Minimum Value	0
Maximum Value	1000000000
Remarks	
Screen Field Width	
Save	Quit

String field
Character field
Boolean field
Integer fields

Window format depends on the current value of field type

Pull-down menu for selection of the required domain.

Domains	Excavation Method
Colours	
Excavation Methods	d
Excavator Types	
Haulage Methods	
Haulage Types	
Network Symbols	
Shift Pattern	
Output Devices	

Item Name	Excavation Method
Field Type	d
Domain	Excavation Methods
Save	Quit

Figure 3.8 Definition of a Database Field using a Variable Window

3.4.7 Multiple Windows

A current trend in user interface design is towards the use of multiple windows to display different batches of information on the screen at the same time. Indeed, in areas such as software development, conventional 'two dimensional' environments are now considered inadequate (Newman *et al*, 1985). An attempt was made to design a similar system for use with the NUmine window facilities already available, so that multiple records could be displayed as a stack of cards for the purposes of rapid selection and data editing. Here, records of information are accessed or manipulated by interactively selecting from a menu consisting of the first fixed field of each card in the stack.

Newman categorises two distinct 'interfaces' which must be considered when developing such a system - that which allows the program to control the windows currently displayed and that which allows the user to manipulate them. In NUmine, the two are closely linked and both are limited by the use made by the existing windowing system of conventional PASCAL write statements, as opposed to a pure bit-map displaying mechanism. This effectively reduces the alpha-screen to a grid of 24 x 30 character cells which is an insufficiently high resolution to give the flexibility required, and also slows down the process of displaying multiple windows considerably.

Add to this the increased likelihood of causing confusion in the user's, not to say the programmer's mind (particularly when dealing with variable record structures) and it is clear that there are few areas where the use of multiple windows wins over the selection for display of one window at a time. With this in mind, the latter was used as the basis for the display of individual records in the NUmine screen editor (Section 3.4.9).

3.4.8 Help Facility

The final user interface feature in NUmine is its help facility. It allows the user to request information about either the program mechanism or what can be achieved with its various components. The help messages are created and stored in completely independent text files which allows them to be updated or altered without having to recompile the module(s) to which they refer. However, they are not, simply, page after page of text. They are structured so that when the help key is pressed, the information immediately displayed in a help window is that relevant to the stage in the program which has just been reached. This is achieved by calling

the help files which refer to particular program blocks in the opposite order to that in which the blocks are themselves called - a process known as reverse nesting (Figure 3.9).

Figure 3.10 shows selected excerpts from two help files used in association with the haulage analysis problem described in Chapter 6. By setting the help file before entering the pull-down-menu, a call to help during the selection process would result in the lowermost message being displayed on the screen. Depending on whether or not this message provided sufficient information for the user to continue, he could then choose either to select one of the 5 'More' options displayed at the bottom of the window or to leave the help mode altogether.

In general, help messages are directed towards an experienced user of NUmine who needs to know what the program requires as input in order to achieve an objective rather than how to go about putting it in. However, for the benefit of others, an explanation of the latter, which may, as illustrated, be stored in a separate help file, is also made available. In the extreme case, where no help files have been built into an application, the 'How to do it' helps are the only form of assistance provided.

3.4.9 NUmine Screen Editor

The NUmine screen editor is the best example of a module which is frequently utilised from within application programs but which could just as easily stand alone as a program in itself. It is also one which makes extensive use of the user interface.

From within the screen editor, individual relations can be displayed on the screen and edited or restructured as described in Section 3.3. In addition:

- statistical information can be obtained for each field;
- data can be viewed and manipulated graphically;

and, as an alternative method of viewing the data,

- records can be viewed in isolation on a default card with controls available for individual record selection or for scrolling through the stack of records which is formed by the relation as a whole.

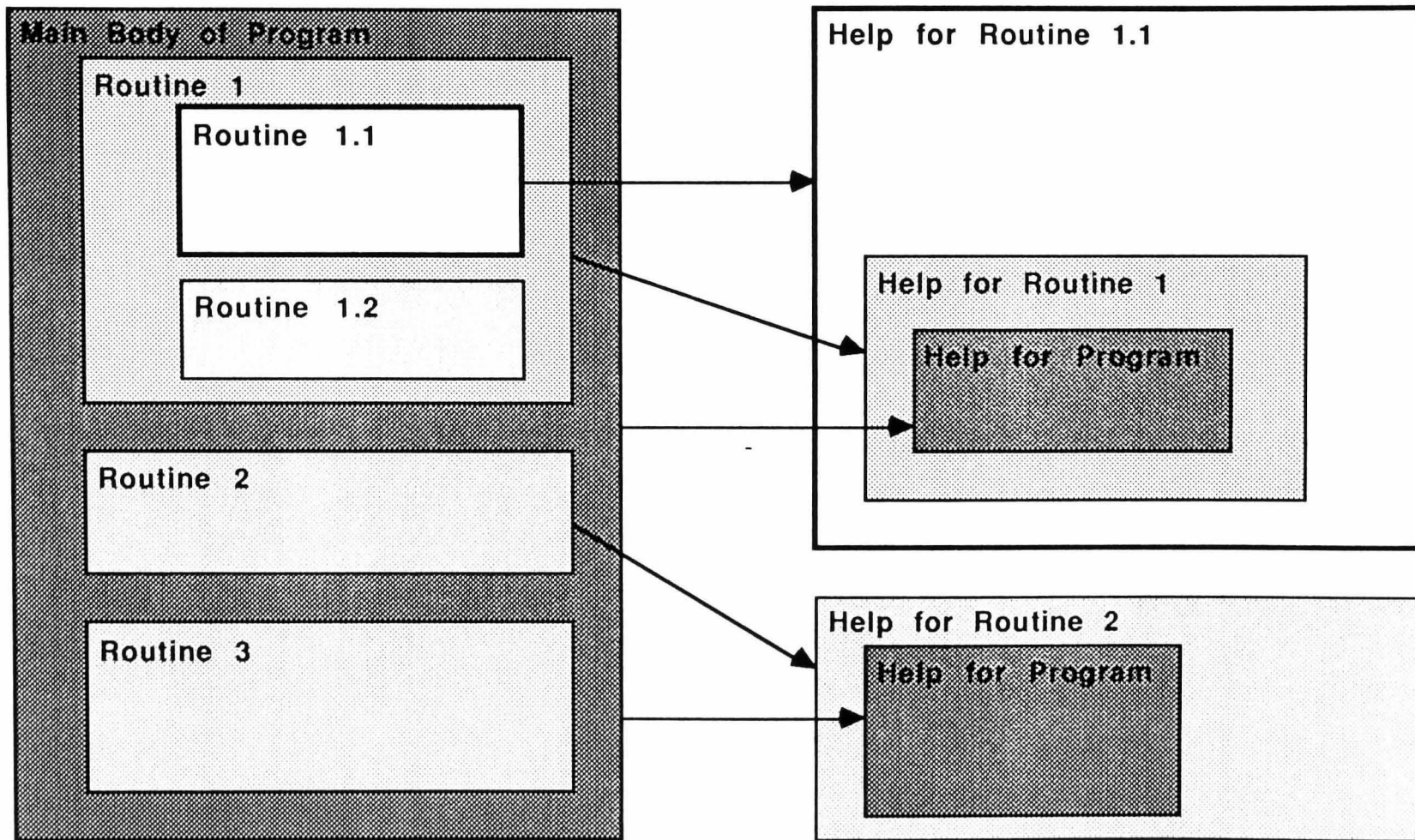


Figure 3.9 Principle of reverse nesting as applied to help files

```

=====
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
* 06 HELP for menus                                {File = '#16:selHELP.TEXT'}

Select the menu option you require
either
  a) by using the mouse or the arrow keys
      to highlight the option, then pressing
      SELECT (Mouse #1) or RETURN (Mouse #2)
or b) by typing in the character in the option
      which has been underlined.
      (only works if the option is active).

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
=====
* 01 HELP for File Operations                      {File = '#16:netHELP.TEXT'}

New Project:   Creates a new project
Open Project:  Opens an existing project
Save Project:  Saves the current project under the
               same heading
Save Project as: Saves the current project under a
               different heading
Close Project: Disposes of the current project without
               saving it
Quit :        Leaves application.

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
* 05 HELP for Haulage Network Design System

File :        Project Filing Operations.
Edit :        Edit project or relation data.
Options :    Activate (De-activate) switches.
Graphics :    Entry to NUgraph.

+ #01:netHELP.TEXT 01 More on File Operations....
+ #02:netHELP.TEXT 02 More on Editing Project.....
+ #03:netHELP.TEXT 03 More on Option Switches..
+ #04:netHELP.TEXT 04 More on Use of Graphics for Networks

+ #05:selHELP.TEXT 06 More on Menus....

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

```

Figure 3.10 Excerpts from Two Help Files used in the Haulage Network Design Example.

3.5 Conclusions

This Chapter has been used to describe the framework of the NUmine System. Features covered include the hardware on which it was originally designed to run, the central relational database management system which evolved during the course of development to incorporate useful features common to the various application areas being covered, and the general-purpose user-interface which was brought together as an amalgamation of functions initially built for my own purposes.

The RDBMS allows rapid access to all the information required in truck selection and ensures that data is stored efficiently so that the system can be run on micro-computers. The user-interface, meanwhile, provides a robust, helpful and easy to use environment in which the planner is able to concentrate on the job itself rather than on how to carry it out on the computer. These core modules were made available to every application developed within NUmine and are fundamental to an integrated planning system.

Chapter 4

NUgraph - A Graphics Package for Computer Aided Mine Design

Chapter 4

NUgraph - A Graphics Package for Computer Aided Mine Design

4.1 Introduction

Computer graphics have played an increasingly important role as the technology to produce good quality, high resolution, colour images has developed. It has found many and varied applications in the worlds of business and finance, in science and engineering and in the arts and media. Many of these have been developed for the sole purpose of putting into practice that well-worn expression - 'A picture speaks a thousand words' in that they provide a mechanism for rapid and comprehensible data visualisation. Though this is undoubtedly a very useful facility, the benefits to be gained from producing them on a computer stop after the considerations of speed and quality. The information could be viewed just as easily if it had been hand drawn on a sheet of paper.

For the most part, the examples above can be termed passive since the user has no control over the image once it has been produced. There are, however, many instances where a means of three-way communication between user, computer and the graphical image is desirable. Listed below are some of the best known examples of active computer graphics where the contents of a picture are made to change in response to his commands.

- Air Flight Simulation,
- Computer Generated Maps,
- Electronic Circuitry Design,
- Biochemical Molecular Models,
- Drafting,
- Software Development,
- Arcade Games.

With the exception of air flight simulation and arcade games, the above are also examples of Computer Aided Design.

In terms of mine design and planning, the importance of graphical visualisation cannot be over-emphasised. The vast quantity of information of different types and the high degrees of uncertainty have pointed to the utilisation of general purpose graphics software in the form of a package (NUgraph). The following are examples of the sort of varied information which such a package should be able to handle.

- Automated Surveying,
- Exploration Information: Borehole logs,
- Interactive Geological Modelling,
- Reserve Estimation: Block Models,
- Geotechnical Analysis: Contours,
Sections,
- Mine Design: Plans,
Overlays,
- Operations Design: Cut Diagrams,
- Equipment Selection: Technical Drafting,
Performance Curves etc.
- Production Scheduling: L.P. and Simulation Networks,
- Environmental Analysis: 3-D Visualisation,
- Financial: Graphical output for management
and investment analysis.

Figure 4.1 highlights the four basic requirements of a Computer Aided Design package and illustrates what happens to the graphical information as it passes through the various NUgraph modules. A high level language like PASCAL provides the standard file and interactive text facilities which are fundamental to user-friendliness. An attempt has been made to make NUgraph just as flexible. In an industry where there appears to be some resistance to technological change, it is essential that any software produced is easy to learn and use.

This chapter describes the NUmine graphics system in some detail. It begins by looking at the way in which graphical information can be displayed and combined to form pictures using the Hewlett Packard graphics library. It then goes on to explain how the main features of a computer aided design graphics package (segmentation, modelling and interactivity) have been made available within the context of an integrated planning system.

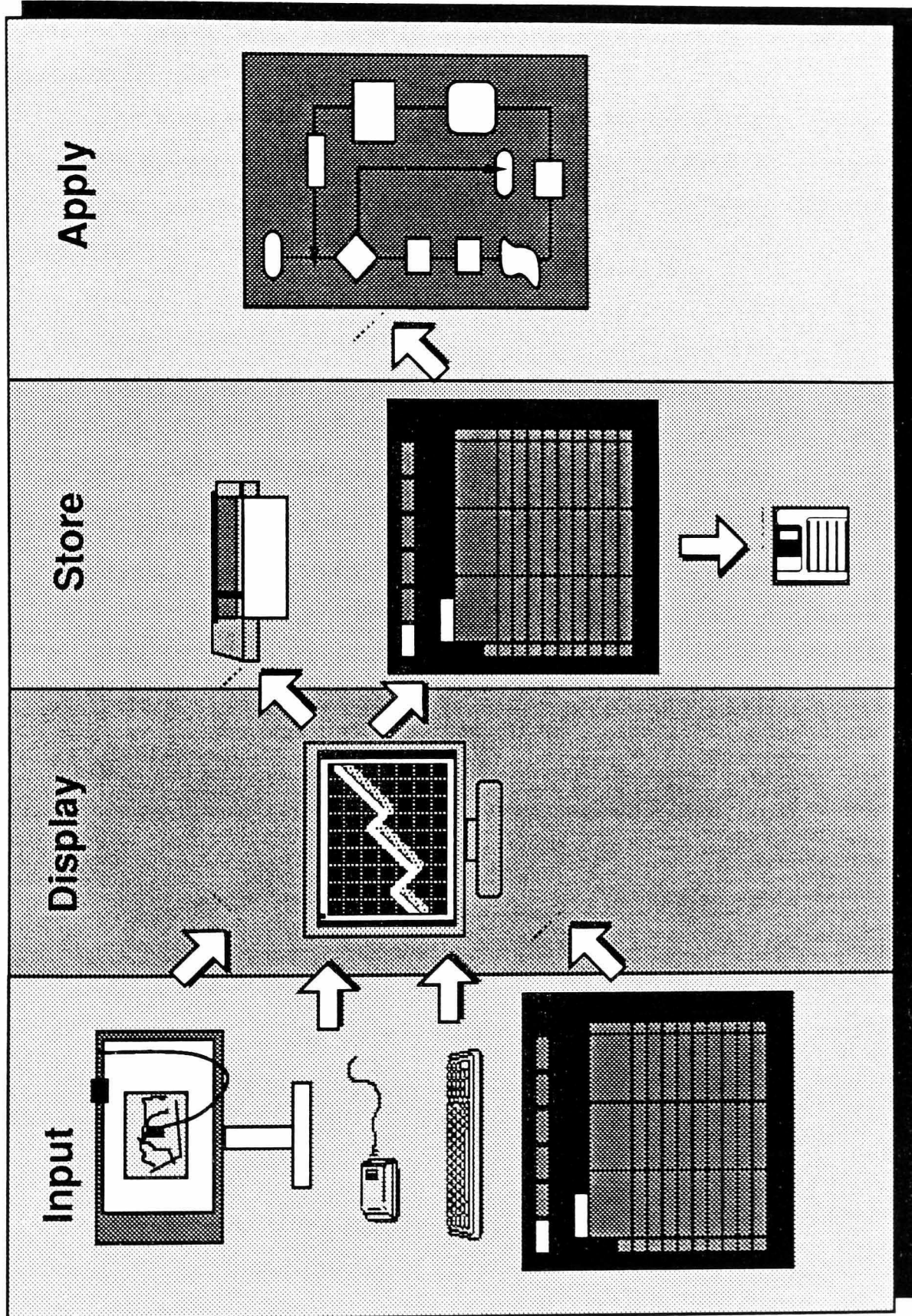


Figure 4.1 The 4 Phases of NUmine Graphics

4.2 Hewlett Packard Graphics Library

Hewlett-Packard provide a device-independent graphics library (D.G.L.) for use with the 9000 series 200 and 300 computers. The library is not exhaustive but gives the programmer the ability to develop almost any conceivable graphics operation (Hewlett Packard, 1985(a)).

The routines which make up the graphics library are based on a cartesian coordinate system of which the main limitation is the resolution of the display device used. They can be split into six main groups each covering a different area of graphics control :-

Initialisation /Termination

These provide access to the graphics system and facilitate customisation to particular external display devices. When the system is initialised, all the various graphics parameters are set to their default values.

Display Area Control

These routines are used to define the characteristics of the plotting surface: its area, its aspect ratio and its local coordinate system (Section 4.3.2). Points are mapped onto the above coordinate system by a series of transformations which can also be called independently if required.

Primitive Drawing

These routines are used to draw graphics primitives onto the current plotting surface. The types of primitive range from simple markers or lines through to text strings and polygons some of which can be drawn in a number of different ways (Section 4.3.1).

Characteristic Controls

These are used to alter the characteristics such as the colour or line style of the next primitive drawn. Each parameter remains applicable until it has been reset by a later command.

Inquiries

These can be called at any time to inquire about the current values of certain graphics parameters such as the current display limits or the type of error which may have just occurred.

Locator

These routines together provide one of the most important requirements of an interactive graphics package - the ability to control the movement of a cursor round a graphics screen.

There were a number of further limitations to the D.G.L. which made it unsuitable as a stand-alone graphics library for the NUMINE system. However, only the locator proved inadequate for complete utilisation. Routines grouped under the other sub-headings were adopted in some form although often modified or combined together to facilitate particular requirements.

4.3 Viewing Information

4.3.1 Graphics Primitives

There are 10 basic types of graphic entity in the NUmine system, most of which, it is inappropriate to describe as primitive except in the respect that they can be appended to a picture by a single command. Each fulfil a range of purposes, a sample of which are described below.

Markers

These are used extensively in the generation of maps and charts for displaying multiple instances of a particular entity or condition. They are drawn to a constant size which means that by increasing the current viewing scale, a closely grouped set of markers are more easily resolved (Section 4.3.3). Figure 4.2 shows how markers can be labelled in order to illustrate borehole positions. There are 19 standard marker styles available in the HP graphics library but these may be supplemented by any number of user-defined symbols.

Lines

These are the single most important constituent of any picture. In fact, every other entity consists of a finite number of line elements. The frequency with which they are used has been reflected in the quantity of research carried out into the way they should be plotted (Newman & Sproull, 1981). One important factor is line-clipping which is applied to ensure that only the visible section of lines which cross the current picture limits are plotted and those which lie completely outside are ignored.

Arcs

Arcs (including circles) have a limited application to mine planning except in the generation of charts or flowsheets. They exist, for completeness, in the NUmine system but the speed at which they can be plotted relative to other graphics entities detracts from their utilisation.

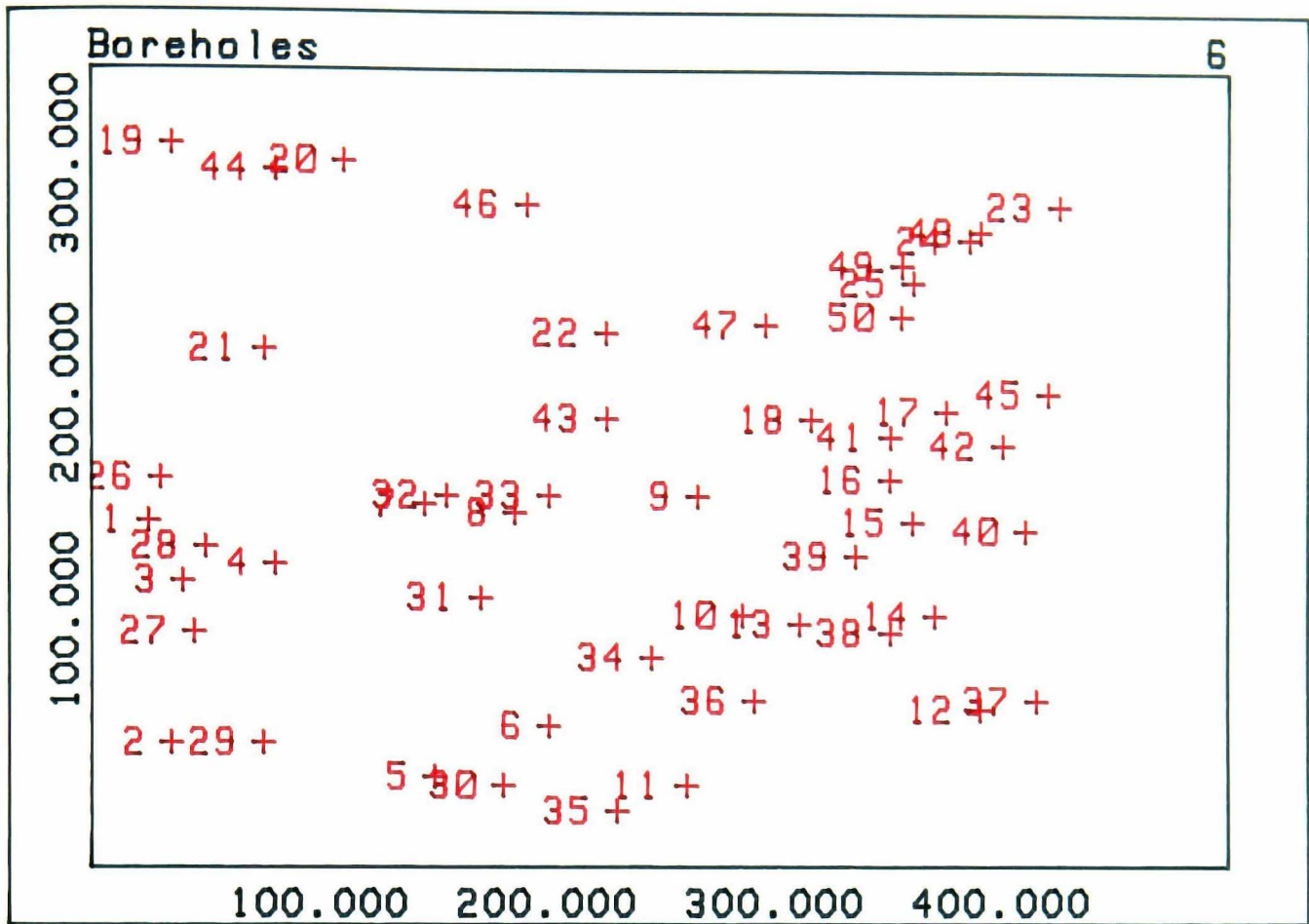


Figure 4.2 Labeled Borehole Positions

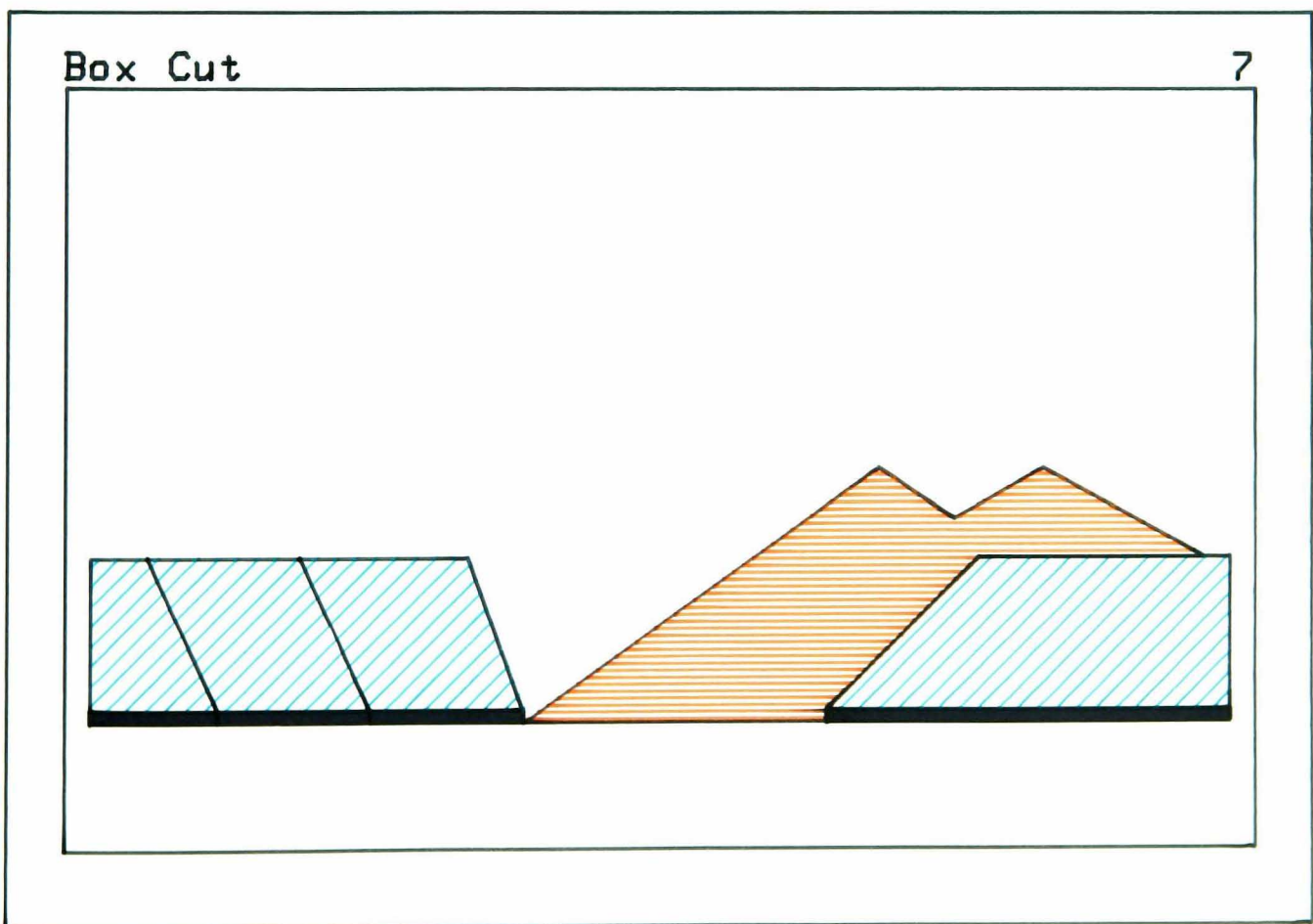


Figure 4.3 Cut Diagrams

❑ Rectangles

These, too, are used extensively in the production of charts and flowsheets but, since they are economic in terms of memory utilisation (requiring only 4 coordinate values), should also be applied in preference to polygons wherever possible. In the 2-dimensional display of block-models for example (Croghan, 1989).

❑ Polygons

Polygons are used to represent complex shapes and to fill solid areas on the screen. They play an extensive role in Computer Aided Design because of their flexibility. Like rectangles, axes, graphs and links, polygons can be referenced by name which makes them more accessible to the current application program. For example, in the production of cut diagrams for dragline selection (Fig. 4.3), the filled areas are named in accordance with the expected order of extraction. The time taken to plot shaded polygons makes the application of an efficient clipping algorithm vital (Sutherland & Hodgman, 1974). To avoid over-plotting when a picture is transferred onto paper, filled polygons must be sorted into a front to back order and clipped against each other .

❑ Text

As seen in some of the previous examples, the ability to display text on the screen is, in most cases, fundamental to the interpretation of the information displayed. In NUgraph, text can be plotted at any angle and to any scale but, unlike marker labels, remains proportional to the rest of the picture. A single text block may consist of a number individual strings which are spaced and justified relative to the box which contains them (Figure 4.4). Any characters which lie outside the box after justification are omitted from the plotting process.



Figure 4.4 Text Justification

❑ Axes (see Graphs).

❑ Graphs

Graphs essentially consist of a set of axes which define a coordinate system onto which a number of datasets have been mapped. In NUgraph, the components are defined separately to allow for their independent alteration and to avoid the duplication of any dataset which needs to be displayed on a separate set of axes.

It is possible, from within the NUmine screen editor, to plot the contents of any 2 fields against each other using the contents of a third to label the plot (Plate 4.1). It is also possible to produce frequency or cumulative frequency histograms. Once plotted, the format or fill characteristics of a dataset may be altered. Similarly the characteristics of the axes, the limits of which are, by default, scaled to fit the datasets which they currently display, can be modified to highlight particular trends.

❑ Links

As an extension to the definition of symbols, one whole picture can be included as part of another. This is achieved through a 'link'. As well as the usual colour and style controls, a link's characteristics allow the user to stipulate the size and shape of the subsidiary picture when it is displayed. For example, in the case of a map overlay, the coordinates of the two pictures can be made to coincide (Fig. 4.5). Alternatively, the facility can be used to apply transformations of rotation or scale to complicated structures which again, unlike markers, remain proportional to the rest of the picture. A distinction will later be made between temporary and permanent links (Section 4.4.3)

❑ Symbols

Symbols are user-defined markers. They are created interactively and appended to a menu-picture from where they can later be retrieved. The choice of menu-picture is made in accordance with current application (see Section 6.2.2)

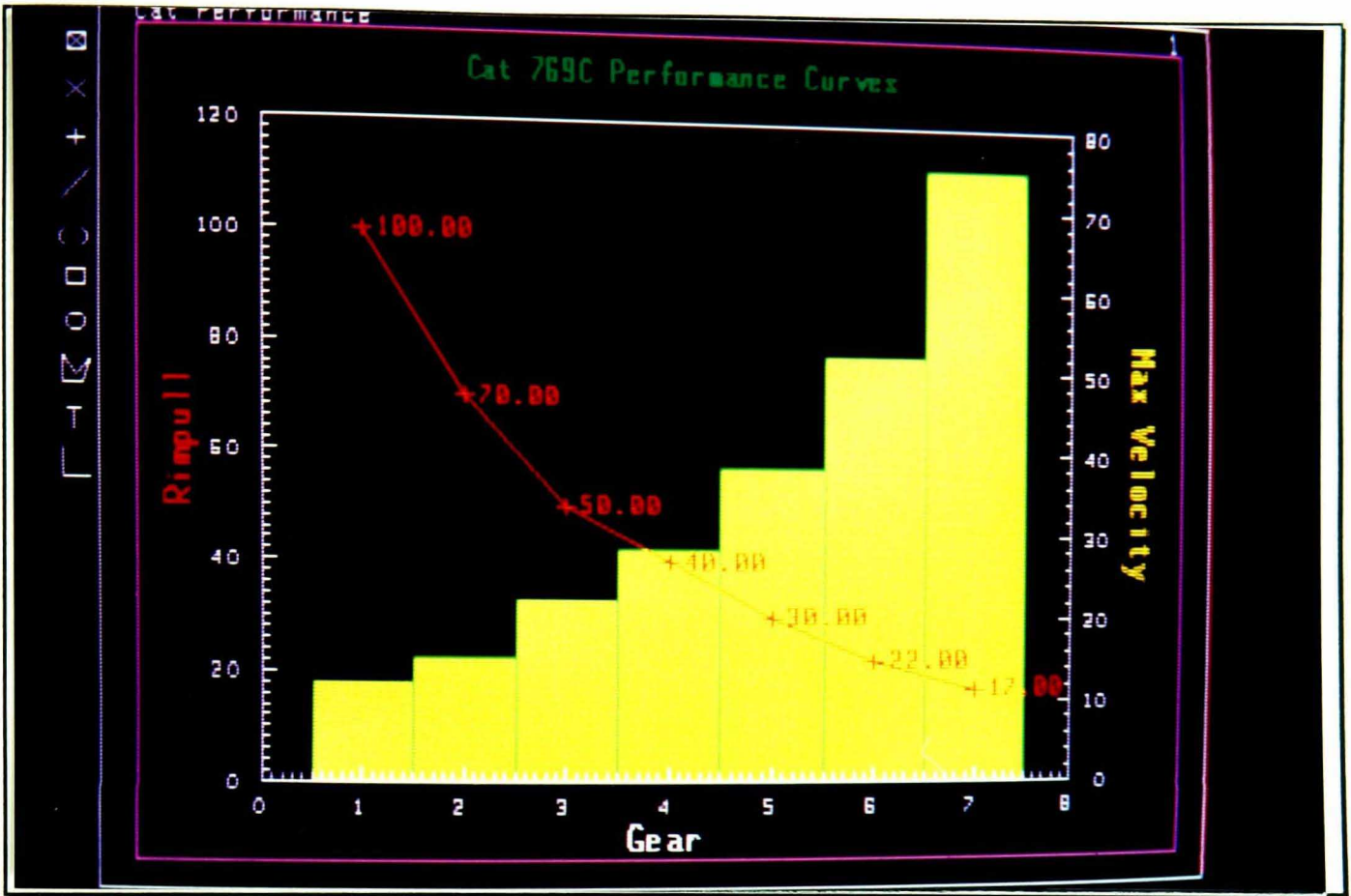


Plate 4.1 NUmine Graph Plotting

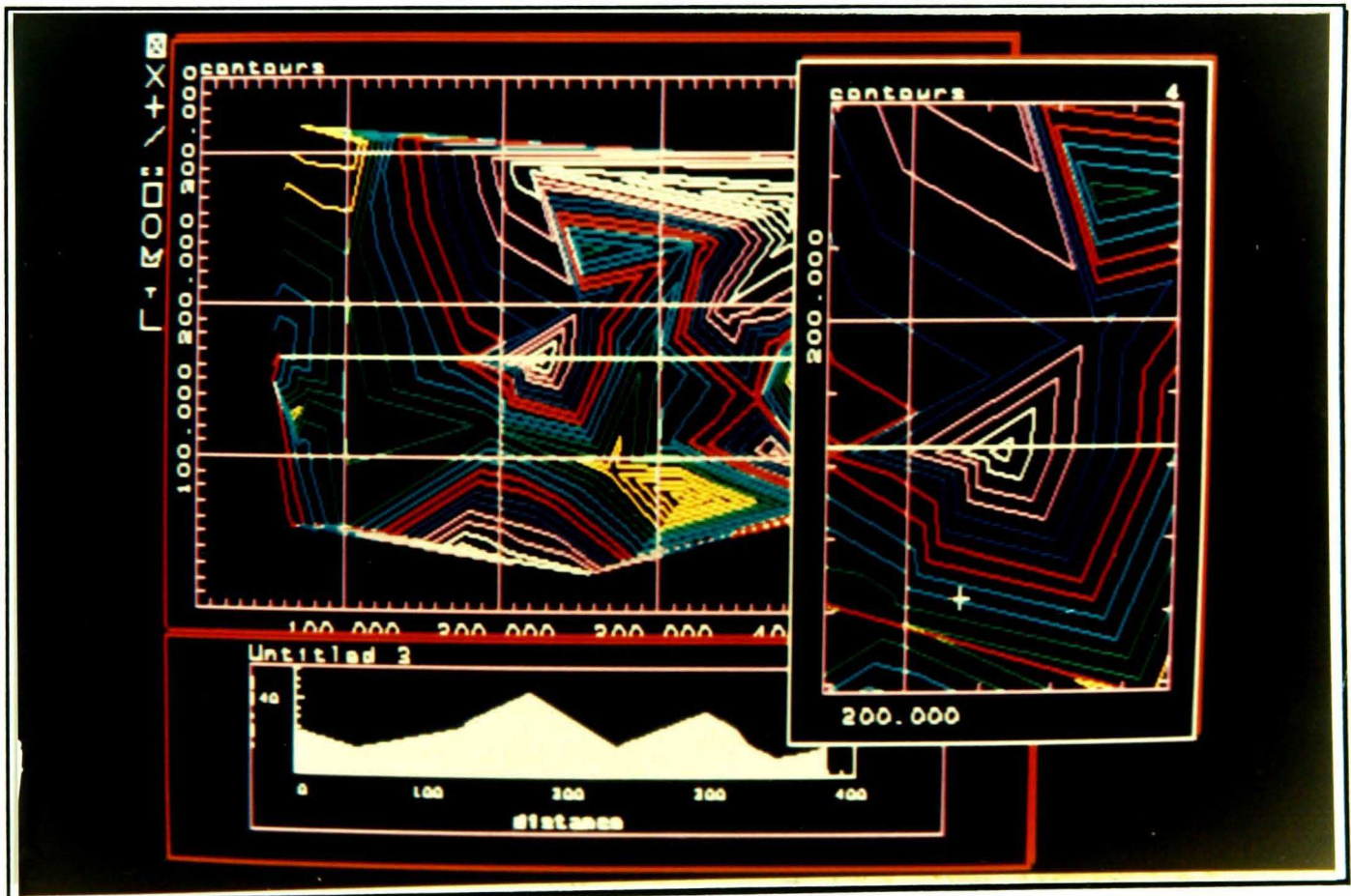


Plate 4.2 Maximising Screen Use using Multiple Graphics Viewing Areas

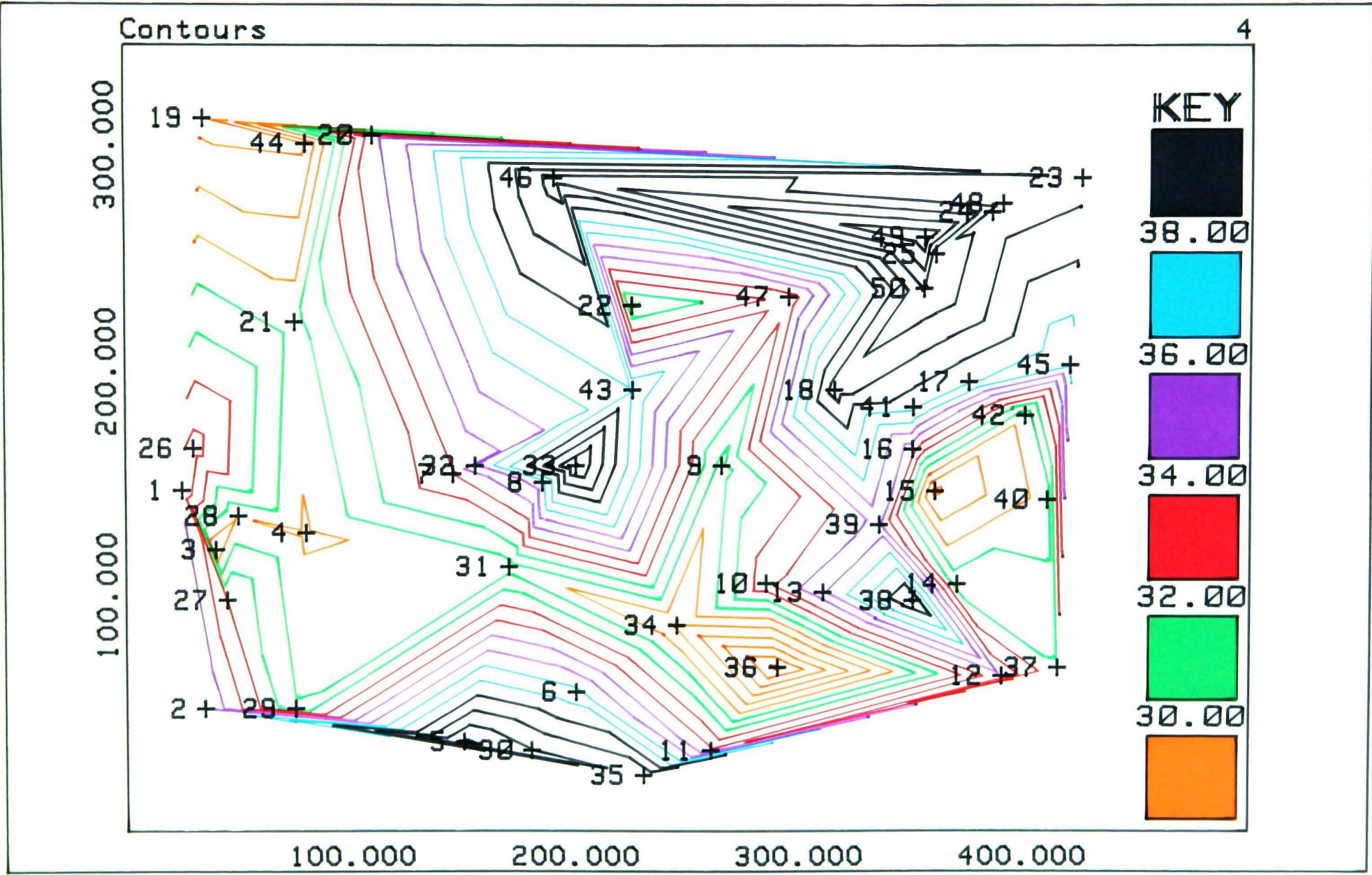


Figure 4.5 Boreholes with Contours Overlaid

4.3.2 Pictures

In order to optimise the usefulness of the graphics monitor as a means of displaying information, it was considered important that the user be given the ability to define a number of viewing areas on the screen simultaneously. In this way, the different graphical outputs associated with various aspects of the same problem could, if necessary, be viewed without having to switch between them and hence logical deduction by comparison would be improved. In Plate 4.2, for example, an exaggerated cross-section has been plotted adjacent to a contour plot showing the section plane.

The majority of mining graphics applications require an isotropic two dimensional plotting surface. For this reason it was decided to make isotropic coordinate systems standard for each viewing area. Achieving this proved more difficult than expected because of the display devices used.

Figure 4.6 illustrates how the programmer would go about defining the parameters of a viewing area using the D.G.L. package. The `set_display_lims` and `set_aspect` routines define the limits of the plotting surface while `set_viewport` and `set_window` are usually combined to produce the required coordinate system. An isotropic window (one in which the X axis scale is exactly equal to the Y axis scale) can normally be established by matching the height to width ratios of the viewport limits and the desired window coordinates respectively. Unfortunately, this method did not appear to produce a regular grid on either of the display devices used since the virtual coordinate system on both was anisotropic.

The obvious way to overcome the underlying distortion effects would have been to apply a compensatory factor to the aspect ratio. However, setting this ratio equal to anything other than that of the display width over its height reduces the total viewing area as seen in Figure 4.6, a rigid system was therefore developed which was able to produce isotropic coordinates on any type of C.R.T. or plotting device. The following algorithm shows how a user-defined display area can be set up for viewing isotropic information on the Microvitec HL colour monitor.

Objective :

To produce a general routine for defining a view area based on its display position in mm, so that a picture drawn on one device has an isotropic coordinate system which is identical in size to one with the same dimensions plotted on any other device.

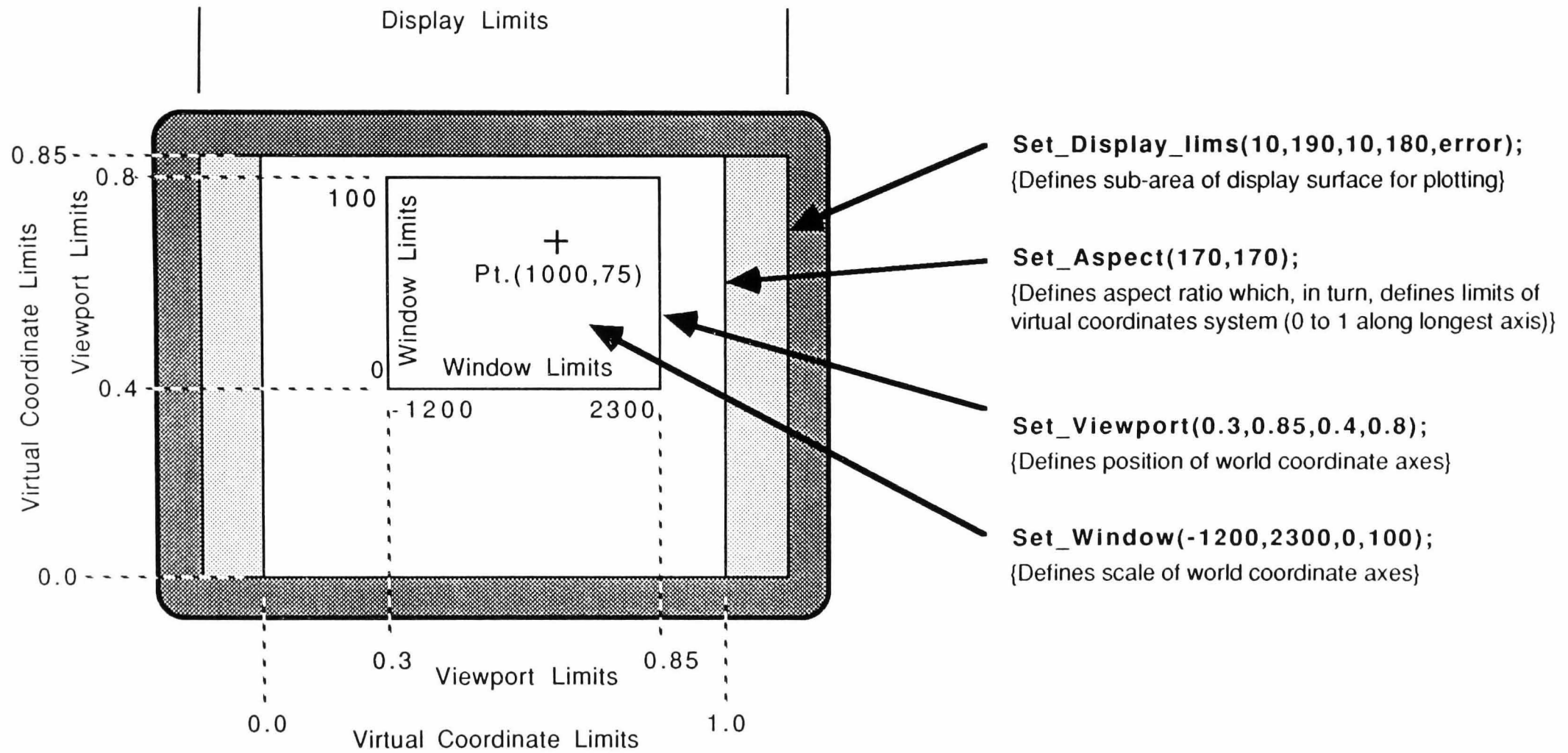


Figure 4.6 Hewlett Packard Viewing Area Definition

Algorithm :

1. Set the display limits so that the full screen area is utilised.
2. Set the aspect ratio to fit the display limits.
3. Set the window so as to produce a mm coordinate system.
4. Convert the limits of the desired viewing area (mm) into virtual coordinates, set the viewport accordingly and draw a box around its perimeter. (Fig 4.7 a)
5. Reduce the viewport by an equal amount in each direction to allow for labelling then fit an isotropic window with the required limits so that the maximum viewport area is utilised. Fig (4.7b)
6. Convert the new window limits into screen mm and reset the display limits to the values obtained so as to provide automatic clipping of any graphical information outside the current window.
7. Finally, reset the required window limits.

The only occasion where an anisotropic viewing surface may be required is when a graph is plotted. To avoid having to redefine the display area when plotting datasets, the transformation from the coordinate system of the viewing area to that of the axes is handled within NUmine rather than in the equivalent D.G.L. routines.

4.3.3 Picture Manipulation

The advantages of applying multiple windows to a user-interface have been explained (Section 3.3.7). The singular benefit of increased flexibility makes their role in the field of interactive graphics of equal importance. The potential utilisation of multiple viewing areas dictated the need for a collection of manipulative routines to control them.

When NUgraph is initialised, a blank, untitled picture is created. Thereafter, new areas can be defined anywhere, on any display currently linked to the computer, for viewing any picture, even one which is already visible through another viewport. It was made possible to bring a viewing area to the front of the stack, to move it to the back, to resize it, relocate or copy it either to another position on the screen or to another display device such as the HP-7475A Plotter. To simplify the configuration of a viewing area, a number of default paper sizes were established. Where possible, these were made constant for each device. In this way, an indication of the appearance of a picture which is to be transferred onto paper can be obtained from its screen image.

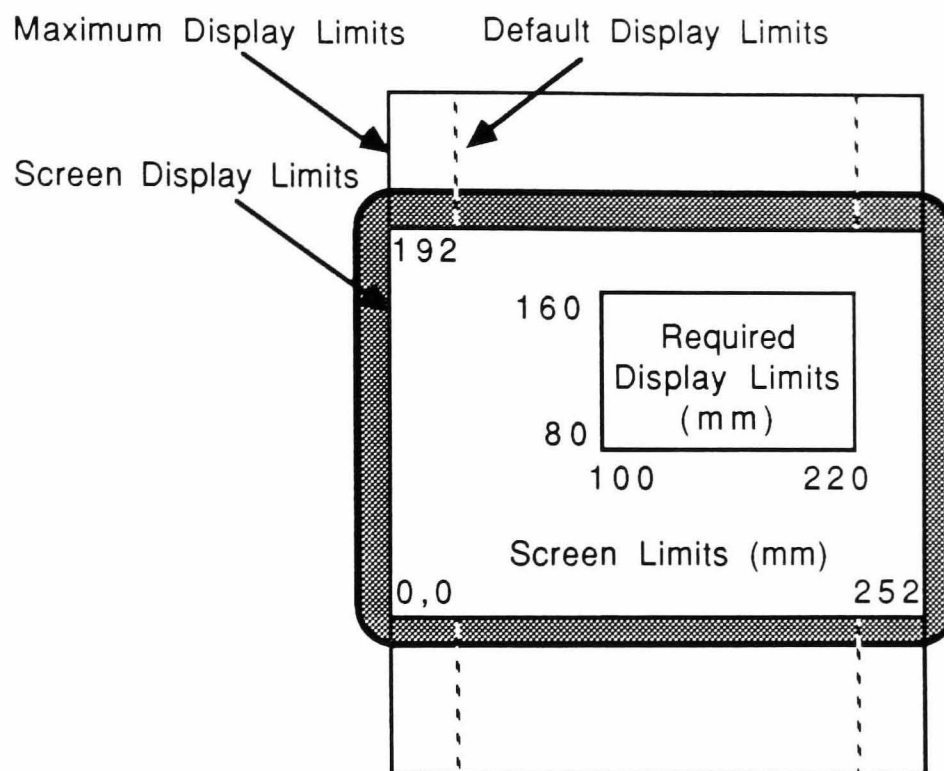


Figure 4.7a NUmne Display Area Definition

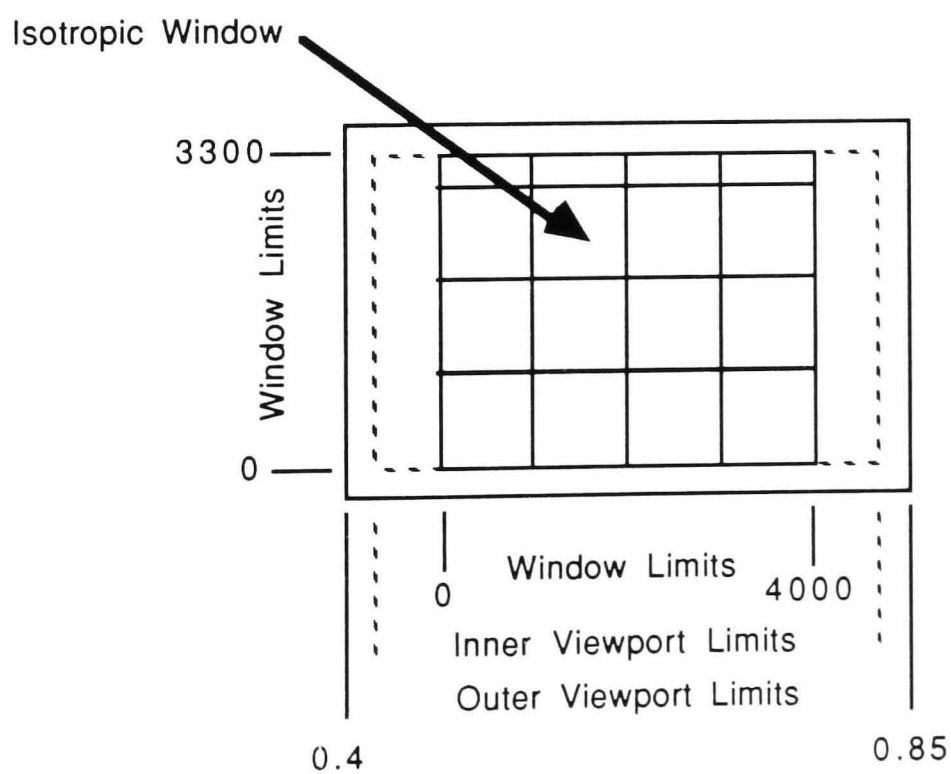


Figure 4.7b Isotropic Window Definition

In addition to the manipulation of a viewing area, the characteristics of its contents can also be altered:

- ❑ different types of grid can be overlaid on a picture;
- ❑ its border can be scaled or renumbered for accurate point location (Section 4.4.4);
- ❑ its limits redefined to suit a particular application or geographical location.

The picture's viewing angle can also be altered. That is to say it can be viewed in either the X,Y (plan view), X,Z (front elevation) or Y,Z (side elevation) cartesian planes. It can even be viewed in 3 dimensions as an isometric representation. A single command will split the current screen into 4 equal sectors showing the same picture in the 3 third angle projections together with an isometric view.

Finally, transformations can be applied which enable the user to home in on a particular area and view it in greater detail. The ability to display the same picture in more than one viewport means that the whole picture and a blown up portion of it can be displayed simultaneously. This is particularly advantageous where the interactive editing of detailed graphical information requires a degree of cross reference with the overall diagram (Plate 4.2).

4.4 The Graphics Package

The factors which differentiate a computer aided design graphics package from a simple graphics library are: The ability to access graphical information once it has been plotted by storing it in a sequential storage facility (segmentation); the ability to combine items together to form solid or composite objects (modelling); the ability to select and modify items by moving around on the screen (interactivity).

In the context of NUmine, these provide many advantages over conventional 'painting' techniques which can be used to display information but do not allow it to be later interrogated. For example, purely graphical information can often be stored as a segmented picture more efficiently than as a bit-mapped image (depending, of course on how much information the picture contains), while modelling and interactivity together provide an ideal platform for analysing geological information (Croghan, 1989) and designing networks for production appraisal. These and other advantages will be expanded upon throughout the rest of this chapter.

4.4.1 Ground Rules

There are 5 ground rules in the development of a graphics package (Newman and Sproull, 1981):-

Simplicity

Ease of use is important - generally a difficult package to explain is also a difficult one to use.

Consistency

The same techniques should be applied to similar operations.

Completeness

The package doesn't have to be exhaustive but if it is made possible to perform a certain operation on one item, it should be possible to perform the same operation on any other.

Robustness

The user should be incapable of causing an irrecoverable error.

Performance

Response time and display quality are the two important factors here. In a good graphics package these should depend primarily on the device not the programming.

During the development of NUgraph, high priorities were placed on all of the above so that the optimum number of NUMine applications can utilise the package to its full potential.

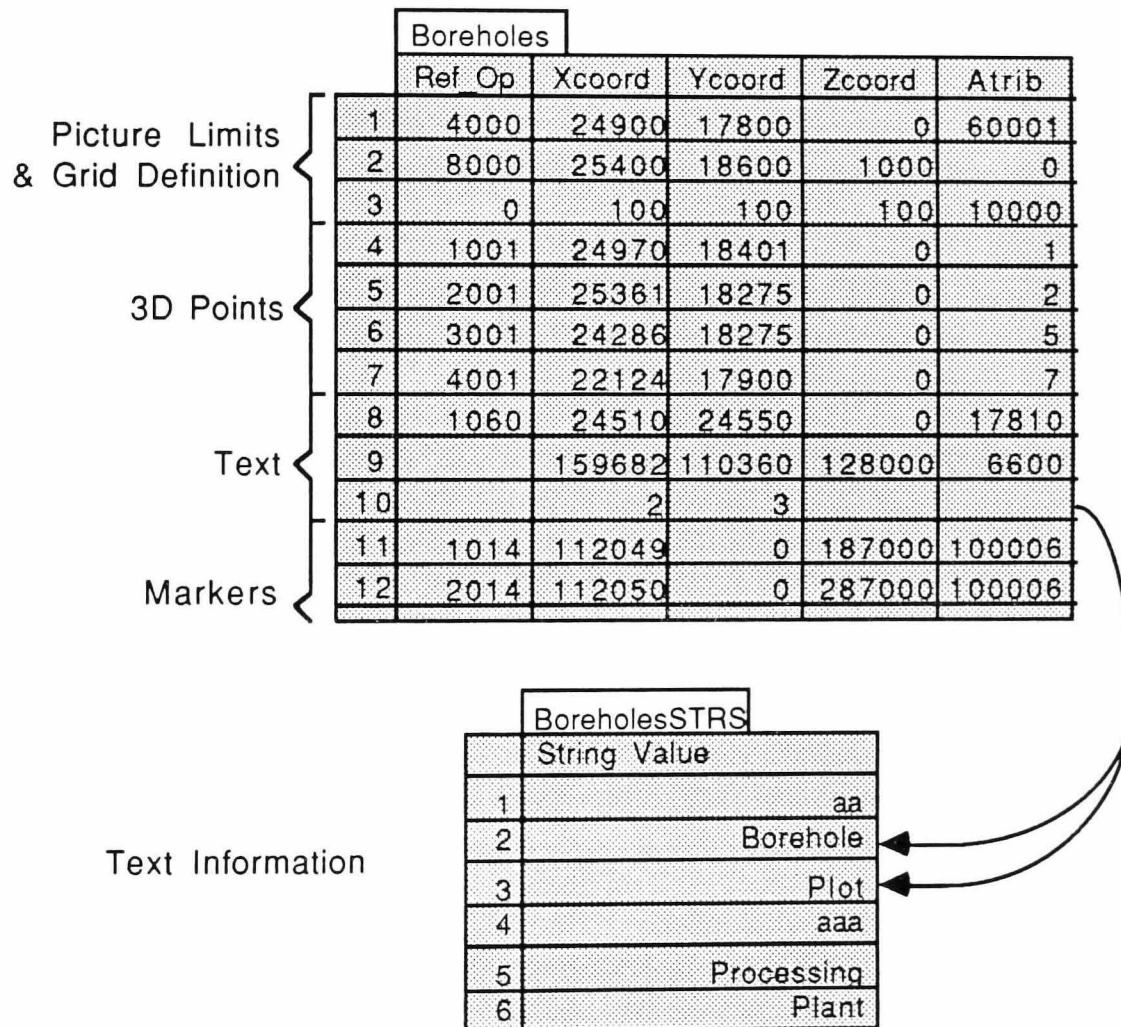


Figure 4.8 Graphics Relations

4.4.2 Segmentation

In order to make selective modifications to a picture, its individual pictures must be maintained in a dynamic storage facility such as a file. The NUmine philosophy of making each data item available to the rest of the system, dictated the use of the central database for picture storage which provided the additional advantage of being able to view and, if necessary, edit the picture in its relational format. However, even though many of the attributes of different picture items are similar, the limitations of speed and memory made it impossible to store the large variety of components which make up a picture in a purely relational manner. Consequently, a structure was developed which compacted each picture down into two relations making full use of each available field, often to store more than one piece of information (Fig. 4.8).

The main relation consists of 5 integer fields (ref_op, xcoord, ycoord, zcoord, and atrib). The ref_op field stores the picture item's identifier and its operator - that is an integer code indicating what type of object it represents. The other 4 choices of field name have little significance except when the item they refer to is a point in space. The only redundant field elements exist when a list of components form part of the item's description. For example, a polygon with an odd number of point components, which stores its X and Y coordinate values alternately in the Xcoord/Ycoord and Zcoord/atrib fields does not utilise its last two elements. All the picture's text information is stored in a second relation - individual strings being accessed from the main relation by their position in the second.

The main limitation of the use of the NUmine database at the time the graphics package was developed was the speed at which information could be read and converted into a form recognisable to the graphics library. In addition, it was considered unwise to attempt the considerable amount of data manipulation required on relations directly for fear of corrupting their contents or structure. A temporary storage facility was needed which could provide for the rapid sequential display and manipulation of picture items.

The method chosen uses pointers - a PASCAL language type which allows the user to access variables by pointing to their position in memory rather than referencing them by a specified name (Schneider *et al*, 1982). Rather than allocating a fixed amount of 'static' storage space to variables as they are defined, pointers seize and release space dynamically, giving the programmer a greater degree of control over how the available memory is utilised.

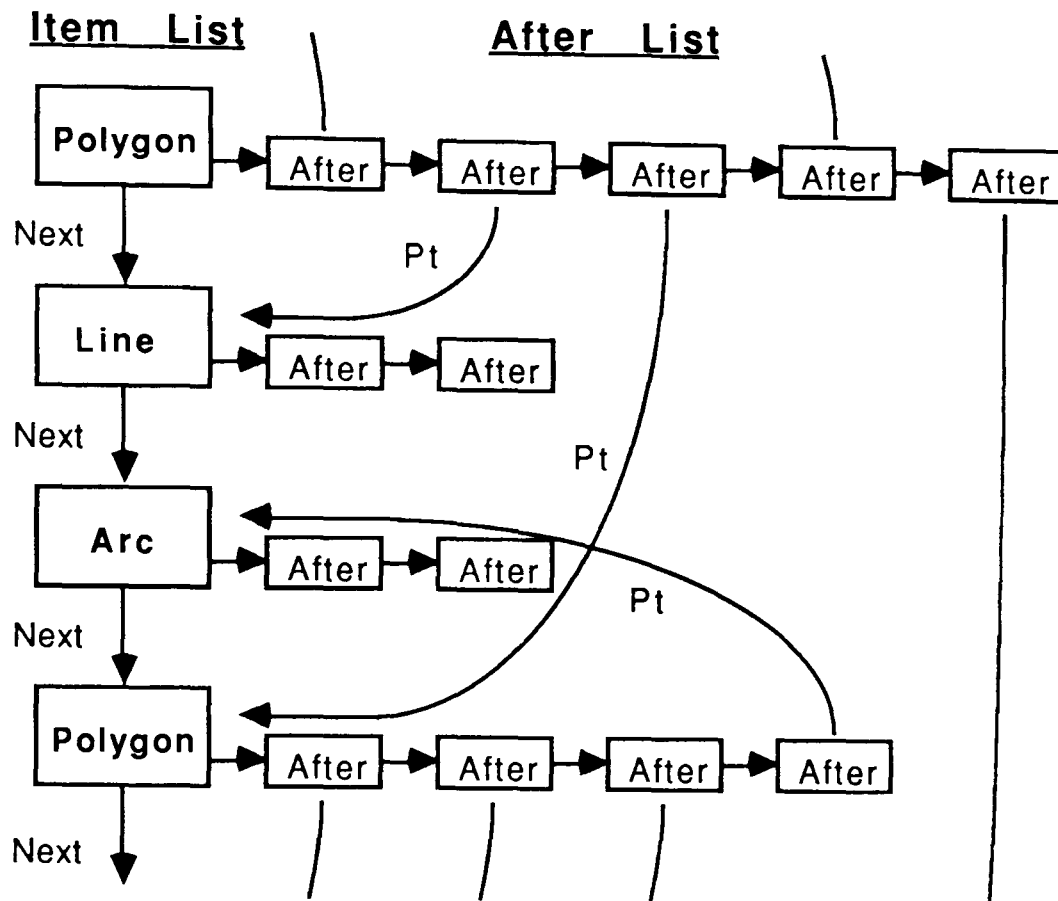


Figure 4.9 Graphics Primitives - Pointer Structures

As an illustration, the D.G.L. polygon plotting routines require arrays which define the coordinates and operators for each vertex. Had the same arrays been used to store this information outside the database, their size would have required definition at the programming stage which would have placed an immediate restriction on the number of datapoints per item. The choice of array size could only have been estimated as a trade-off between one which would have been able to cope with the largest envisaged number of data-points and one which, when multiplied by the potentially large number of multi-point items (polygons or datasets) which could exist at one time, would not have caused memory problems. Whatever the choice, a large amount of storage space would have been taken up by redundant array entries and there would always have been the possibility of array overflow.

These problems were eradicated by storing items in a linked agglomeration of pointers to records and allocating the required amount of memory to each according to its current size. Figure 4.9 illustrates how items which take up more than one record gain access to further storage space through a secondary list of 'after' pointers so that the original sequence from one item to the 'next' is preserved.

An item's pointer structure is similar in many ways to its relation format, the main differences being that the reference and operator values, most often required for searching purposes, are separated and that strings are accessed by a direct link from the first record. This simplifies the storage and retrieval process but manipulation of the pointer format itself is more complex than using a direct interface with a database relation. For this reason, a number of procedures were developed for automatically adding graphical items to a picture or editing those which already exist (Fig. 4.10). An equivalent number of reading routines were included (one for each data type) which return all the characteristics of a particular item in an individual, decoded form. When an item is removed from a picture, its properties are no longer required, hence the deletion process for each item type can be handled by a single routine.

4.4.3 Modelling

Modelling, in the graphical sense, is the ability to combine items together to form composite structures. A picture can, of course, be thought of as a model in itself and is frequently used as such in the NUmine system. However, it is the interconnection of objects on a lower level which provide the greatest benefit in terms of storage space and performance - particularly when dealing with 3-dimensional or 'solid' models. For example, in its relation format, a line in 3 dimensional space would require 6 field elements in which to store the coordinates of its end points alone. However, if the end points are stored separately and referenced from the line by their respective identifiers, two fields are required at most (one if the integer references are combined on a single field). Though the aggregate number of fields is greater at this stage, as soon as one end point becomes accessed by a second item, the overall storage space will have been reduced.

These points show up clearly in NUmine's three dimensional solid modeller where surfaces are defined by triangles consisting of 3 vertices connected by 3 lines and solids as polyhedra which are combinations of the surfaces. Figure 4.11, which does not go as far as considering the definition of the polyhedra, shows that, even when the points themselves are superfluous to the problem (the reason why they are not included in the 'unlinked' model), a considerable saving can be made.

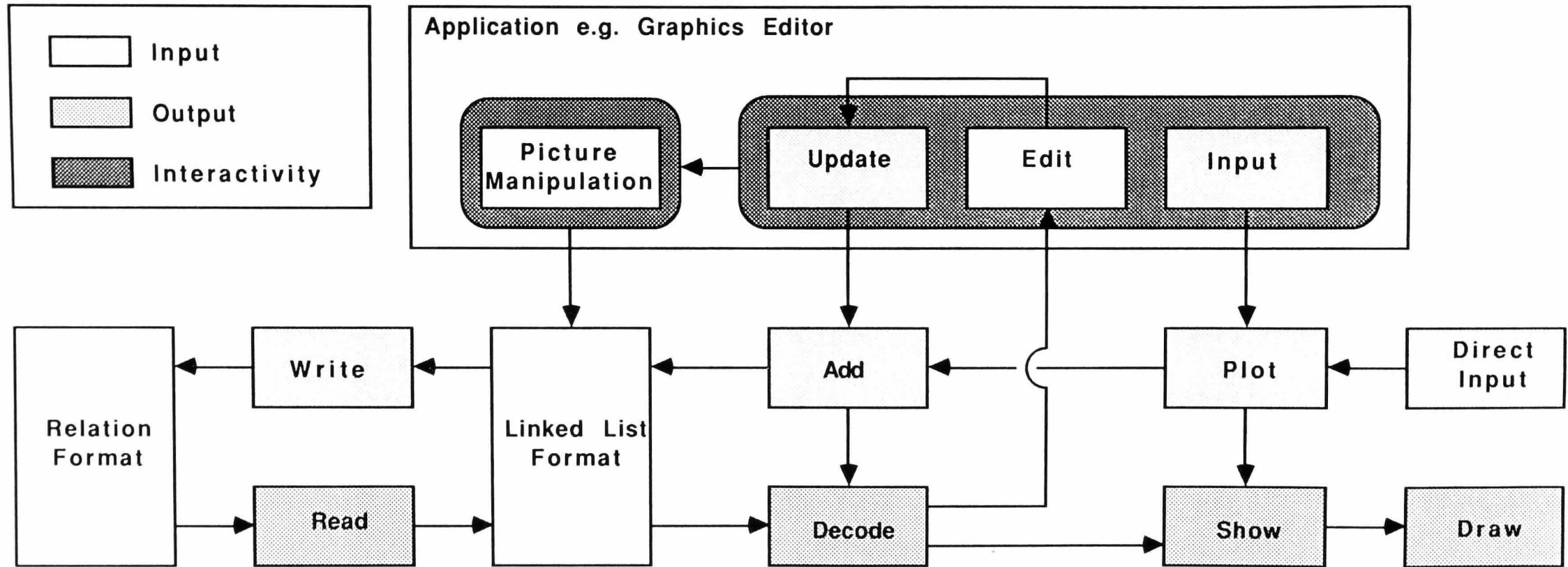


Figure 4.10

Adding, Editing and Decoding Graphical Information

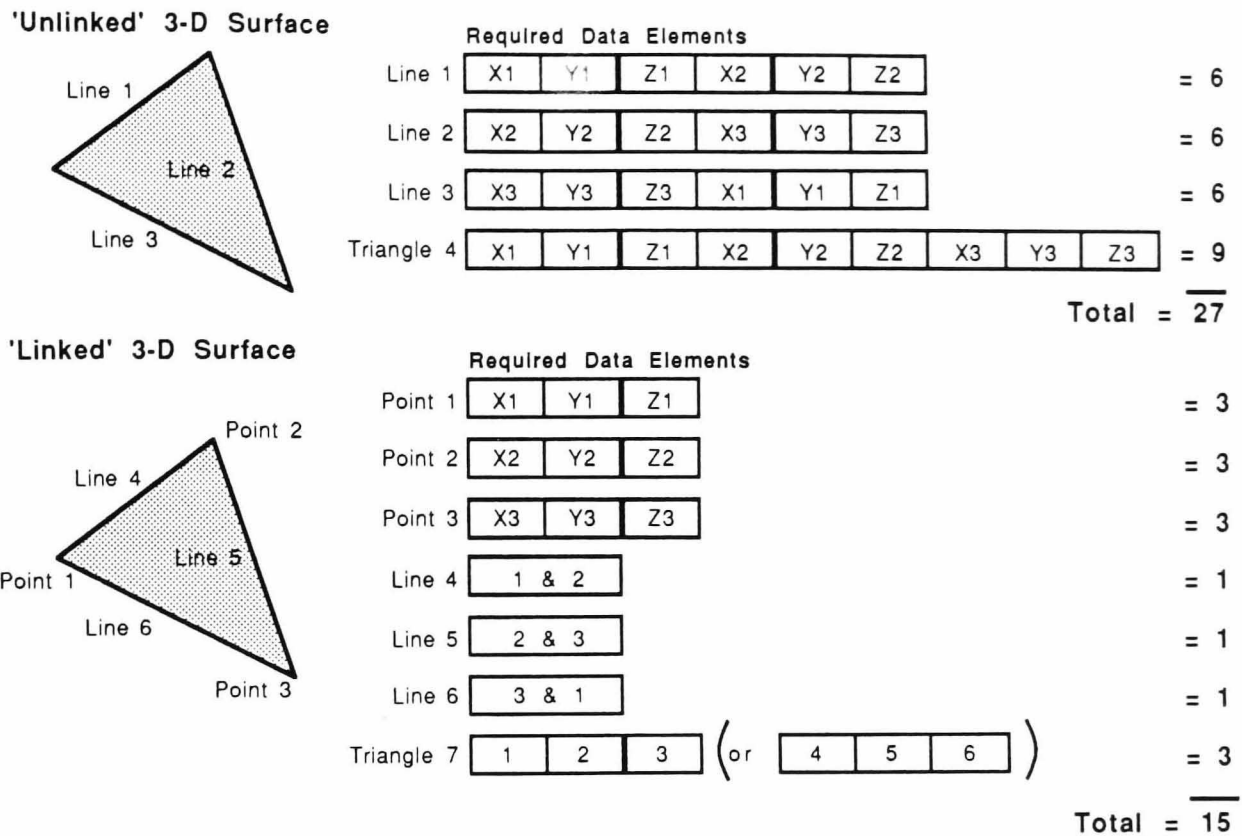


Figure 4.11 Graphical Modelling of 3-dimensional Surfaces

It is not only in 3.D.- C.A.D. representations that the benefits of modelling can be found, in the construction of networks for project scheduling or simulation a number of interconnected nodes are defined which have no dimensional characteristics other than their positions on the screen. This often has no direct bearing on the application for which the network was created so by being able to access points by reference only, irrelevant positional information is ignored and conversion to the required format simplified (see Chapter 6).

A picture in its pointer format actually consists of three inter-linked lists (Fig. 4.12), one to store item characteristics, one to store 3D-points and the other to store the characteristics of any secondary pictures which are to be displayed on the first as a background or overlay (these are temporary links which are lost when the picture is saved as a relation). When the modelling/network facility is activated, multi-point items link to their various components through a list of secondary pointers as described in the previous section. In the case of polygons, the connection may be made to the lines which make up its perimeter rather than its vertices - an important feature when defining haul-routes and an economic one when two shaded areas share a mutual edge.

Viewing Areas on the Screen

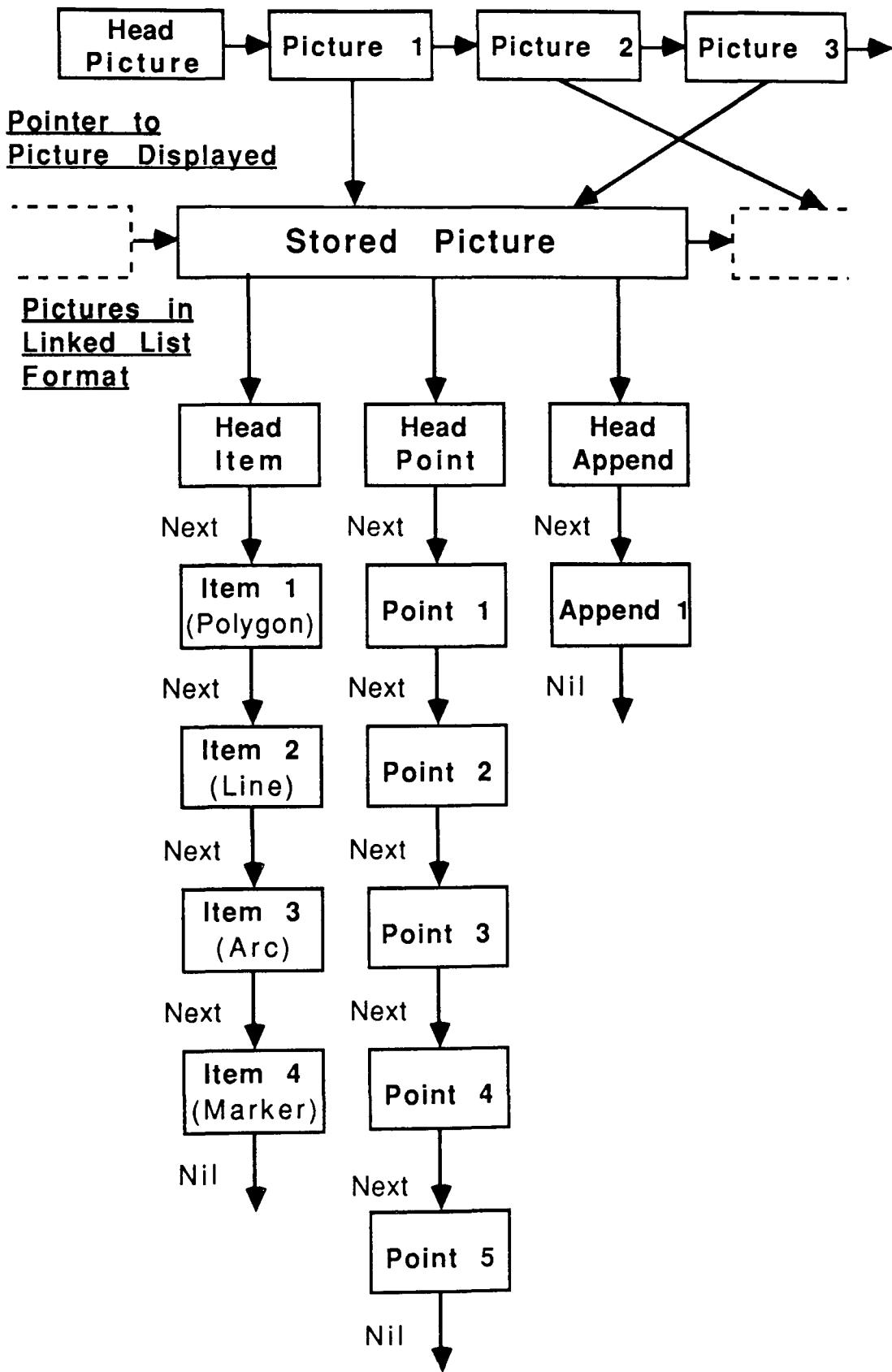


Figure 4.12

A Picture's Pointer Structure

4.4.4 Interactivity

The way in which NUgraph interprets the activation of the system's input devices (Section 3.2.1) is an extension to the event-handling example illustrated in Figure 3.4. The variety of responses to the user's actions play an important part in improving their effectiveness and allow him to achieve specific tasks in the shortest possible space of time. One of the most important of these is the use of feedback which, by providing an immediate graphical response to each input action, reduces the user's uncertainty about its consequences.

The most basic example of feedback is the graphical cursor which moves around the screen in accordance with mouse movement, activation of scrolling keys or repositioning of the digitising stylus. The NUmine system uses a purpose-built location mechanism which is called every time a positional reference is required. If necessary, the user can specify coordinate values exactly by typing them directly into a locator window which is displayed on the alpha-screen (Fig. 4.13a). As an alternative, the trace option continuously monitors the cursor position and updates these values automatically (Fig 4.13b). When the cursor has been positioned, a single keystroke notifies the computer that the desired location has been reached and what action is to be taken. For example, under normal circumstances, a '2' indicates that a marker is to be added.

Where more than one 'location' is to be made, such as in the definition of a line, feedback is applied in the form of a rubber band joining the position of the first end-point indicated to that of the cursor (Fig. 4.13b, d & f). The accurate repositioning or rotation of existing graphical items such as arcs or polygons is simplified by echoing its outline as the cursor moves. However, the additional plotting time involved demands that this only be carried out once the cursor has remained stationary for a specified period. For multiple point definition, the position of the cursor relative to the previous location is also displayed. These can be fixed, at any time, to place a directional constraint on the object currently being drawn (Fig. 4.13 c-f). Movement of the cursor across a particular view area can be made incremental by forcing it to lie on the nearest intersection of a user-defined grid (Fig. 4.14a). This, naturally, places a modular constraint on the coordinate position returned.

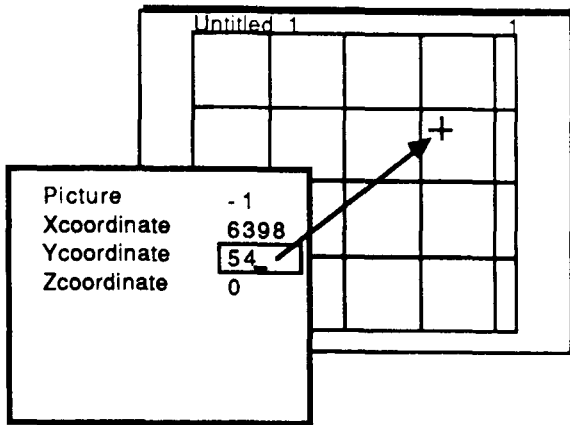


Figure 4.13a Typed Input

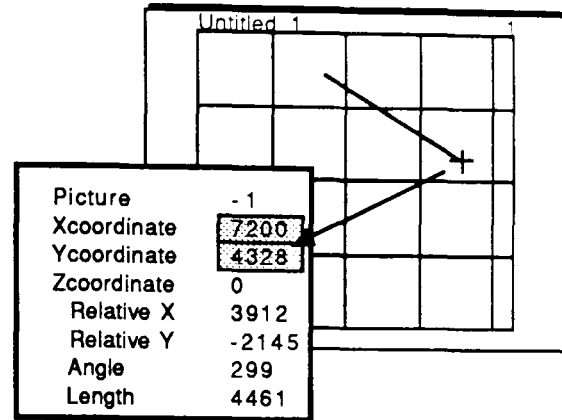


Figure 4.13b Tracing the Position of a Line's End Point

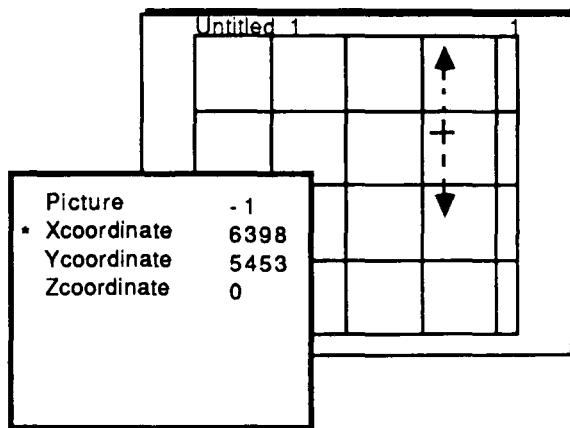


Figure 4.13c X-coordinate Constraint

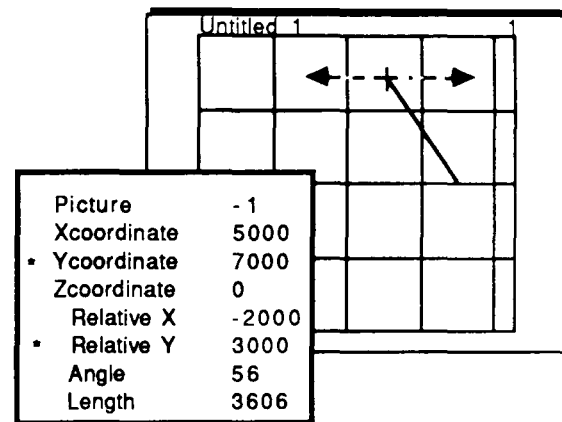


Figure 4.13d Y-coordinate Constraint

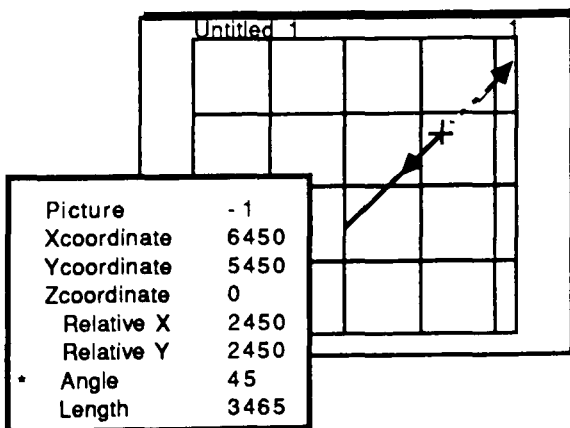


Figure 4.13e Angular Constraint

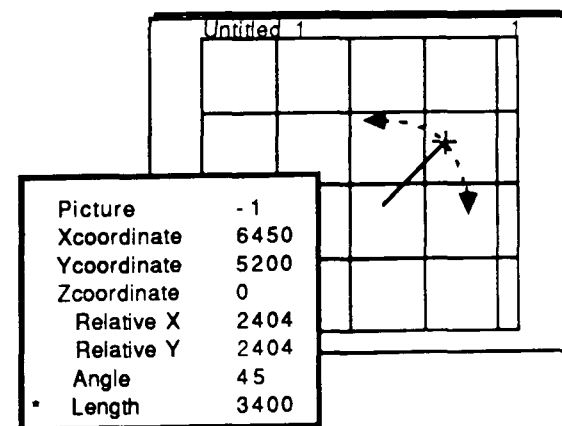


Figure 4.13f Constrained Distance from Previous Point

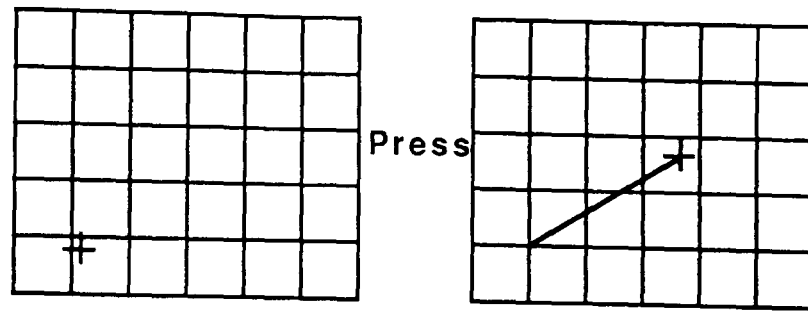


Figure 4.14a Modular Constraint - Snapping to a Grid
(after Newman & Sproull, 1981)

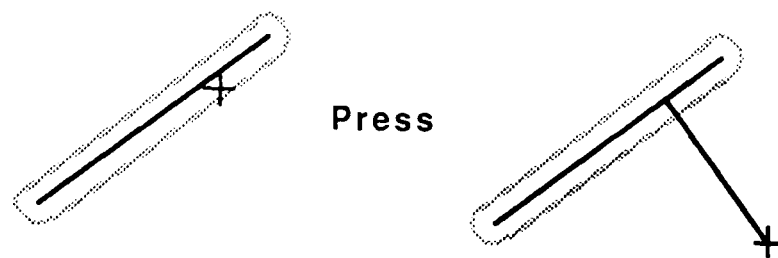


Figure 4.14b Tolerancing to Existing Graphical Items
(after Newman & Sproull, 1981)

A major requirement of the modelling process is the ability to attach two or more items together. This is very difficult to achieve on a screen of limited resolution especially when there are a large number of points which lie off the grid. This problem has been resolved by applying a gravity-field to each line and end-point in the picture (Fig. 14.b) whereby a 'location' made within a certain tolerance of the item, automatically snaps onto it (Newman & Sproull, 1981).

The importance of an efficient clipping algorithm in the reduction of plotting time was explained earlier (Section 4.3.1). However for complex items such as polygons and datasets, this is, in itself, a time consuming process which should, therefore, only be applied to those partially visible. To eliminate the clipping process from those multi-point items which lie either completely inside or completely outside the current view, a 'box' is established surrounding the item which can be interrogated prior to plotting in order to determine its coordinate limits. If the simple comparative statement :

$$\begin{array}{llll}
 (\min X_{\text{box}} < \max X_{\text{view}}) & \text{and} & (\max X_{\text{box}} > \min X_{\text{view}}) & \text{and} \\
 (\min Y_{\text{box}} < \max Y_{\text{view}}) & \text{and} & (\max Y_{\text{box}} > \min Y_{\text{view}}) &
 \end{array}$$

is true then the item is at least partially visible. A further check that

$$\begin{aligned} &(\min X_{\text{box}} > \min X_{\text{view}}) \quad \text{and} \quad (\max X_{\text{box}} < \max X_{\text{view}}) \quad \text{and} \\ &(\min Y_{\text{box}} > \min Y_{\text{view}}) \quad \text{and} \quad (\max Y_{\text{box}} < \max Y_{\text{view}}) \end{aligned}$$

indicates that the item is entirely visible unless impeded by an overlying object.

The box is also used in the interactive selection process. Here the user is given the ability to select an individual object for editing or interrogation by locating a position somewhere within its bounding rectangle (Fig. 4.15). In some cases, the shape and position of the box itself can be modified to apply transformations of scale or rotation to the object which it contains.

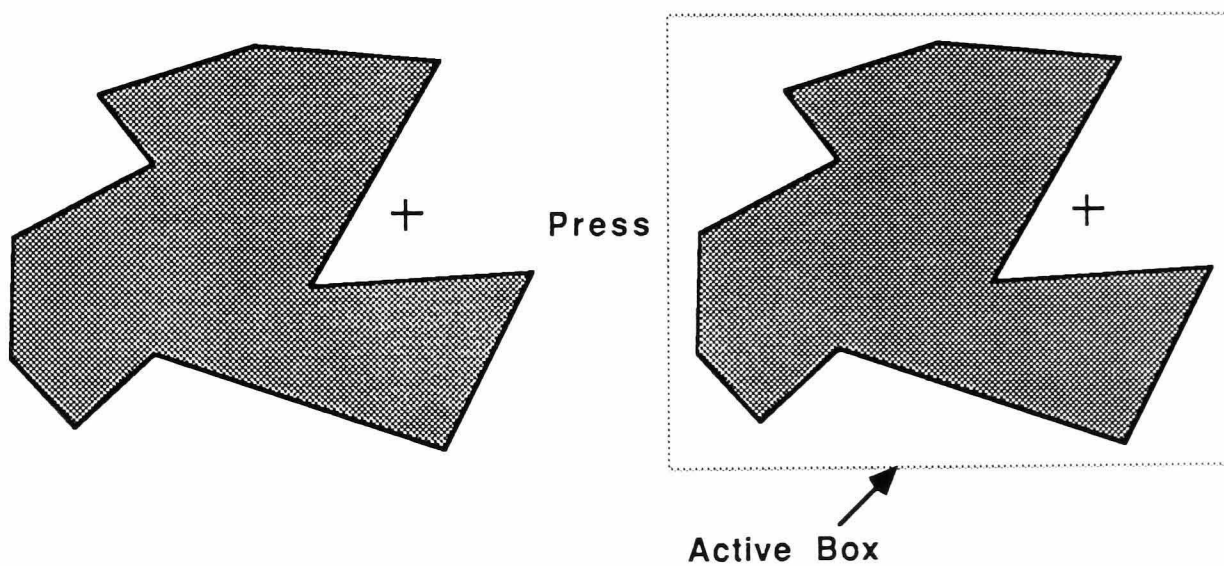


Figure 4.15 'Boxing'

4.5 Conclusions

This Chapter has been used to describe the NUmine graphics system (NUgraph). Its main purpose is to act as an effective means of viewing and manipulating data which either resides in the central relational database or is temporarily present in one of NUmine's satellite applications. It does so by displaying and combining graphics primitives in particular configurations in any number of user-defined viewing areas on the screen. The information can be interactively edited and either stored for future reference or used as a model for immediate interpretation (see Chapter 6).

My initial aim was to develop a graphics system to act as a vehicle to assist the planner in his task of building digital models which could then be used in the haulage analysis process. However, so many of its basic requirements were common to NUmine's other project areas that, in the interests of integration, the package was generalised and NUgraph emerged. Its role in the equipment selection and scheduling module will be expanded upon in Chapter 6.

Chapter 5

***NUsim - A Simulation Package for Computer Aided
Mine Design and Planning.***

Chapter 5

NUsim - A Simulation Package for Computer Aided Mine Design and Planning

5.1 Introduction

The principle of simulation languages, and the various languages which are available today have been discussed (Section 3.2). Their benefits are known (Brown *et al*, 1987), their disadvantages are less well documented but what is clear is that a computer-aided mine design system would not be complete if it didn't include some form of simulation package.

Two major conditions of the NUmine system, are that its component units are both user-friendly and completely integrated. They must, in other words, be fully operational modules which can be called up and used from either the master program or other working modules. Their general use cannot involve writing code. NUsim was developed to fulfil these conditions and, at the same time, act as, and retain most of the important features of a simulation language. It is actually based on the fundamental principles of SLAM (Pritsker & Pegden, 1979) though the use of PASCAL has allowed for a highly structured and recursive environment in which to work. A high priority is placed on graphical representation and on the interactive editing of network items. These improve the user's ability to transform a real problem into an analogous model and to interpret the model's results.

This chapter describes the NUsim module. It shows how standard PASCAL routines can be recognised by the system and allow it to be used as a simulation language. It shows how the various components of a NUsim network interact (Sections 5.3, 5.4 and 5.5) and how they would be applied in a simulation model (Section 5.6). It also describes the various processes handled by the module at different stages in a simulation project. Throughout, are several examples relating directly to the truck/shovel model developed in the following chapter.

5.2 NUsim Fundamentals

In order to use the NUsim module as a language, it was necessary to make available a full range of predefined routines. These could then be combined to produce equations and functions relevant at various times in the modelling process. Since all program generation was carried out in a PASCAL environment, identical terminology and syntax structures were adopted throughout NUsim. However, the routines themselves could not be used directly. Trying to recognise source code from within a running program is like trying to decode a scrambled message - rules are required which tell the decoder what to do when it sees a particular word or symbol.

The first objective was to make all the standard PASCAL features available i.e. recognition of real and integer numbers, standard symbols such as +, -, :=, <=, >=, etc. and parenthesis. Then, to incorporate basic functions (round, mod, sqr, cos, and, not, etc.) as well as real named variables. (The basic functions all made indirect use of their standard PASCAL equivalents). Finally, some simulation-specific routines were added including a set of random number generating functions. A full list of NUsim functions is given in Appendix I)

5.2.1 Compilation and Calculation

A large proportion of actual simulation time is spent decoding strings of characters. As indicated above, these may have been made to represent constant real values, variables, random variables or any function thereof. For simplicity, these will henceforth be referred to collectively as NUsim functions. To speed up the process and allow a larger quantity of information to be handled simultaneously, NUsim functions must be compiled before any calculations are allowed to take place. In so doing, all symbols, variable names and function headings are replaced by single byte characters. Only numbers and single-character symbols retain their full length.

This process of recognising and converting NUsim function strings is carried out in a similar way to the calculation of the function itself. Values in brackets (including function parameters and array indices) must be given priority over those outside. Since it is possible to nest functions inside others and even refer to them by name only (see The 'Variables' Relation), the problem is a recursive one (Fig. 5.1).

A significant amount of compilation is carried out prior to the simulation process starting. However, the calculated value of a NUsim function will usually depend on the current status of its variable components. Calculation, therefore, is always carried out at the time the function is called.

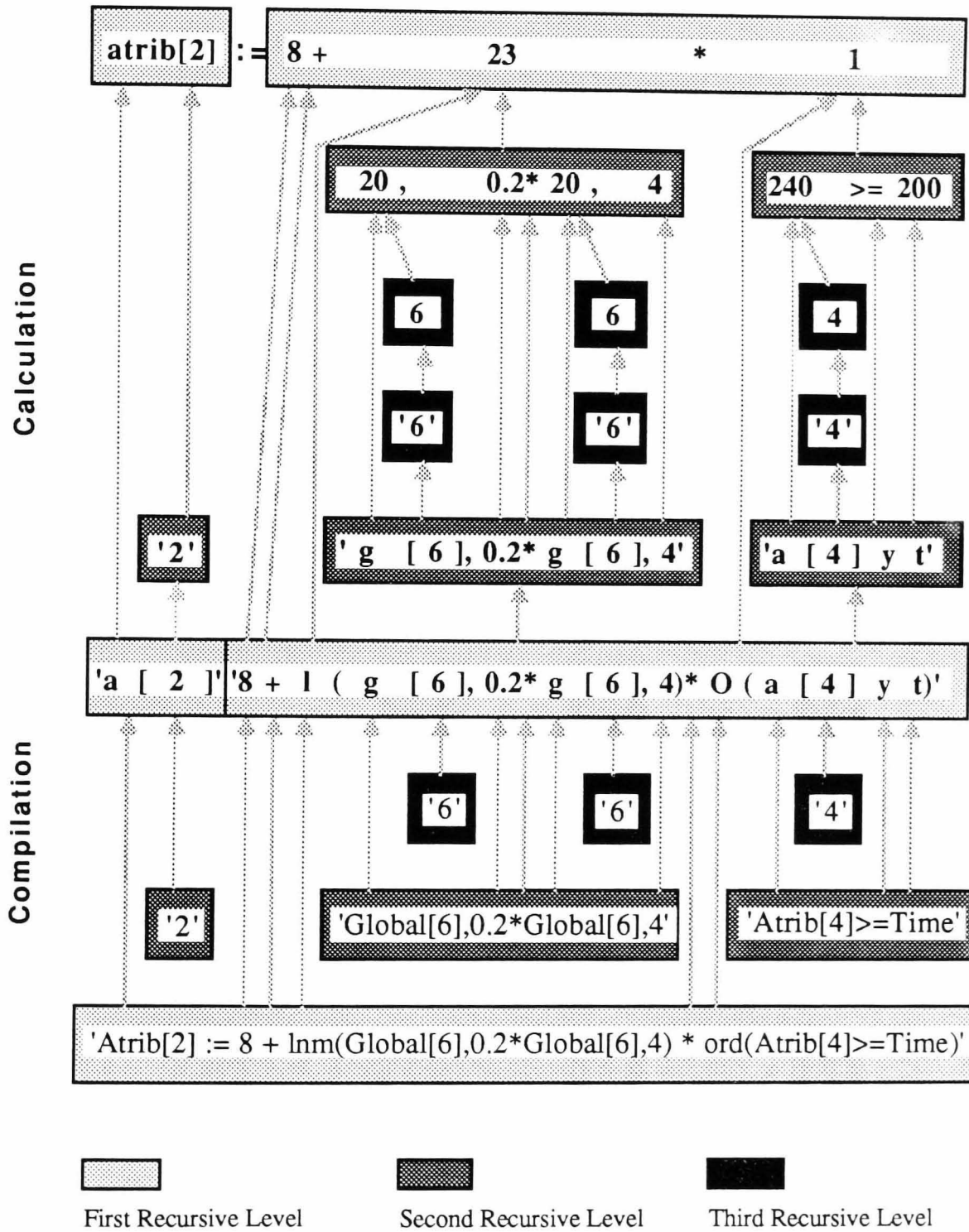


Figure 5.1 Compilation and Calculation of a NUsim Function

5.2.2 Random Functions

The essence of probabilistic simulation is the consistent use of random as opposed to constant real variables. These provide a means of modelling the uncertainty which is inherent in a stochastic process.

The effective application of random variables is wholly dependent on the availability of an efficient random number generator. A great deal of literature has been written on the subject of generator suitability to different applications but, for the purposes of NUsim, it was considered sufficient to make use of the one developed by Hewlett Packard (1985(b)). Using their in-built system, a random integer in the range 0 to 32766 inclusive may be generated from a single line of source code. The number is, in fact, pseudo-random in that it is derived algorithmically from a specified seed. The seed is itself pseudo-random - each time the function is called, a new value for the seed is generated and returned. In this way, different generating streams, stored in an array called the seedbed, may be applied to specific areas. If a process is modelled twice using the same set of initial seed values, identical results will follow.

Unfortunately, random processes are not always uniform in nature. If they were, numbers generated from the above function could be translated and scaled for any application. More often, an uneven distribution of outcomes is associated with the process and alternative techniques for obtaining typical values from it must be used. A common sampling technique is the Monte Carlo method. This applies a random real value in the interval 0 to 100 to a cumulative frequency curve, collated in percentage terms, of the process in question (Fig. 5.2). The result produced is more likely to lie within a range of high frequency outcomes (steep gradient) than elsewhere.

In NUsim, distributions of this type are created and retrieved as datasets stored in a NUgraph picture. However, the prohibitive speed of calculating values from cumulative distributions, makes the use of random number generating functions which are specific to the main distribution types, a necessary alternative. Here, 'known' processes which approximate to standard distributions can be sampled far more efficiently.

A total of 11 generating functions (including Monte Carlo) were added to the list of standard NUsim functions (Appendix I). Each employs the built-in random number generator at least once per calculation which ensures that the seed gets iterated as normal. Further information on the way these functions operate can be found in 'Statistical Distributions' (Hastings & Peacock, 1975), 'Introduction to Simulation and SLAM' (Pritsker & Pegden, 1979) and 'Numerical Recipes' (Press *et al*, 1986).

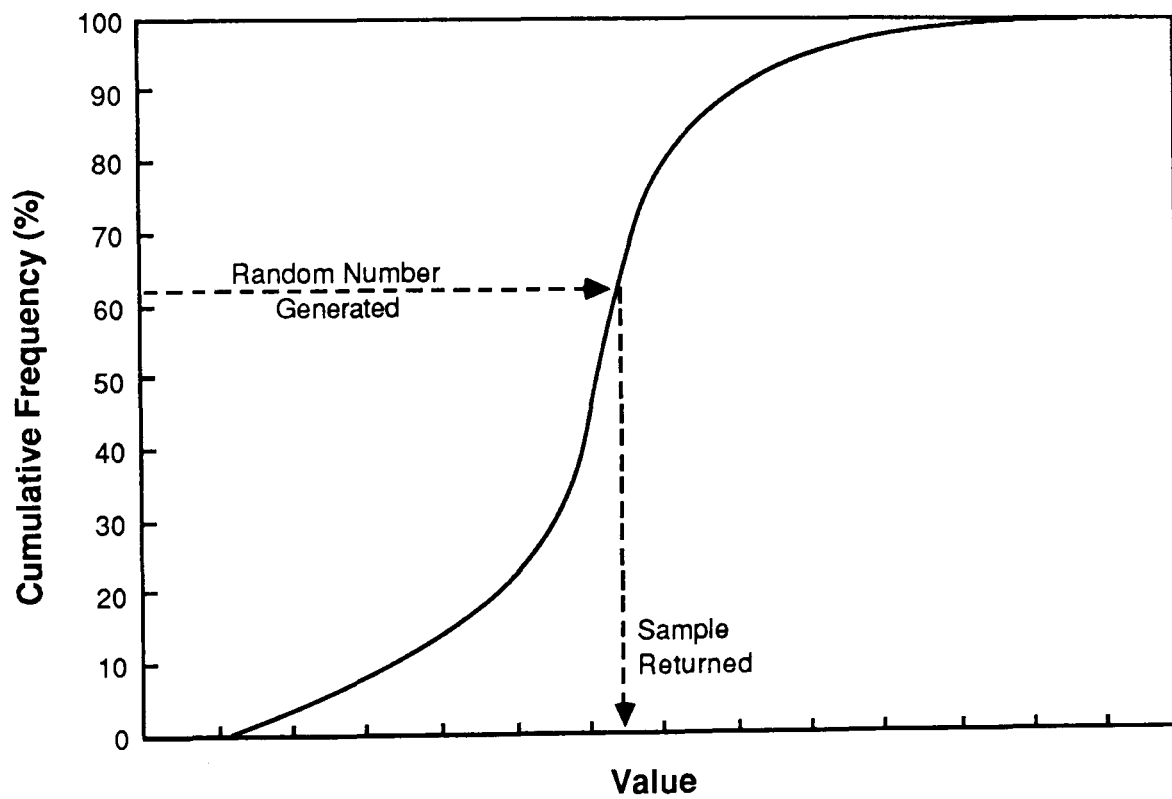


Figure 5.2 Sampling from a Cumulative Frequency Curve

5.3 Network Modelling

Any discrete process may be looked upon as a series of entities flowing through a network. Each entity consists of a set of attributes which can be assigned real values at any time during a simulation run. These enable the user to identify entities and model changes in their individual characteristics. The network is composed of a series of nodes and branches which together represent the different types of event which occur during the process being modelled, the links between them and the passage of time (Fig. 5.3).

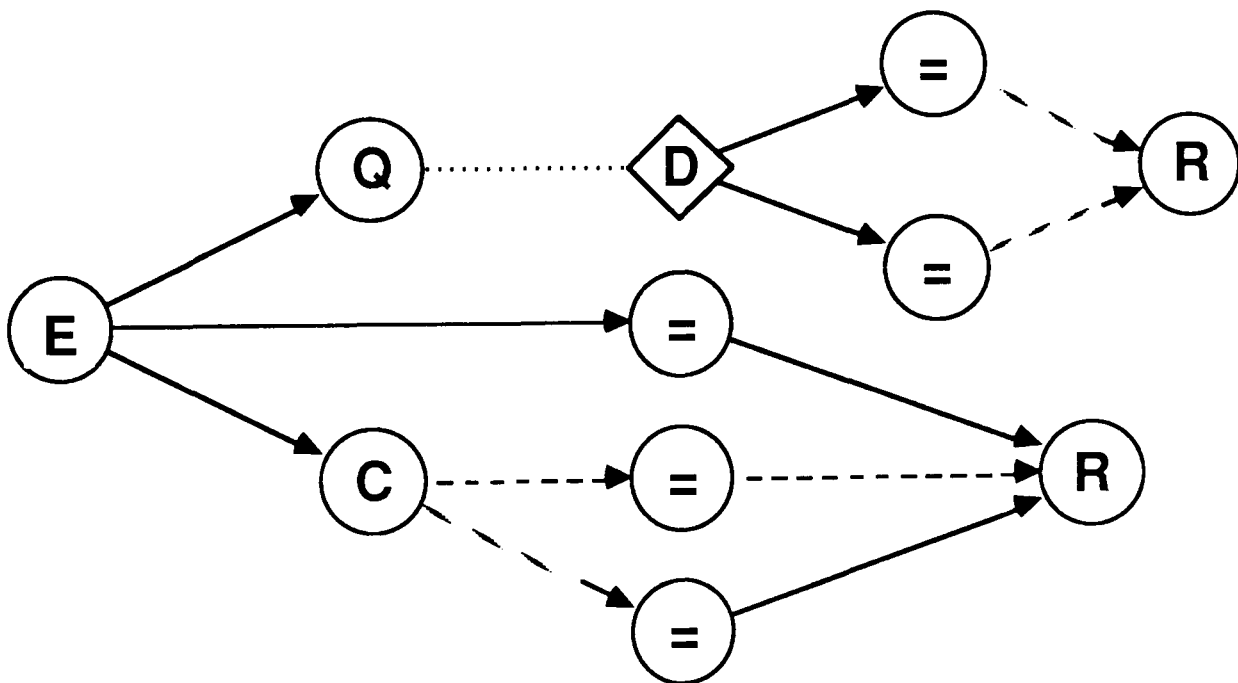


Figure 5.3 NUsim Network Example

The network can be created and edited graphically or manually. Interactive graphical editing allows for the rapid creation and comprehensive visualisation of each problem. Using the graphics editor, branches between nodes are represented as dotted lines. When editing the network manually, however, nodes can be linked together directly and the branches between them need not be defined (Fig. 5.4). To avoid confusion, throughout this description, a branch will always be assumed to exist between nodes.

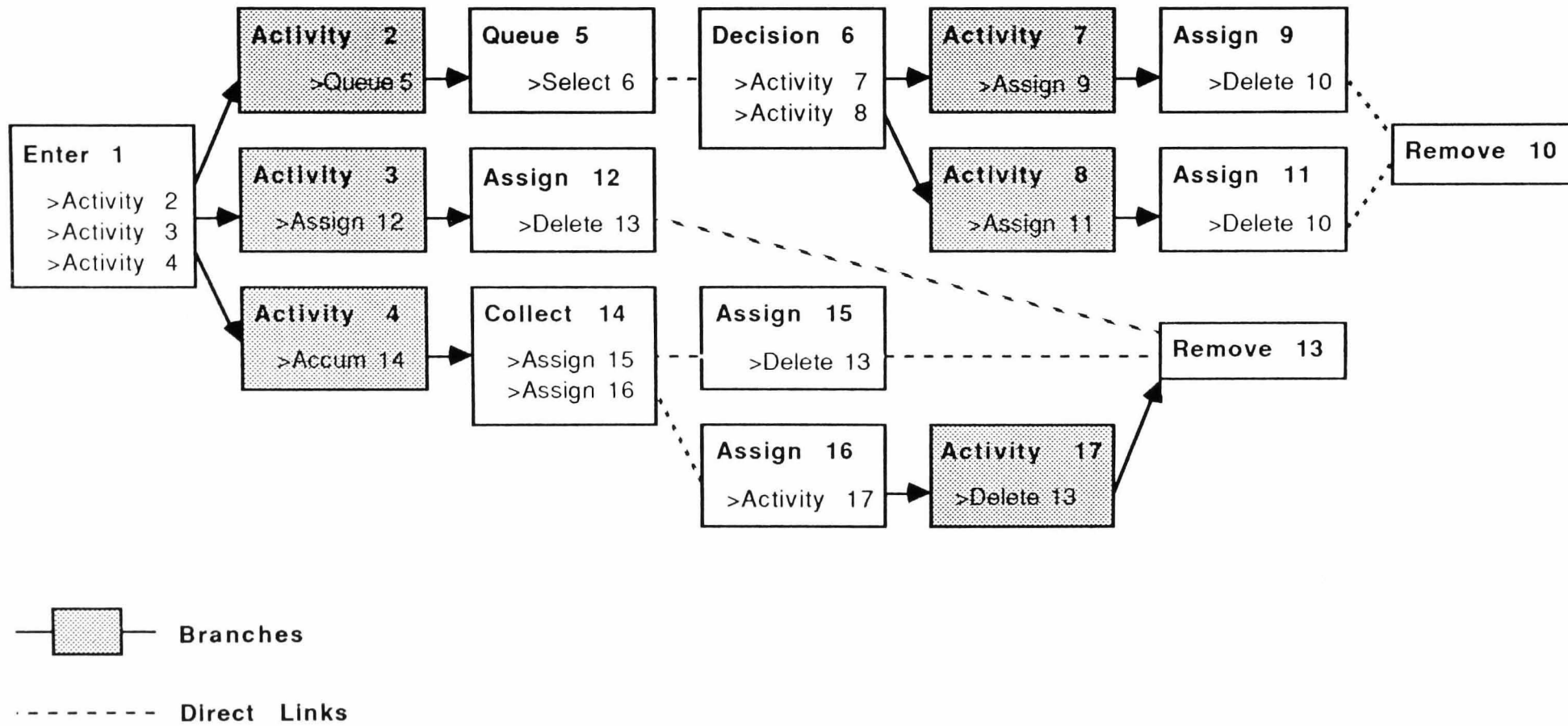


Figure 5.4 NUSim Network Example (Manually Defined)

5.3.1 Modelling the Passage of Time (Activities).



The passage of time is represented, in a NUsim network, as a solid line between two nodes called an activity. Each activity is, in fact, a branch for which a duration has been specified. Like a number of other field values, an activity duration may be specified as any NUsim function. An activity's other parameters are:

- probability or condition.

An activity can either be designated a probability, which is specified as a real value less than 1, or a condition which is expressed as a boolean equation. Because of their mutual exclusivity, the two values can be stored in the same field and are differentiated by the style of the declaration (see routing entities from nodes)

- the number of parallel servers (Section 5.3.7)

When an activity starts processing an entity, a value for the time taken to complete the activity is calculated and a new event is scheduled which, at the appropriate time in the future, will indicate that the activity has been completed. This new event is appended to an 'event calendar' which progressively stores all future events in the order which they are scheduled to occur. In this way the simulator repeatedly takes the next event from the calendar, advances the current time to the time for this event, processes it, and disposes of the record containing the event information. It continues to do this until one of the four conditions for termination of the simulation have been met (see Section 5.3.6).

Figure 5.5 shows how the event calendar system works. Events which are scheduled to occur simultaneously, are processed in the order they were inserted into the calendar. In addition, the entity associated with a particular event is advanced as far through the network as possible before a new event is taken from the calendar. Events which occur instantaneously as a result of one entity's actions are also given priority over all other events. This is achieved by processing events recursively.

An activity duration may be expressed as the release time for another node or activity. Each time a release from a node occurs, all activities are checked for this condition.

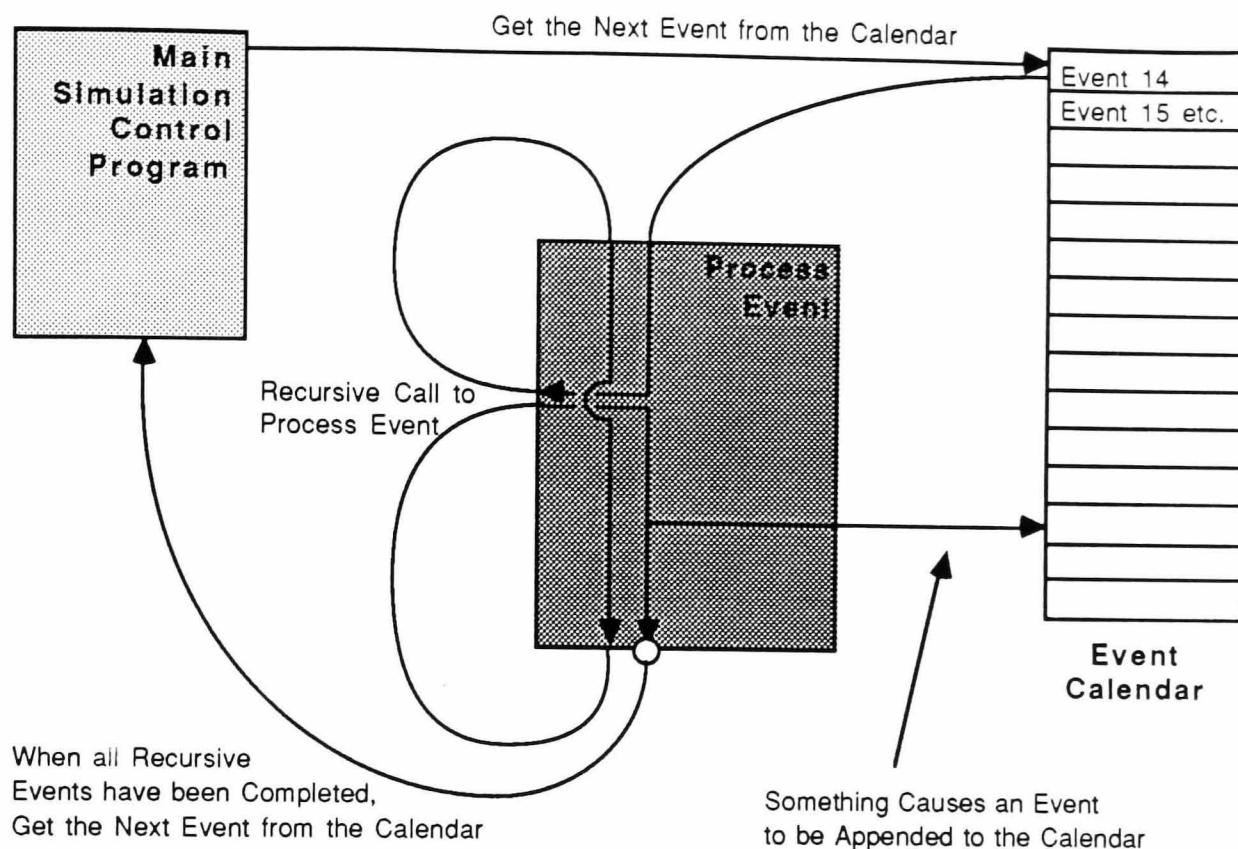


Figure 5.5 The Event Calendar

5.3.2 Routing Entities From Nodes (Branches) -->

In general, more than one branch may stem from a node. When this is the case and a release occurs, the entity is duplicated and simultaneously routed along some or all of them. There are several ways in which the number and choice of selected branches can be controlled.

First, most node types have a field which stores the maximum number of entities, E , emitted when a release occurs. This number may not be reached. In fact, if E is greater than the number of branches, it will never be reached. For example, for non-service activities, $E = \infty$ which suggests that all possible branches should be taken. Where E is less than the number of emanating branches, the first E 'acceptable' branches are selected.

The question of a branch's acceptability, depends on its associated probability or condition*:-

- ❑ Probabilities are used to select one from a group of n possible branches stemming from the node. The sum of the probabilities in a group must equate to 1.
- ❑ If an branch is assigned a condition and the condition is met then an entity is routed along that branch.

Branches, conditional activities and groups of probabilistic activities are tested in this way until E entities have been routed from the node. The remaining branches (if any) are disregarded. For example, in Fig. 5.6, there is a 70% chance that A2 is to be selected as opposed to A1. A4 and A5 form a second group of probabilistic activities which will only be tested if the condition specified in A3 is not met.

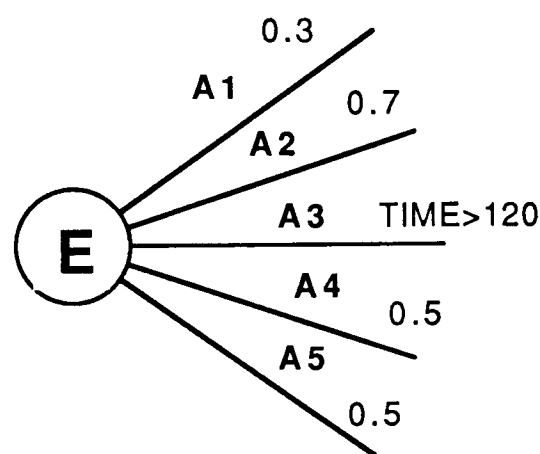


Figure 5.6 Routing Entities Example

* Branches which directly link one node to another, and activities with no associated probability or condition are assumed to have a probability of 1 or a condition which is permanently true.

5.3.3 Inserting Entities Into a System. (Enter Nodes)



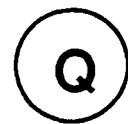
An entity can be entered into a system in two ways: the first is by pre-defining it and inserting it into a specified position in the network (Section 5.6.3) (Here, the user is able to set all the required attribute values for the entity before simulation begins); the second is by using an enter node.

Specified for the enter node are:

- the time at which the first entry is to be made;
- the time between entries which occur thereafter (TBE);
- the number of entries to be made and
- the attribute in which the entry time for an entity is stored.

At the start of a simulation run, the first entry is inserted in the event calendar according to the time specified. The time for subsequent entries is determined as the preceding ones occur, by adding a new value calculated for the time between entries to the current time. Entities continue to be created in this way until the required number of entries has been reached or the simulation terminates.

5.3.4 Queue Nodes



Queues develop in a network whenever entities must wait for an activity with a limited capacity to become available (see Section 5.3.7). In these places, queue nodes must be inserted to store the waiting entities.

Specified for each queue node are:

- the initial length of the queue;
- the maximum length of the queue;
- an overflow facility and
- a priority rule deciding the order in which entities must wait to be released (Table 5.1).

At the start of a simulation, entities may already reside in the queue from a previous run or having just been inserted. Whatever the current situation, the number of new entities required to make the length of the queue equal to the initial

length, are automatically created and inserted into it. These are then routed to any available service activities which follow the node.

During a simulation run, when an entity arrives at a queue node and a following activity is idle, the entity passes straight through the node and is taken up by the activity. Conversely, if no activities are available, the entity is inserted into the queue node according to the specified priority. When, at a later time, an activity does become available, the entity at the head of the queue is removed and taken up.

When an entity arrives at a queue which is full to capacity, the entity will, by default, be disposed of. However, it is possible to specify an overflow facility (Table 5.2) to route the node elsewhere or withhold it temporarily until the blockage is cleared.

Priority Rule	Wait Order
FIFO (first in first out)	Order of arrival at the queue
LIFO (last in first out)	Reverse order of arrival at the queue
High(A)	Order based on the highest current value of attribute A
Low(A)	Order based on the lowest current value of attribute A

Table 5.1 Priority Rules For Waiting in Queues

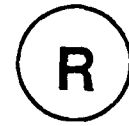
Overflow Facility	Meaning
Block	Block a preceding branch (must be a service activity)
Balk (node_name)	Send the entity to a secondary node identified by node_name.

Table 5.2 Queue Overflow Facilities

5.3.5 Continue Nodes

Continue nodes are inserted in a network wherever entities are to be duplicated and routed but no other processing is required.

5.3.6 Deleting Entities from a System (Remove Nodes)



Entities are deleted from a system using remove nodes. When an entity reaches a remove node, the number of deletions which have taken place at the node is incremented by one. If this number reaches a predefined maximum, the simulation run is terminated.

Other causes of run termination are:

- when all the current entities are waiting for an event and there are none left in the event calendar;
- when the next event in the event calendar is scheduled to occur after a prespecified simulation stop time;
- when the simulation is stopped interactively by the user.

5.3.7 Service Activities

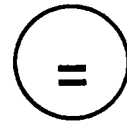


Service activities are those which are limited to handling a finite number of entities simultaneously. An example is a shovel which is only able to load one truck at a time. A service activity is immediately recognisable since, by definition, it must always be preceded by a queue or a decision node.

An entity arriving at a queue, is only allowed to proceed if one or more of the following activities are currently below their serving capacity, NumP. If this is the case, the choice of activity is made either on a probability or condition basis (Section 5.3.2) or through an intermediate decision node (Section 5.3.12). A group of probabilistic activities are used to specify the different durations and routings which are feasible for a single service and as such must all have the same name and capacity. Since only one entity can emanate from a queue node per release, the first activity which is both available and acceptable, is the one selected. If a decision node is used, probability or condition means nothing.

The scheduled completion for the selected activity is appended to the event calendar and statistical data, based on the time the activity spends processing multiplied by the number of servers being used, is updated. During the course of a simulation run, service activities may become blocked causing a reduction in the available capacity. When this happens, statistics are kept on the total time multiplied by the number of servers blocked.

5.3.8 Assign Nodes



An assign node is used to prescribe new values for any global variable or any attribute of an entity passing through the node. Assign equations should be of the form:

$$\text{Global}[9] := \text{Atrib}[2] - \text{Nrm}(10,1,1)$$

though no spaces need be left between variables and signs.

Up to 8 assignments can be made at each node. If more are required, a second assign node must be added in series with with the first.

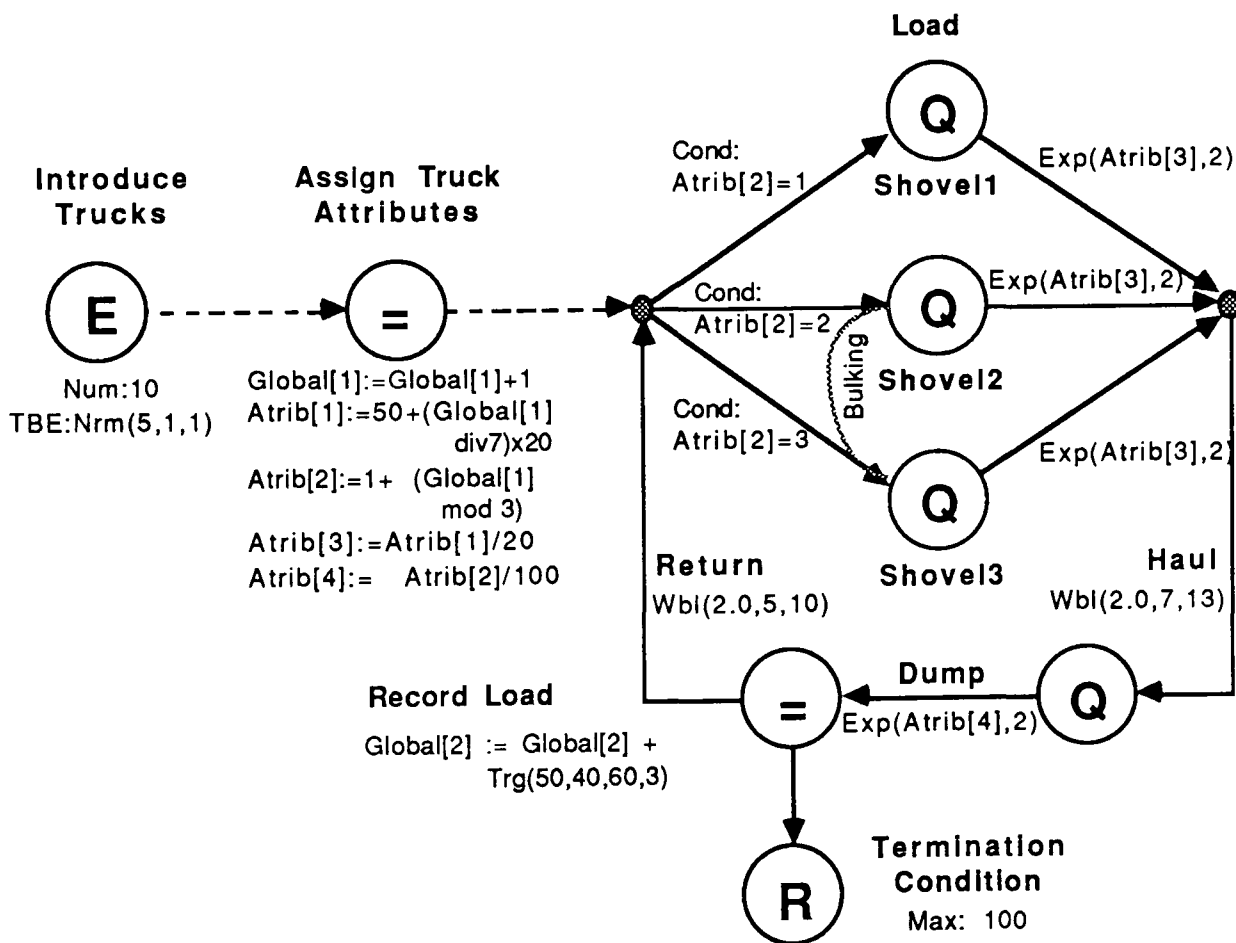


Figure 5.7 Three Shovel Example Network

5.3.9 A Three Shovel Example

The example illustrated in Figure 5.7 consists of 3 shovels loading material from the same area which is to be hauled to a single dump using 6 fifty-tonne and 4 seventy-tonne trucks. A fixed dispatching system is used, where the trucks are assigned to specific loaders throughout the simulation run. The haul and return times for a truck are independent of its payload but the load and dump times vary according to truck size. The system is to be simulated until 100 dumps have been completed.

The truck entities are introduced through the enter node at a rate which resembles that normally found in existing operations (approximately 1 truck every 5 minutes). They are immediately assigned attribute values which they will keep throughout the simulation. The attributes represent:

1. truck payload (First 6: 50 tonnes, Last 4: 70 tonnes)
2. shovel assignment (Done in rotation)
3. mean load time (Payload / 20)
4. mean dump time (Payload / 100)

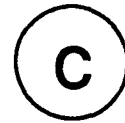
'Global[1]' is used as a truck counter. The trucks are then passed into the main haulage loop via a continue node.

The path taken from here depends on the value assigned to 'atrib[2]' which will only be able to fulfil one of the shovel selection conditions. Once routed, a truck may have to wait in a queue until its allocated shovel has finished loading all preceding trucks. A similar situation is encountered when the truck arrives at a dump. Owing to the proximity of shovels, a truck allocated to shovel 2 which sees that two trucks are already waiting when it arrives, may be temporarily reallocated to shovel 3 and vice versa. This is modelled as a balking overflow.

Having completed the loading, hauling and dumping sections, a value for the truck's actual load, considered to vary about the payload according to a triangular distribution, is calculated and added to the total (stored in 'Global[2]'). From here, the truck is returned to the continue node where it entered the haulage cycle and a dummy truck entity is routed to a remove node. When the hundredth dummy entity arrives at the remove node, 100 loads have been completed and the simulation is terminated.

This example forms the basis of a dispatching model. To simulate dynamic dispatching, assignment of the shovel allocation attribute (atrib[2]) would be carried out, according to more advanced criteria, within the haulage loop.

5.3.10 Accumulation (Collect Nodes)



Branching from a collect node is only allowed once a predefined number of entities have accumulated at it. The properties which must be specified for a collect node are:

- an initial release requirement;
- a subsequent release requirement, s ;
- a code specifying which entity is to be saved.

The first release takes place after the initial requirement has been reached. Thereafter, s entities must arrive at the node before each subsequent release can occur. A collect node has the capacity to store just one entity which, if necessary, gets duplicated on release. The attributes of the saved entity depend on the value specified for the save code (Table 5.3)

Code	Entity whose Attributes are Saved
First	The first entity arriving at the node after the last release.
Last	The entity which causes the release to occur.
High(i)	The entity with the highest value of attribute i
Low(i)	The entity with the lowest value of attribute i
Sum	A new entity with attributes equal to the sum of those accumulated
Product	A new entity with attributes equal to the product of those accumulated.

Table 5.3 Saved Entities

5.3.11 Data Recording (Stats Nodes)



A stats node is used to gather data on global variables or attributes of entities passing through the node. The information collected depends on a specified value which can be any single item or NUsim function. Its mean and standard deviation are continuously updated as are the minimum and maximum recorded values each

time these are exceeded. In addition, it is possible to obtain a graph of the collected data. This is achieved by specifying the number of cells (numcells) for the frequency polygon and an upper and lower limit for the complete range of expected values (see Section 5.5.1).

5.3.12 Decision Nodes



These are inserted whenever decisions regarding the route that an entity should take are required. For example;

- When an entity has to choose between a number of parallel queues;
- When an activity has just been completed and one entity has to be selected from a number of non-empty queues;
- When an entity has to choose between a number of idle service activities;

The first two cases require a queue selection rule upon which decisions can be based (Table 5.4). The latter requires an activity selection rule (Table 5.5). It is possible to combine queue selection and activity selection decisions on a single node. This is required when a number of parallel queues are linked to a number of non-identical parallel servers. However, since queues cannot lie on both sides of a decision node, there is never a cause for two queue selection rules to be combined.

It is also possible for decision nodes to become blocked. This occurs when the node is followed by a number of queues each of which is full to capacity. The same overflow rules apply here as when a queue is full to capacity (Table 5.2).

Rule	Definition
Order	Priority given in the order in which queues following the select node are linked or queues preceding the select node are defined.
Cyclic	Priority given to the queue which follows the last one selected. (same order as above)
Random	Queue selected at random.
Largest Mean	Queue with the largest average number of entities in it to date
Smallest Mean	Queue with the smallest average number of entities in it to date.
Longest Wait	Queue which has been waiting longest since its last release.
Shortest Wait	Queue which has been waiting shortest since its last release.
Longest Queue	Queue which currently has the greatest number of entities in it.
Shortest Queue	Queue which currently has the least number of entities in it.
Largest Space	Queue which currently has the greatest unused capacity
Smallest Space	Queue which currently has the least unused capacity
Each Queue (selection from queues only)	Entity to be released only when every queue preceding a service activity contains at least one entity. The attributes of the entity to be released are saved according to table 5.3, (see collect node)

Table 5.4 Queue Selection Rules

Rule	Definition
Order	Priority given in the order in which service activities following the queue are linked.
Cyclic	Priority given to the service activity which follows the last one selected (same order as above).
Random	Activity selected at random.
Most Busy	Activity with the largest mean utilisation to date.
Least Busy	Activity with the smallest mean utilisation to date.
Longest Idle	Activity which has been waiting longest since its last completion.
Shortest Idle	Activity which has been waiting shortest since its last completion.

Table 5.5 Activity Selection Rules

5.3.13 A Two Shovel Example

The example illustrated in Figure 5.8 consists of two shovels assisted by a single bulldozer which piles up material for loading. A truck arriving at the loading area, must wait not only for one of the shovels to become available but also for there to be a sufficient load accumulated by the dozer. After being loaded, the trucks haul material to a single crushing area where they must select one of two hoppers in which to deposit their loads. Statistics are required on the mean time taken for an excavated load to reach the crusher.

Here, trucks are entered by placing them directly into the various queue nodes which make up the network and simulation termination is handled by specifying a stop time. No truck entry or remove nodes are therefore required. Loads are accumulated using a collect node which is supplied by dozing at a rate of approximately 2 minutes per load. Before either shovel can begin loading, at least 3 dozer loads must be available, thereafter, a residual amount is usually left between trucks and only two additional dozer loads are required. The dozer load entity saved at the collect node is the one with the smallest value of `atrib[6]`. This represents the earliest extraction time for the load and is assigned automatically on entry into the system.

The simultaneous requirement of shovel, load and truck availability is modelled by a decision node which uses an 'Each Queue' queue selection rule. When all three nodes contain at least one entity, a new one is created with the sum of all three sets of attributes saved. This is then allocated according to the available shovel (If both are available the 'Longest Idle' activity selection rule is applied). On completion of loading, a dummy truck entity is looped back to the shovels queue via an assign node which sets all its attributes to zero. This ensures that shovel entities have no effect at the summation stage.

The crusher decision node is used to select destination queues ahead rather than supply queues behind. The decision is based on the shortest queue at the time of arrival. Before returning to the loading area and having recorded its load, the current time minus the load entry time, stored in `atrib[6]`, is recorded at a stats node. This will later be interrogated to create a frequency histogram of the time a load spends in the system. To avoid later misinterpretation, `atrib[6]` is set back to zero immediately afterwards.

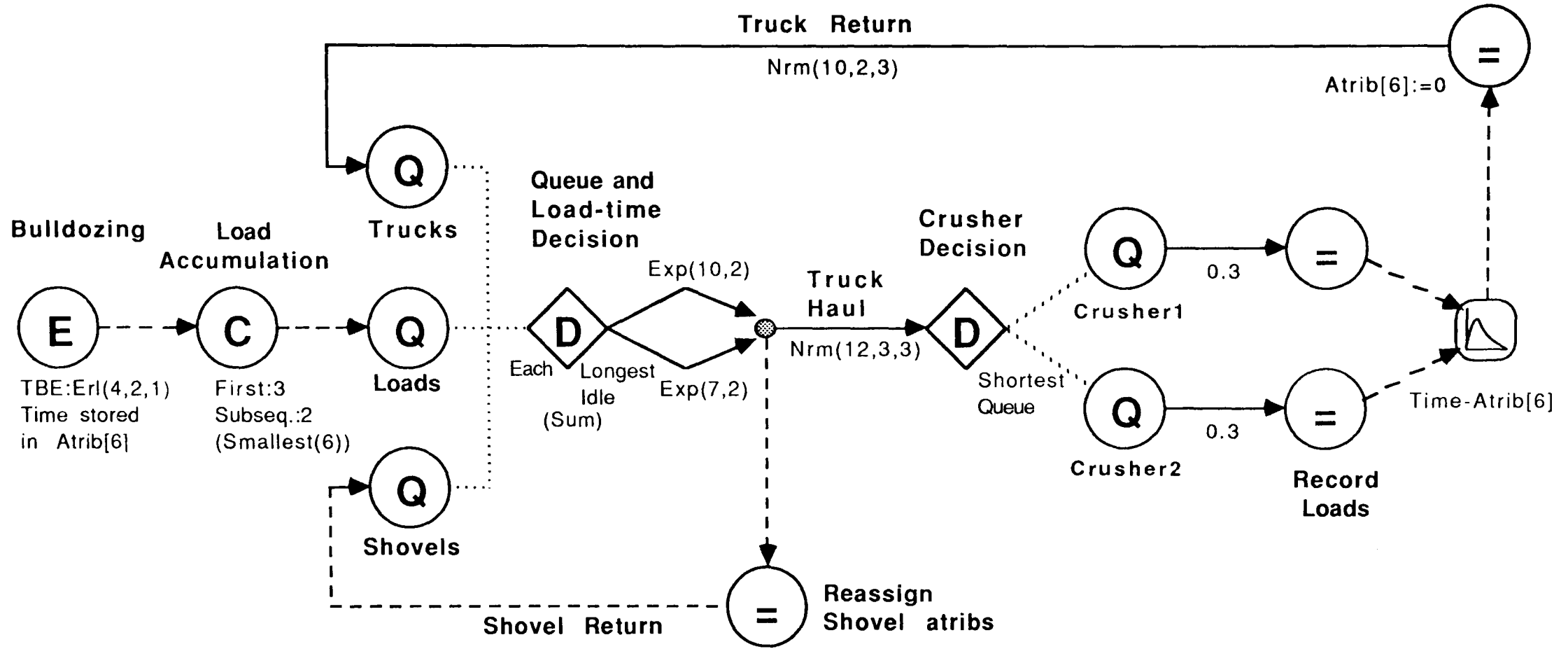


Figure 5.8 Two Ore Shovels Assisted by a Single Dozer

5.3.14 Match Nodes



Entities are only allowed to pass through match nodes when the value of a specified attribute is the same for at least one entity currently residing in each of the preceding queue nodes. Here, routing of entities is carried out on an individual basis i.e. for each queue preceding the match node there is a corresponding branch on the other side to which entities from the queue are routed.

If more than one match is linked to from a queue, priority is given to the match with the highest specified matchorder. If no match is made here, the next match is tried and so on. Alternatively, a decision node can be inserted between queues and matches to determine which match is given priority. The match selection rules are identical to the activity selection rules given in Table 5.5. The number of matches made to date is retained as a measure of match utilisation.

When editing matches manually, both the match nodes and the corresponding out routes are linked to directly from the source queue. To distinguish them from other match combinations associated with the queue, they are always defined together and in the order shown (Fig 5.9). If a *select* node is to be used, it is also linked to directly and must be specified before any match combinations. No branches ever stem from a match though this may appear to be the case when viewing the network graphically.

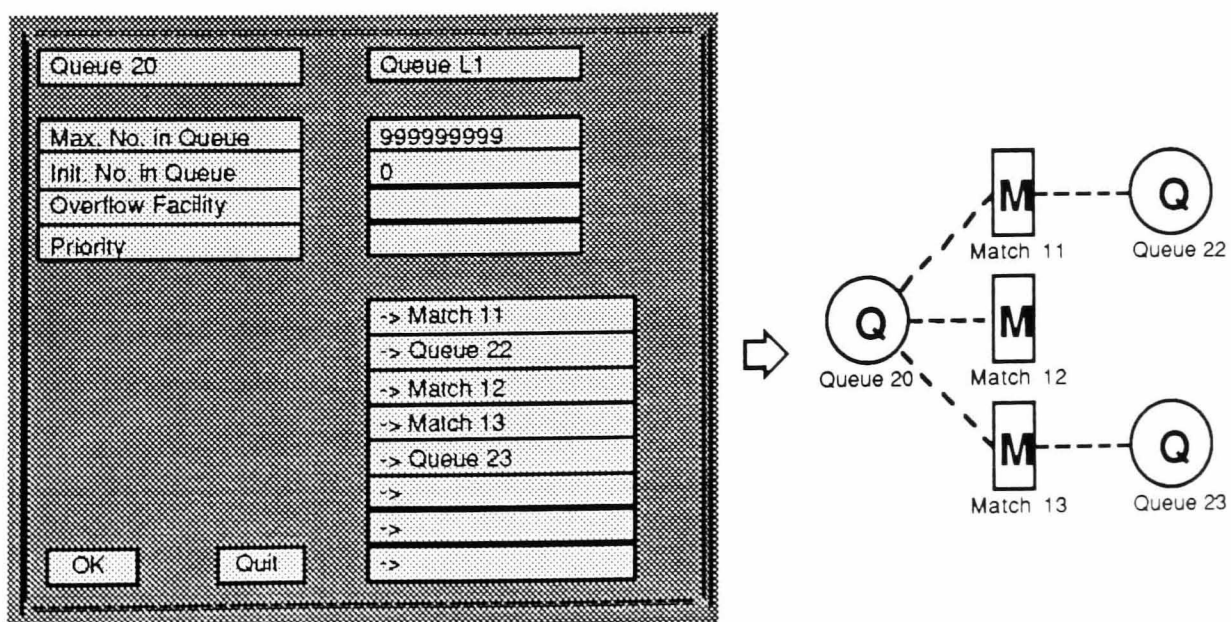


Figure 5.9 Branching to Matches from Queues

5.4 Resources and Switches

At various times throughout a process, the completion of a set of activities which make up part of the process may depend on the availability of certain resources, e.g. truck drivers, fuel etc.

In a simulation model, the flow of entities is restricted by the availability of analogous resources. Generally, at the start of a simulation run, the number of units of each resource type (its capacity) is stipulated. Thereafter, individual or groups of units may be seized and later released by entities flowing through the network or the total number of units of a particular resource may be changed.

Switches are a particular type of resource which can be thought of as having either an infinite number of units (on) or none at all (off). When a switch is on all entities are allowed to flow through and when it's off, none are.

5.4.1 Wait Nodes



An entity arriving at a wait node is only allowed to proceed once the required number of units of each resource type are available. Until this is the case, entities are stored in the node, in the same way that entities accumulate at queues. Specified for a wait node are:

- An array of resources together with the number of units of each required (specified in brackets).
- A priority rule used to decide the order in which entities should be removed from the line of those waiting.
- An order number which stipulates the relative priority of the node itself.

No restrictions are placed on the number of entities which can wait in a wait node. Therefore, no blocking or balking rules need be considered. Statistics are kept on the mean and standard deviation of the number of entities waiting at one time, as well as the maximum and minimum value. Those wait nodes which, due to the recent availability of certain resources, simultaneously fulfil all the criteria for a release to take place, are separated according to their priority order. As many entities as possible are routed from the node with the lowest value of 'order' before the node with the next lowest is tested. No unit requirement need be stipulated for switches since, when open, an infinite number is assumed to be available.

5.4.2 Free Nodes



When a group of resource units have been seized by a particular entity, they must, at some stage, be released. This is achieved when the entity passes through a free node. A free node consists of an array of strings which can be used to store up to 8 normal resources together with the number of units of each to be released. When units are freed, all wait nodes which associate with the affected resources are checked, in order of relative priority, to see if the new level of availability is sufficient to meet their own release requirements.

5.4.3 Alter Nodes



These are used to change the current capacity of a resource. If the number of units of a resource is to be reduced and there is an insufficient number of units currently available to effect the whole reduction, those which are available are removed from circulation immediately and the rest of the alteration takes place as more become available. i.e. once more of the remaining ones have been released from entities. The number of units of a particular resource can never be negative.

5.4.4 On Nodes



An on node turns a switch on. When this occurs, all the wait nodes associated with the switch are checked. Any wait nodes which depend on the current status of the switch alone, will be in a position to release all their entities and will do so unless the switch is instantaneously turned off by one of the first released.

5.4.5 Off Nodes



An entity passing through an off node causes all switches with a specified name to be turned off.

5.4.6 A Single Shovel Example

Figure 5.10 illustrates the use of match nodes and resources in a simulation model. The main network represents the somewhat unusual configuration of a single shovel loading both coal and waste which must be hauled to separate dumping sites. Priority is given to coal loading but this can only take place after the coal has been cleaned, i.e. when all remaining overburden has been stripped from its upper surface by dozers. This is carried out at a mean rate of one full truck load every 5 minutes. When no coal is available, the shovel spends its time extracting and loading overburden.

Of the 15 trucks which make up the fleet, 4 are road vehicles which can only haul coal and 5 are large off-highway trucks which can only haul waste. The remaining 6 are able to transport both types of material. The truck entities are assigned and entered manually with attributes three and four used to indicate loading suitability. As examples: for the lorries, 'atrib[3]' will equal 1 and 'atrib[4]' will equal 0; for entities representing the shovel and the dual-purpose trucks, both the attributes will equal 1.

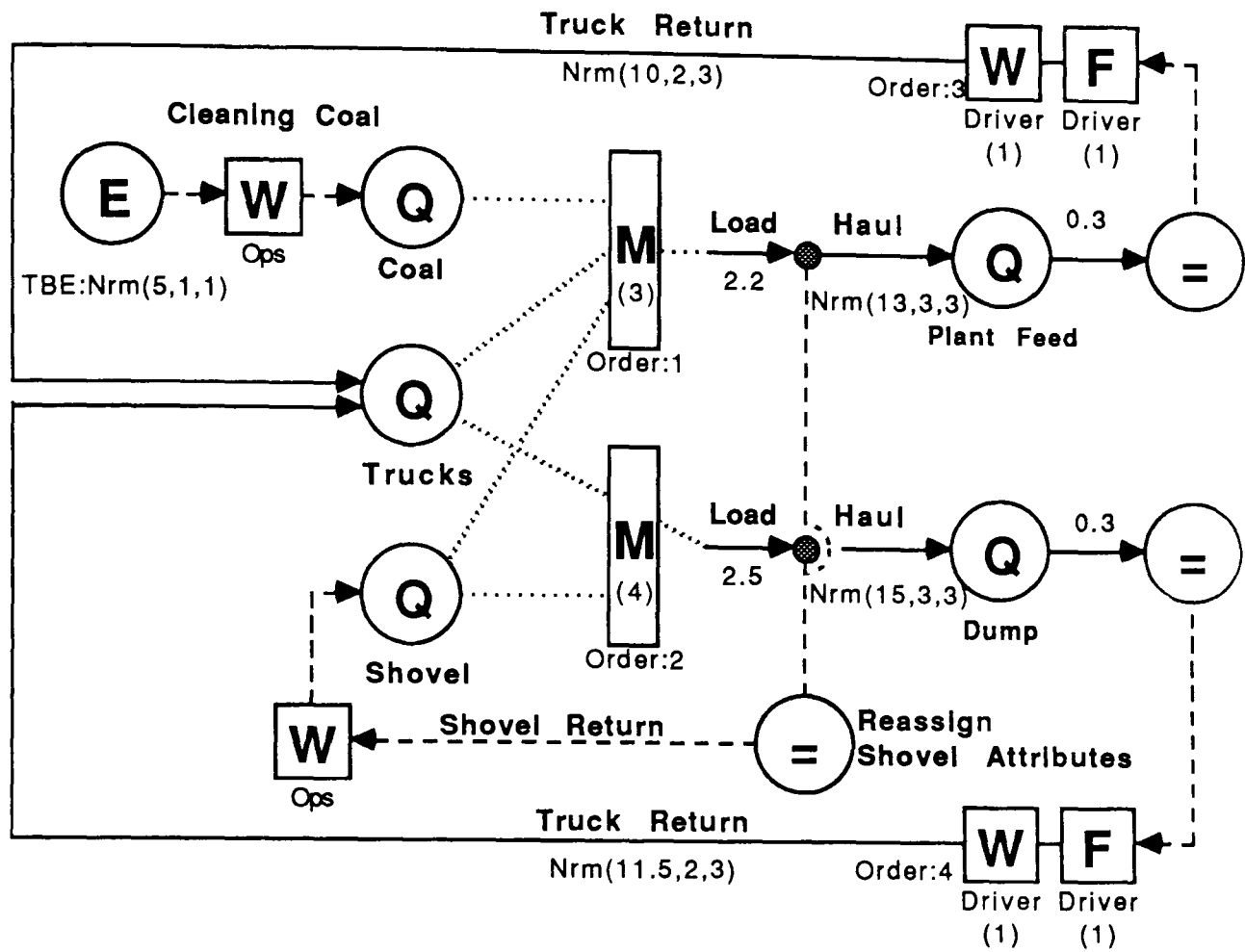
The selective loading process is modelled using two match nodes in parallel. When sufficient coal, a truck which is able to transport it and the shovel, are all available, the match node with the highest priority will initiate coaling and route the truck off to the plant. Otherwise, the existence of one entity in each of the lower two queues, whose 'atrib[4]'s are both equal to 1, will initiate waste removal to the dump.

Built into the example, is a shift model which makes use of both resources and switches to simulate a week of operations. An entity loops round the network at a rate of one revolution per day and run termination is enforced after a 5 day period.

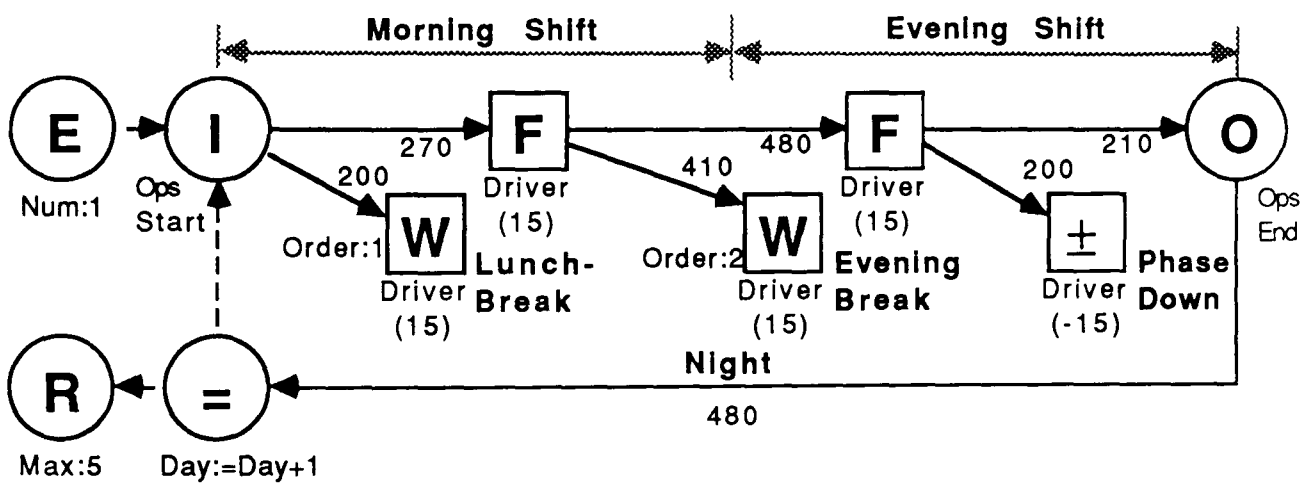
At the start of every morning shift, the ops switch is turned on which allows cleaning and shovel loading to begin. However, the truck drivers arrive randomly over a half hour period and the mine takes time to build up to peak production. This has been modelled by inserting wait nodes in both the truck return loops (again, coal trucks are given priority over waste trucks) and altering the total number of driver resource units approximately every two minutes (bottom network).

Lunch breaks are scheduled to occur 3¹/₂ hours into the shift. Trucks which complete a cycle within 10 minutes of this are not likely to start another. Priority is given to wait nodes in the shift network, and individual units which are freed immediately prior to those in the main network, are transferred. The lunch break ends after 4¹/₂ hours have elapsed and all the driver entities are freed for operations to continue.

Main Network



Shift Network



Arrival of Truck Drivers

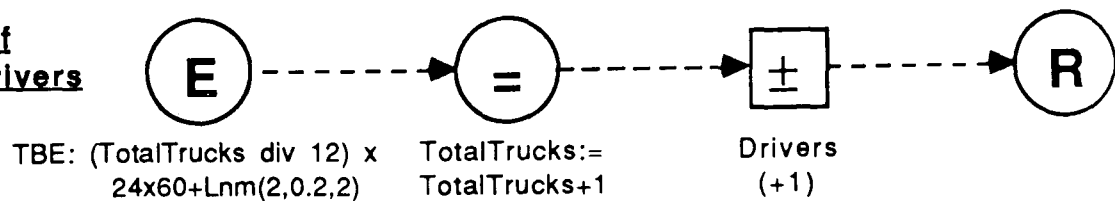


Figure 5.10 Single Shovel Loading Coal and Waste (with Shift Pattern)

The evening shift pattern is identical except that the changeover to it is carried out between cycles causing no appreciable delay. However, 10 minutes before the end of the evening shift, an attempt is made to withdraw all the driver units. This only takes full effect when all the current haulage cycles have been completed. The loading and cleaning operations cease after 16 hours.

5.5 NUsim Reports

Reports can be produced, at any time, from all network items, entities and resources which record statistics. For activities, queue nodes, and wait nodes, these are the ones which have been given a name; for entities, these are the ones inserted into the network manually. By definition, all stats nodes collect data, as do all resources.

The information reported varies according to its source. Tables 5.6 through to 5.10 show the output statistics which are associated with each.

Result	Meaning
Name	The activity's name.
Num.	The number of parallel servers represented by the activity.
Mean	The mean number of entities being processed at one time.
S.D.	The standard deviation of the number being processed at one time.
Total	The total number of entities processed to date.
Current	The current number of entities being processed.
Blocked	The mean number of entities blocked at the activity.
Max. Busy	The maximum number being processed (busy parallel servers).
Max. Idle	The maximum number of idle parallel servers (service activities only).

Table 5.6 Activity Statistics

Result	Meaning
Name	The node's name.
Mean	The mean number of entities waiting at one time.
S.D.	The standard deviation of the number of entities waiting at one time.
Min	The minimum number waiting.
Max	The maximum number waiting.
Current	The current number of entities waiting.

Table 5.7 Queue Node and Wait Node Statistics

Result	Meaning
Name	The node's name.
Num	The number of times data has been recorded at the node.
Mean	The mean recorded value.
S.D.	The standard deviation of the recorded value
Min	The minimum recorded value.
Max	The maximum recorded value.
First	The time of the first data collection.

Table 5.8 Stats Nodes Statistics

Result	Meaning
Name	The resource name.
Num	The current number of units of the resource
Mean	The mean utilisation of the resource over time.
S.D.	The standard deviation of its utilisation over time.
Min	Its minimum utilisation.
Max	Its maximum utilisation.
Current	Its current utilisation.

Table 5.9 Resource Statistics

Result	Meaning
Name	The entity's name.
Atrib[1] - Atrib[12]	The current values stored in up to twelve of the entity's attributes
Node	The entity's current location

Table 5.10 Entity Statistics

5.5.1 How Statistics are Maintained

It would, of course, be inconvenient to keep a record of every occasion when an alteration to the current state of an item causes its statistics to change. For this reason, alternative methods for calculating the required statistics were adopted.

The mean value of a property over time is generally derived from a running total of its instantaneous value. Whenever a change to a property occurs, the value of the property just before the change multiplied by the time since the last change is added to the running total. The mean value is then calculated from the equation:

$$m = \frac{\sum_{i=1}^n v_i \times t_i}{T} \quad \text{.. (5.1)}$$

where: m = the mean value of the property over time
 i = state index
 n = number of different states
 v_i = value of the property in a particular state
 t_i = time in a particular state
 T = total time

Standard deviations are calculated in a similar fashion but by maintaining a running total of the square of a property's instantaneous value:-

$$d = \sqrt{\frac{\sum_{i=1}^n v_i^2 \times t_i}{T} - m^2} \quad \text{.. (5.2)}$$

where d = the standard deviation of the property over time.

For activities, 'value' refers to the number of entities being processed at a time; for resources, it refers to the number of resources being utilised.

For queue and wait nodes, the program sets aside storage space in the form of a linked list of waiting positions. These are used to physically store entities until they are allowed to continue through the network. Each storage space also retains the amount of time it has been utilised. Just before an entity gets added to or removed from the line of those waiting, the total time of the last used storage space is updated. The sum of all the queue positions (or their squares) multiplied by their

respective utilisation times, represent the first items on the right hand sides of Equations 5.1 and 5.2.

Stats nodes take no account of time, hence, in the above equations, t_i would be taken as 1 and T would refer to the number of times data has been recorded at the node. If the number of cells in a stats node is specified as being greater than 0, a frequency polygon is to be produced depicting the variation in the values calculated for the function of interest. The frequency with which calculated values fall between the upper and lower limits of the defined cells are stored in an array (freq_array) and, when an entity passes through the node, the choice of incremented cell is calculated using the following algorithm:

```
let cell_width = (upper_limit - lower_limit) / numcells
let new_value = value - lower_limit
round new_value down to the nearest multiple of cell_width
then, the cell to be incremented (c) = (new_value / cell_width) + 1.
Hence, set freq_array[c] = freq_array[c] + 1
```

If the value produced lies outside this range, one of two extra cells (freq_array[0] and freq_array[numcells+1]) is incremented, depending on whether the value is below the lower limit or above the upper limit respectively.

5.5.2 How Statistics are Reported (Report Nodes)

The reporting of statistics is usually carried out through a report window (exceptions are the 'instant' reports requested during run-time see Simulation Controls). A report window allows the user to specify:

- the report heading;
- the report destination;
- the report type;
- a single report item (if the type is appropriate);
- a textfile or relation name (if the destination is appropriate) and
- a set of up to 12 NUsim functions.

The report destination indicates where the report is to be generated. The options are as follows:

- output to the screen;
- output to the printer;
- output to an external textfile;
- transfer the results into relation format (see Results Relations);

Outputs to the screen, the printer and to external textfiles take on an identical format. (Figure 5.11 shows a typical output produced with a full report). The advantages of using a printer or an external textfile are that these provide a permanent copy.

Though a full report is often required, the report type field may be used to single out items for observation. Hence, all activities, all queues etc. may be displayed independently from other node types, or single items may be viewed on their own. The 12 NUsim functions allow the user to view the current value of a variable or the calculated value of an important function. For example, he may wish to know the last time a release was made at a particular node. When entities are involved, the functions may include attribute variables.

Reports may be generated automatically during run-time by inserting report nodes into the network. When an entity passes through, these have exactly the same effect as a report request from the main pull-down menu.

NUsim REPORT

Network : Three Shovels

Report Name : Command

Start Time : 0.00 Current Time : 943.78

Activity Statistics

Activity	Cap	Mean	S.D.	Tot	Cur	Blocked	MaxBusy	MaxIdle
Load L1	1	0.1157	0.3198	58	0	0.0000	4.77	103.59
Load L2	1	0.1173	0.3218	57	0	0.0000	3.72	103.16
Load L3	1	0.1375	0.3444	67	0	0.0000	3.80	102.34
Return	1	0.2967	0.4568	181	0	0.0000	43.21	7.17
Dump	1	0.2214	0.4125	182	0	0.0000	0.86	79.94
Haul	1	0.2471	0.4256	182	0	0.0000	37.85	5.57

Queue Statistics

File	Mean	S.D.	Min	Max	Cur
Queue L1	0.5414	0.1270	0	1	0
Queue L2	0.4872	0.0458	0	1	0
Queue L3	1.0347	0.2563	0	3	1
Queue Dump	5.4312	1.1281	3	9	6

Wait Statistics

Collected Statistics

Variable Values

L1	7787.358
L2	7624.912
L3	8830.502
Dumped	24192.781

Entity Statistics

Truck 1	At	13 - Queue Dump	1.000	136.000	0.000	4587.358	33.000	1.000
			0.000	0.000	0.000	0.000	0.000	0.000
Truck 2	At	13 - Queue Dump	1.000	136.000	0.000	3624.912	26.000	1.000
			0.000	0.000	0.000	0.000	0.000	0.000
Truck 3	At	10 - Haul	1.000	136.000	0.000	4430.502	31.000	1.000
			0.000	0.000	0.000	0.000	0.000	0.000
Truck 4	At	13 - Queue Dump	1.000	136.000	0.000	4277.055	29.000	1.000
			0.000	0.000	0.000	0.000	0.000	0.000
Truck 5	At	5 - Queue L3	1.000	136.000	0.000	3358.071	21.000	1.000
			0.000	0.000	0.000	0.000	0.000	0.000

Figure 5.11 A Typical NUsim Output Report

5.6 Program Structure

The NUsim module is simply a tool which can be called up by any other module in the NUmine system. The main processes which are handled within the module are:

- network definition;
- initialisation of the various parameters required;
- compilation;
- the simulation run itself and
- termination.

The volume of diverse information required to model a complete process using NUsim requires the utilisation of a purpose-built project. Network definition and parameter initialisation involves the entry of or adjustment to data stored in a number of the project's relations, the various types of which will be detailed below.

5.6.1 Network Definition

A NUsim network can be constructed either manually or graphically. Both make extensive use of the windowing environment described in Section 3.4.6. A network consists of an interlinked list of nodes similar to a picture's structure in that the branches stemming from one node to another are linked directly rather than by reference, (Fig. 5.4). Nodes may be added to the linked list by calling the appropriate 'add-' procedure, e.g. add-activity, add-queue etc. from within the program itself. However, more often, the list of node types are displayed in a pull-down or graphics menu for the user to select from (in graphics mode, the symbol which represents the new node is dragged into position on the existing network). The window associated with that particular node type is then displayed with default properties specified so that only those which require alteration need be entered.

A node may be edited either by selecting it from a pull-down menu of all existing nodes or by double-clicking on its graphical symbol. All the existing nodes of a particular type may be edited simultaneously using the multiwindow facility. Selected nodes can also be deleted from the network. Activities are treated as nodes since they fit into the network in an identical fashion.

Once created, the network can be saved in a relation and reloaded at any time. To avoid the need for having an excessive number of relations open at the same time, the format used is non-relational. The use of a code field to indicate where new nodes begin and the type of attribute stored in each record, followed by a string field (Strings can be used to store any type of data), reduces the total storage space required and simplifies the problem of retaining multiple branches from the same node. The structure of a network relation is of no importance to the user who is able to view and edit the network much more easily in two completely different environments.

5.6.2 The 'Initialise' Relation

The relation which is currently accessed by the 'Initialise' identifier is used to store a list of those assignments which are to take place immediately prior to any simulation run. The first record must be used to set the simulation start time and this is the only occasion where TIME is allowed to form the left hand side of an assignment statement.

5.6.3 The 'Entities' Relation

The relation accessed by 'entities' is used to store the permanent entities which form an inherent part of problem being studied and as such are to be entered into the NUsim network automatically, e.g. trucks in a haulage model. For each entity is specified:

- a reference number;
- a name;
- a series of attributes and
- a current node location.

Though the maximum number of attributes per entity is 20, the number which can be stored in a relation is, at present, limited to 12.

When a call to load-entities is requested, entities in the relation are converted into a linked list which allows for rapid searching and manipulation during run-time. At the start of a run, the entities are taken in their list order,

inserted into the network in their specified locations and, if possible routed away. Thereafter, temporary entities, created during the simulation run at an enter node or by multiple release from any other node, are appended to the list but are distinguishable because they have no name. At the end of the run, any remaining temporary ones are removed leaving only those permanent entities which have been retained throughout the simulated period. Most, if not all, of these will have changed location but will be left as they are unless a new call to load-entities is made.

Entities may also be stored in relations which are normally accessible from an outer level. In this way, the types of entity associated with a particular problem can be defined in terms of their actual properties rather than as a series of attributes. In this case, a routine analagous to load-entities must be written for the transfer from relational into linked list format.

5.6.4 The 'Resources' Relation

The current 'resources' relation contains a list of the various resource types together with the number of units of each type which currently exist. The full field list consists of:

- a reference number;
- a name;
- a type specifying whether the record refers to a normal resource or a switch;
- a number used to store the resource capacity or whether the initial status of a switch is either open (1) or closed (0).

When the procedure load-resources is requested, resources in the relation are converted into a linked list in a similar manner to the entities above, and during the run, statistics are maintained on the utilisation of each. At any time during the simulation period, the number of units of each resource type can be reverted back to the values specified in the relation (reset_resources) though the completion of negative changes may be delayed, like those resulting from alter nodes (Section 5.4.3), until the required number of units become available. Units of resource which are in use at the end of a simulation run, remain in use unless a further call to load-resources is made. Here the linked list is completely recreated and all properties are reset to their initial values. Like entities, resources may be defined at

an outer level by specific routines which create the linked list based on information stored in different relations.

5.6.5 The 'Variables' Relation

The relation currently accessed by 'variables' is used to store the names and definitions of variables associated with a particular problem. For example, the payload of a truck may be stored in attribute 3 of a truck entity. Rather than refer to it throughout the simulation network as 'atrib[3]', it would be more convenient to actually call it 'Payload' and let the simulation program handle the decoding mechanism (Section 5.2.1). In this case, the name of the variable would be 'Payload' and its definition 'atrib[3]'.

The string stored in the variable definition field may be any NUsim function, e.g. `NRM(Payload,10,1)`. Every time this type of variable is referenced during a simulation run, a new value is calculated. No value can be assigned to it specifically since there is no storage space put by for its use. As a result, the single items which make up the left hand side of an assign statement must be defined as either a global variable or an entity attribute, e.g. `Load (defined as 'atrib[6]) := Nrm (payload,10,1)`.

5.6.6 The 'Seeds' Relation

The 'Seeds' relation is used to store the initial value in each seedbed. A call to `reset-seeds` at any time causes these values to be inserted into the seed array and the next time a random number is requested, the new value in the specified seedbed will be used. Due to the way in which pseudo-random numbers are generated by the in-built Hewlett Packard routines, the ideal initial value for a seed is an integer with several digits where the least significant digit is a 1, a 3 or a 7.

5.6.7 'Results' Relations

These relations are used to collect statistics at various times throughout a simulation run and after. In addition to the regular information gathered for each node type, the report node name and the time the report was generated are also itemised. A call to `rewrite_results` clears all the information currently stored in the results relations but prompts the user for confirmation before doing so.

The 'Graph Results' relation is used to store any frequency curves generated at stats nodes. If necessary, these can be viewed graphically using the standard graph plotting routines available within the screen editor.

5.6.8 Simulation Controls

A simulation is invoked by the 'Run' command. This causes the screen to be refreshed and a set of single character options, which override the main pull-down menu, to be displayed along its bottom line. As indicated in Section 5.3.1, a run is little more than a repeat loop which produces a list of results. However, the user is able to pause the run temporarily (using the space bar) or stop it completely (using the select key). The 'Stop' command allows him to regain access to the main command level from where, the run can be either continued or terminated manually. A subsequent call to 'Run' will cause an unfinished simulation to be terminated before the new one is allowed to begin. Changes to the network structure or project definition may also enforce a premature termination.

There are many occasions, particularly during network testing and validation, when a facility for observing run-time transactions as they occur is required. The two main run-time controls are the Trace switch and the Step switch. Each can be activated or deactivated from single options on the main pull-down menu or, in simulation mode, by typing a 'T' or an 'S' respectively. The Trace switch, when on, causes messages or groups of associated messages to be printed out on the screen or printer as transactions occur (Fig. 5.12). The Step switch causes the simulation to halt between groups of transactions until the user either turns the switch off or presses the return key. The output destination is determined from the current status of the 'Output' switch (printer-on). This, again, can be altered from either the main menu or by typing a character ('P'). The grouping is a result of the recursive manner in which events are processed.

A number of other control parameters exist which may be activated or deactivated in simulation mode only. A full list is given in Table 5.11.

Control commands may be inserted into the network using control nodes. When an entity passes through a control node, the commands declared in the node have exactly the same effect as those activated manually. A trace, for example, can be enforced automatically over a specified period.

```

43.03 Truck 2 Completed Activity 10 - Haul
43.03 Truck 2 Inserted in Queue 13 - Queue Dump
43.31 Truck 4 Completed Activity 2 - Return
43.31 Truck 4 Inserted in Queue 3 - Queue L1
43.31 Truck 4 Removed from Queue 3 - Queue L1
43.31 Truck 4 Scheduled To Complete Activity 6 - Load L1 at 45.24
45.24 Truck 4 Completed Activity 6 - Load L1
45.24 Truck 4 Released from Continue 9 -
45.24 Truck 4 Scheduled To Complete Activity 10 - Haul at 48.32
46.27 Truck 3 Completed Activity 2 - Return
46.27 Truck 3 Inserted in Queue 5 - Queue L3
46.27 Truck 3 Removed from Queue 5 - Queue L3
46.27 Truck 3 Scheduled To Complete Activity 8 - Load L3 at 49.01
46.34 Truck 1 Completed Activity 2 - Return
46.34 Truck 1 Inserted in Queue 5 - Queue L3
48.32 Truck 4 Completed Activity 10 - Haul
48.32 Truck 4 Inserted in Queue 13 - Queue Dump
49.01 Truck 3 Completed Activity 8 - Load L3
49.01 Truck 3 Released from Continue 9 -
49.01 Truck 3 Scheduled To Complete Activity 10 - Haul at 51.58
49.01 Truck 1 Removed from Queue 5 - Queue L3
49.01 Truck 1 Scheduled To Complete Activity 8 at 50.24
49.51 Truck 5 Completed Activity 14 - Dump
49.51 Truck 5 Assignment at Assign 1 -
49.51 Truck 5 Scheduled To Complete Activity 2 - Return at 54.29
49.51 Truck 2 Removed From Queue 13 - Dump
49.51 Truck 2 Scheduled To Complete Activity 14 - Dump at 50.47
50.24 Truck 1 Completed Activity 8 - Load L3
50.24 Truck 1 Released from Continue 9 -
50.24 Truck 1 Scheduled To Complete Activity 10 - Haul at 52.11
50.47 Truck 2 Completed Activity 14 - Dump
50.47 Truck 2 Assignment at Assign 1 -
50.47 Truck 2 Scheduled To Complete Activity 2 - Return at 54.10
50.47 Truck 4 Removed From Queue 13 - Queue Dump
50.47 Truck 4 Scheduled To Complete Activity 14 - Dump at 52.06
51.58 Truck 3 Completed Activity 10 - Haul
51.58 Truck 3 Inserted in Queue 13 - Queue Dump
52.00 Truck 1 Completed Activity 10 - Haul
52.00 Truck 1 Inserted in Queue 13 - Queue Dump

```

Figure 5.12 NUsim Trace Report

Command	Character	Meaning
Stop	SELECT	Stop the run and return to main command level.
Pause	SPACE BAR	Pause the simulation run temporarily.
Continue	RETURN	Continue a run which has been paused or is currently in step mode.
Trace On/Off	'T'	Turn the trace on or off.
Step On/Off	'S'	Turn the Step mechanism on or off.
Printer On/Off	'P'	Direct output to either the printer or the console.
Speed	'X'	Adjust the speed of the simulation run.
Activities	'A'	Obtain an instant report on all the named activities.
Queues	'Q'	Obtain an instant report on all the named queues.
Waits	'W'	Obtain an instant report on all the named wait nodes
Stats. Data	'D'	Obtain an instant report from all the stats nodes.
Resources	'R'	Obtain an instant report on all the resources.

Table 5.11 Simulation Controls

5.7 Conclusions

NUsim can be thought of as a shell for simulation network formulation and utilisation. It retains all the advantages of a simulation language but is not limited by the inflexibility and incompatibility which is normally associated with one. Instead, it interfaces directly with other modules in the NUMine system and can therefore be made to use information about the operation to be modeled which already exists.

Unlike the NUMine user-interface and NUgraph, NUsim was built as a general-purpose system from the outset. While the requirements of this project could have been fulfilled with a slightly less complex package, the author felt that the extent to which simulation is used to derive selection and scheduling solutions meant that the development of a complete system was necessary. In the long term, these will also make any enhancements to the haulage analysis program a great deal simpler to achieve.

This chapter has described NUsim in some detail. Examples of how network modelling can be applied to the analysis of truck/shovel systems have been included and the way these are adopted for more general use in the equipment scheduling program is described further throughout Chapter 6.

Chapter 6

***Haulage Equipment Selection and
Scheduling Module***

Chapter 6

Haulage Equipment Selection and Scheduling Module

6.1 Introduction

Having described, in some detail, the factors which affect the selection of haulage equipment as well as the relevant modules of the NUmine system, it is now possible to return to the two main objectives.

The first, to demonstrate how the modules can be used to simplify and shorten the process of analysing truck/shovel systems, will become apparent as the benefits of each are pulled together in a single, all-encompassing program. The second, to determine the extent to which mathematically derived solutions can be met in real time by the application of dispatching policies, will be tested by simulating actual operations using the above program.

6.1.1 Selection and Scheduling Problem - General Description.

Flow through the haulage analysis program is closely related to the format of the problem were it to be carried out using conventional, manual methods. The scheme is illustrated in the data flow diagram (Fig. 6.1).

The two main inputs to the system are a digitised plan of the haul routes and the production requirements at each load point. The digitised plan is used as the basis for the formulation of a haul route network and this is covered in Section 6.2. The production requirements, meanwhile, enable preliminary truck selection as described in Section 3.3.2. Loader selection is a complex decision process and is not considered here. However, work encompassing the broader area of excavation equipment (Clarke, 1989) is currently in progress elsewhere in the NUmine system.

Typically, a range of acceptable haulage equipment types would now need to be tested against each other. The next stage is, then, to calculate the time taken for each acceptable truck to complete a haul or return journey over the valid routes in the network. These values, together with the production requirements and the loading times for each item of loading equipment, form input to the linear programming

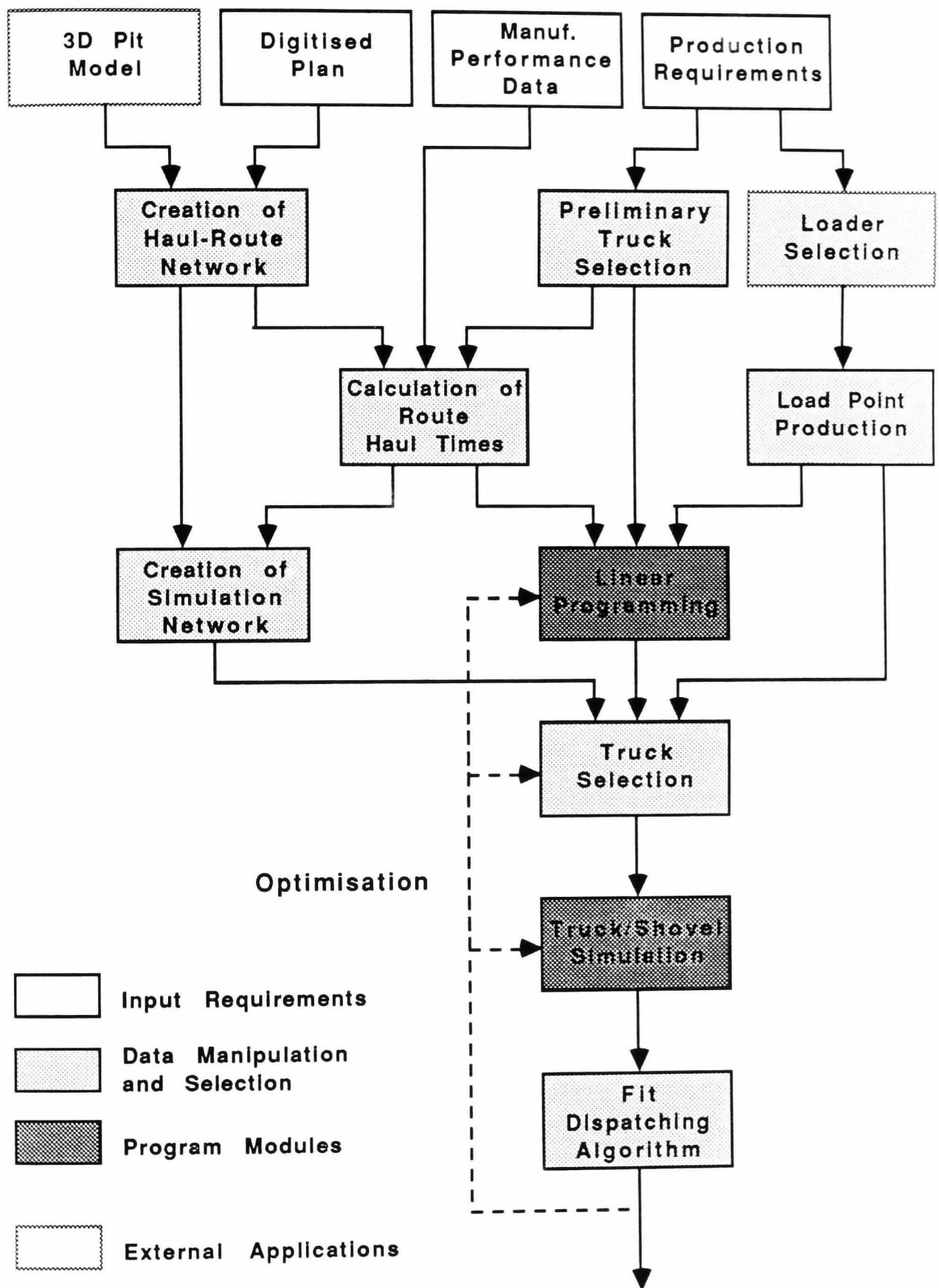


Figure 6.1 Haulage Analysis Flowsheet

module which, as demonstrated in Section 2.7.1 will indicate how many trucks should be allocated to each route in order to 'optimise' synergy. The cycle time calculation and linear programming mechanisms are covered in Sections 6.2 and 6.3 of this chapter respectively.

The next stage would be to carry out cost comparisons in order to establish those possible combinations of truck types which need to be analysed further. These are passed into an automatically derived simulation network for final selection with dispatching taken into consideration. These areas are covered in Section 6.4. Unfortunately, the financial appraisal aspect is also outside the scope of this project though it too is feasible within the NUmine system.

6.1.2 Program Objectives.

It can be seen how the latter stages of the selection process can become extremely repetitive, especially when there are a large number of alternative solutions within an acceptable cost range. It is therefore essential that the elimination of invalid solutions is carried out as early as possible. This can only be achieved if the relevance of the initial, mathematically derived solutions is known.

The main objective of the program is, then, to simplify model development by automisation and by allowing easy access to data and to the relevant processing modules. In this way, the comparisons between a number of mathematical and simulated solutions can be tested efficiently.

6.1.3 Program Flow

For both conformity and flexibility, the user is able to activate processes in any order using the conventional NUmine structure. The options which are available from the program's main pull-down menu are illustrated in Figure 6.2.

- The first group of functions, as in any other NUmine application, relate to file handling. In this case, they are concerned with the access to and the storage of whole projects.
- The second group allow for alteration to the project's component relations. These will be discussed more fully in the following sections.
- The third group are concerned with the optimisation process carried out within the linear programming module.

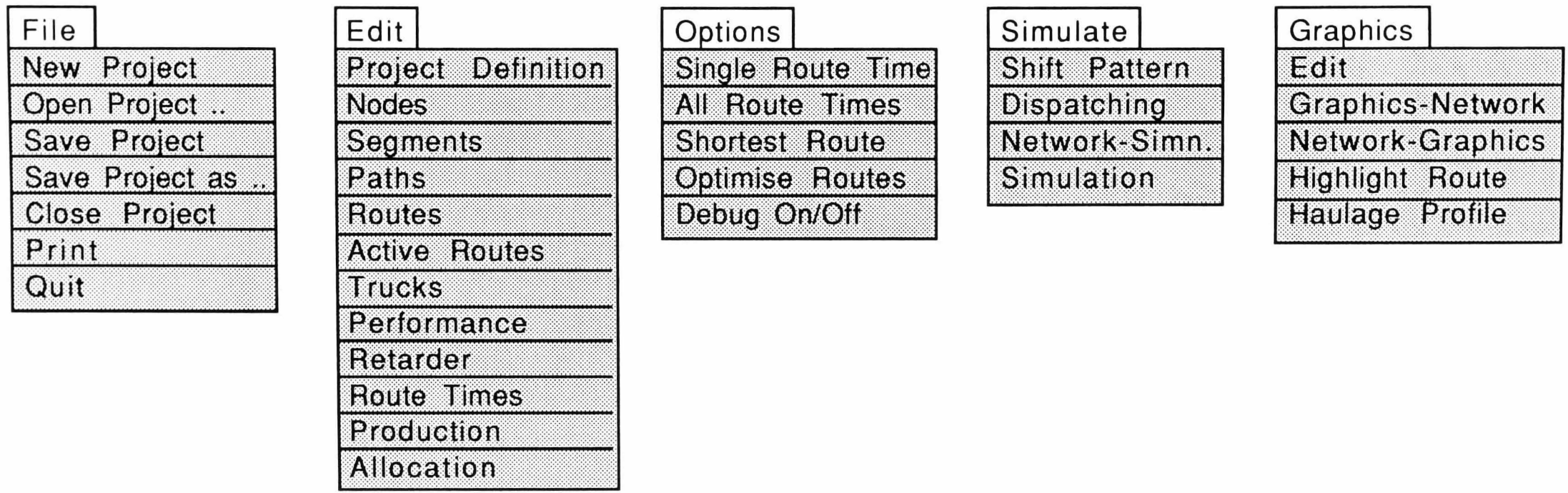


Figure 6.2 Pull-down menu options in the Haulage Analysis Program

- ❑ The fourth are concerned with the configuration of the simulation network.
- ❑ An additional fifth group of functions allow access to the graphics system and carry out the necessary conversions between the graphical and relational formats which the haul route network can assume.

6.1.4 Quarry Example

Throughout Sections 2 to 4 of this chapter, reference will be made to a simplistic network representing a hypothetical quarrying operation. In the example, material is being excavated from 3 loading points and may be hauled to one of two dump points. Productive capacities at each loading point are listed in Figure 6.3. The permanent crusher at the plant has a maximum capacity of 7100 tonnes per hour. The remaining material must be hauled to the secondary 'in-pit' crusher where it can be stockpiled, if necessary, for later crushing and conveyance.

Figure 6.4 shows the quarry's infra-structure overlayed on a plan of the overall operation. The important nodes are represented by labelled markers and are linked together either by individual lines representing segments or by unfilled line polygons representing sub-routes on the lowest recursive level (Section 6.2.2). Trucks are able to haul material in either direction along any of the routes shown.

Production			
Node Name	Capacity (tonnes)	Time (secs)	Queueng
Ld1	3000	45	true
Ld2	5000	60	true
Ld3	3500	45	true
Dp1	7100	15	false
Dp2	4400	15	false

Figure 6.3 Quarry Example - 'Production' Relation

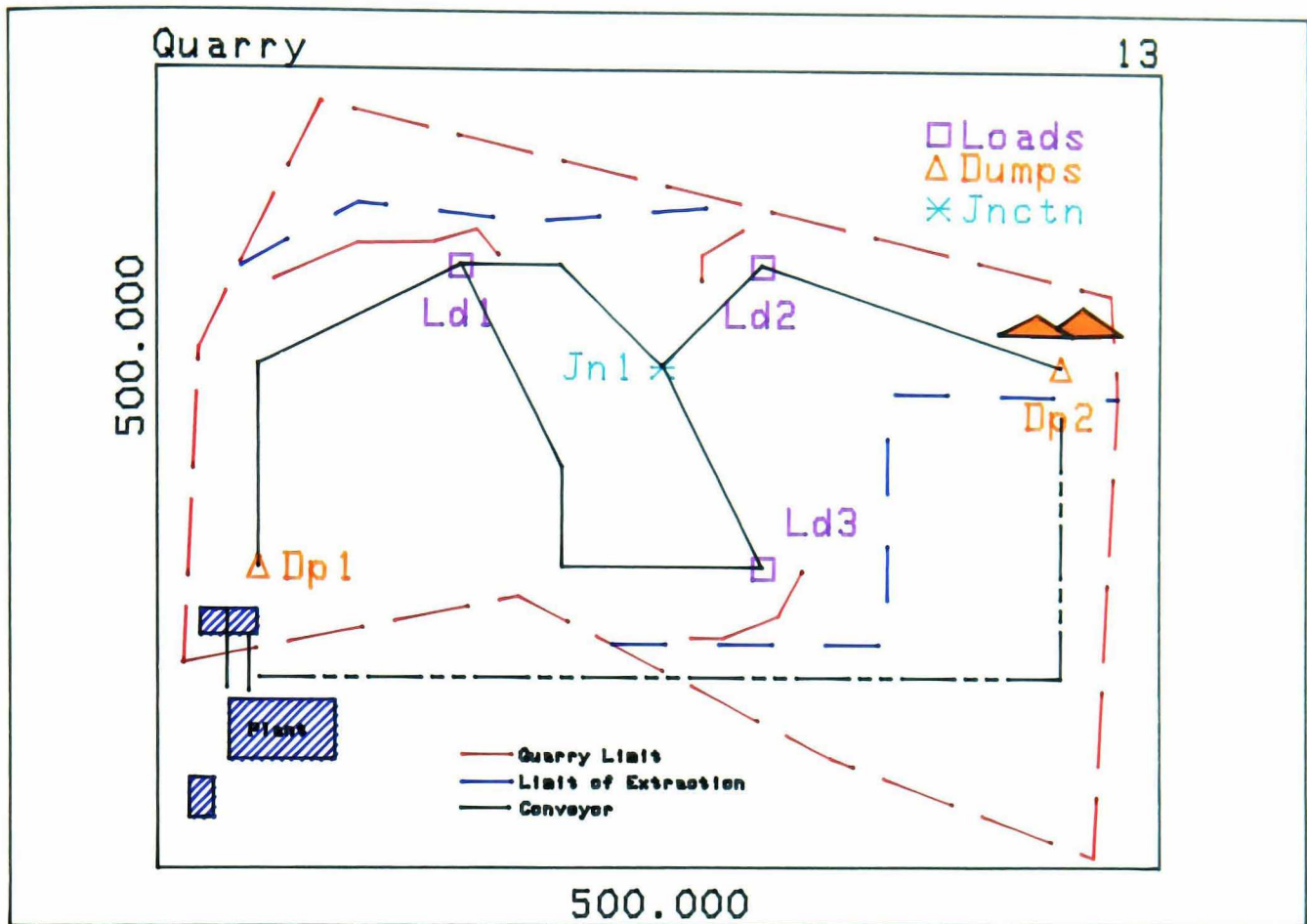


Figure 6.4 Quarry Example - Infrastructure and Road Plan

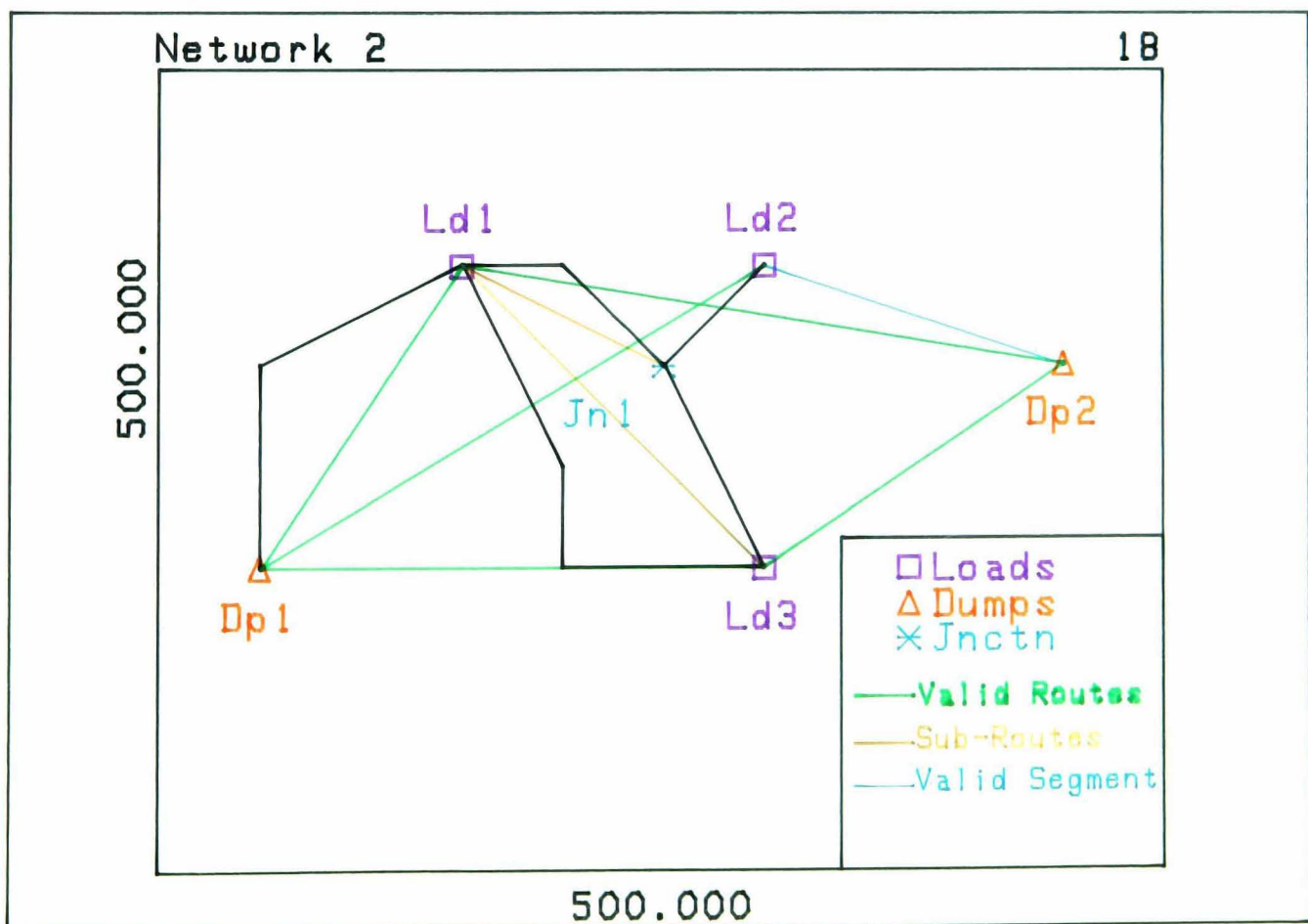


Figure 6.10 Quarry Example - Valid Routes and Sub-Routes

6.2 The Haul Route Network

The haul route network is purely relational in structure. This ensures that data storage is efficient with regard to actual file utilisation. However, this is not the only consideration when attempting to minimise the storage capacity required for what can be a substantial data manipulation problem. It is also important to reduce any unnecessary duplication of existing data structures and consequently the total work load. This is achieved by adopting an overall structure which is slightly different from the one used conventionally, one which is made possible by a) the use of a relational database and b) recursion.

6.2.1 Network Structure

As described in Section 2.2.2, the haul route network consists of a number of interlinked homogeneous road segments each end of which forms a node. Some of the nodes are load or dump points, others are road junctions, the rest simply separate those segments which have different gradients and/or rolling resistances.

Figures 6.5 and 6.6 show the 'nodes' and 'segments' relations for the quarry example. The 'nodes' relation stores the coordinates and types of each node, the 'segments' relation stores the distance, gradient and rolling resistance of each segment. Already, it is apparent how much typing would be required if the data were to be input from the keyboard. There is also a noticeable absence of any information which links the segments to their respective end nodes. The reason for this will become clear later.

The next stage is the definition of the haul routes themselves. Conventionally, this would be carried out by listing, in order, all the nodes through which a route passes. The path characteristics would then be obtained by using the above 'absent' information to find the segments which link each pair of consecutive nodes in the list. Figures 6.7a and 6.7b illustrate this for the definition of the route network in the quarry example. While this method is generally acceptable for the analysis of small operations and those with few valid routes, as soon as two routes covering the same stretch of road are defined, a degree of duplication is experienced.

The NUmine alternative is to split each path down into sub-routes which represent those stretches of road utilised by more than one route and to make the system recursive. i.e. the sub-routes themselves may consist of a number of smaller sub-routes.

Nodes				
Node Name	X	Y	Z	Type
Ld1	300000	600000	141500	Load
Ld2	600000	600000	147000	Load
Ld3	600000	300000	145500	Load
Dp1	100000	300000	156000	Dump
Dp2	900000	300000	153000	Dump
Jn1	500000	500000	143000	Junction
7	100000	500000	150500	
8	400000	300000	149500	
9	400000	400000	141500	
10	400000	600000	142500	

Figure 6.5 Quarry Example - 'Nodes' Relation

Segments			
Route Name	Distance	Gradient	Rol. Res
	mm	Grads	
9	223607	4.02	2.00
10	200000	2.75	2.00
13	200000	2.00	3.00
14	100000	-8.00	3.00
15	223607	0.00	2.00
17	100000	1.00	4.00
18	141421	0.35	4.00
Ld2-Dp2	316228	1.90	4.00
19	141421	-2.83	2.00
20	223607	-1.12	2.00

Figure 6.6 Quarry Example - 'Segments' Relation

Node List						
Route Name	Node1	Node2	Node3	Node4	Node5	Node6
Ld1-Dp1	Ld1	7	Dp1			
Ld1-Dp2	Ld1	10	Jn1	Ld2	Dp2	
Ld2-Dp1	Ld2	Jn1	10	Ld1	7	Dp1
Dp2-Dp2	Ld2	Dp2				
Ld3-Dp1	Ld3	8	9	Ld1	7	Dp1
Ld3-Dp2	Ld3	Jn1	Ld2	Dp2		

Figure 6.7a Node List

Segment Links		
Route Name	Start Node	Stop Node
9	Ld1	7
10	7	Dp1
13	Ld3	8
14	8	9
15	9	Ld1
17	Ld1	10
18	10	Jn1
Ld2-Dp2	Ld2	Dp2
19	Ld2	Jn1
20	Ld3	Jn1

Figure 6.7b Segment Links

(For Convenience, all the above identifiers are equivalent to those derived automatically using the NUmine system)

Figure 6.7 Quarry Example - Conventional Methods

Referring again to the quarry example, sub-routes would be established between nodes DP1 and Ld1, nodes Ld1 and Ld3 and nodes Ld1 and Jn1. The 'routes' and 'paths' relations are used to store this information (Figs. 6.8 & 6.9). There are no hard and fast rules about the choice of recursive path definition, hence no strict value for the potential saving of a particular layout can be calculated. However, in the Coal Mountain case study which follows, the 178 paths and 122 routes add up to a total of 300 records. This compares favourably with the 568 path entries required to fully define the 24 valid routes using conventional methods. This also assumes that the conventional technique is able to recognise a defined path in both directions. If this is not the case, the storage requirement for the latter would be doubled. Within the NUmine format, only the valid route names, together with their start and end nodes need be redefined for two-directional recognition.

Routes			
Route Name	Start Node	Stop Node	Type
9	Ld1	7	Segment2
10	7	Dp1	Segment2
13	Ld3	8	Segment2
14	8	9	Segment2
15	9	Ld1	Segment2
17	Ld1	10	Segment2
18	10	Jn1	Segment2
Ld1-Jn1	Ld1	Jn1	Path2
Ld1-Dp1	Ld1	Dp1	Path2
Ld3-Dp1	Ld3	Ld1	Path2
Ld2-Dp2	Ld2	Dp2	Segment2
19	Ld2	Jn1	Segment2
20	Ld3	Jn1	Segment2
Ld1-Dp2	Ld1	Dp2	Path2
Ld2-Dp1	Ld2	Dp1	Path2
Ld3-Ld1	Ld3	Dp1	Path2
Ld3-Dp2	Ld3	Dp2	Path2

Figure 6.8 Quarry Example - 'Routes' Relation

Paths		
Route Name	Order	Sub-Route
Ld1-Dp1	1	9
Ld1-Dp1	2	10
Ld3-Ld1	1	13
Ld3-Ld1	2	14
Ld3-Ld1	3	15
Ld1-Jn1	1	17
Ld1-Jn1	2	18
Ld1-Dp2	1	Ld1-Jn1
Ld1-Dp2	2	19
Ld1-Dp2	3	Ld2-Dp2
Ld2-Dp1	1	19
Ld2-Dp1	2	Ld1-Jn1
Ld2-Dp1	3	Ld1-Dp1
Ld3-Dp1	1	Ld3-Dp1
Ld3-Dp1	2	Ld1-Dp1
Ld3-Dp2	1	20
Ld3-Dp2	2	19
Ld3-Dp2	3	Ld2-Dp2

Figure 6.9 Quarry Example - 'Paths' Relation

6.2.2 Graphical Formulation

One of the disadvantages of the above structure is that it is difficult to define by manual numeric entry. Though the overall amount of input is considerably reduced, the degree of cross-referencing between the relations makes it more complicated. Graphical formulation of the network is, on the other hand, a straight forward process made no more difficult by the new structure.

The simplest input method is to digitise items as markers or symbols with the NUgraph 'network' option switched on. In this way, each time a marker or symbol is entered, a 3-dimensional point, to which the marker is automatically linked, is also created. In Figure 6.10, the polygons linking the above nodes were entered digitally by marking points along a haul route using the usual line-polygon entry mechanism, i.e. a '7' to indicate the start of the polygon followed by '3' to indicate line input and thereafter '2's or '3's (No type change can be made after the second point).

The rest of the picture manipulation can be carried out using the mouse and keyboard. First, a line must be associated with each sub-route polygon linking its start node to its end node directly. The convention here is to use yellow for pure sub-routes and green for valid haul-routes, i.e. those routes which link load points to dump points and vice-versa. Where the sub-route is to be used in one direction only, a broken line is used.

To link a number of segments and sub-routes together into a new (sub-) route, the polygon definition will follow a path of existing lines. Here, the initial '7' will automatically activate one of the lines joined to the first point. If this represents the first sub-route/segment required, a '3' is entered within the area defined by the line's active boundary, the cursor moves to its other end point and a new line is activated. Otherwise, the correct line is activated by typing a '1' somewhere along its length.

After making sure that each marker and polygon has been labelled and named correctly, the picture can be converted into a relational format for further analysis.

6.2.3 Integration with a 3-dimensional Pit Model

In any 2-dimensional digitised system, the third coordinate value, in this case the elevation or z-coordinate of each node, must be typed in manually. While this facility is available within NUgraph, the time spent activating points and entering the correct values for a large number of nodes would be considerable. Instead, the integrated nature of the NUmine system makes it possible to obtain the elevation of each node from a three dimensional model of the mine's surface.

Whichever method is used, the node elevations allow for the automatic calculation of slope distance and grade. The final field in the segments relation, rolling resistance is, therefore, the only one throughout the whole model which will always require manual input.

6.2.4 Additional Graphics Features

Two additional graphics features exist within the truck/shovel program: the first highlights a haul-route in order to a) tell the user that the route has been correctly defined and b) help him select one from a set of alternative routes; the second is also concerned with the latter - it produces an exaggerated profile of the haul route which is automatically displayed on the screen next to the network (Plate 6.1). A direct comparison can be achieved by plotting each alternative on the same set of axes.

Both of these functions require temporary reconstitution of the complete lists of nodes and segments which define the route. The NUmine network structure demands that this process is also a recursive one. The following flowsheets illustrate this (Figs. 6.11 a & b).

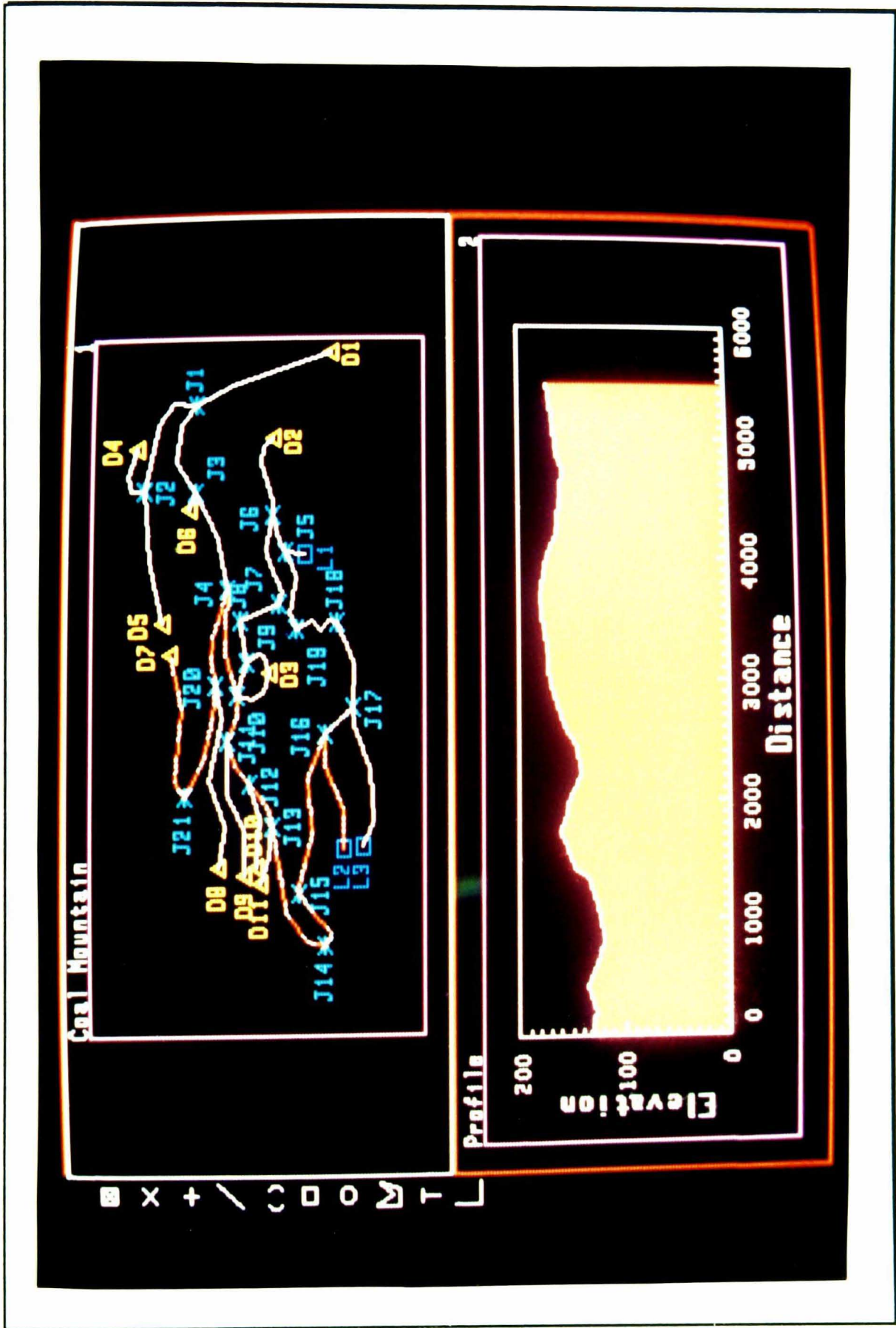


Plate 6.1 Highlighted Route and Profile

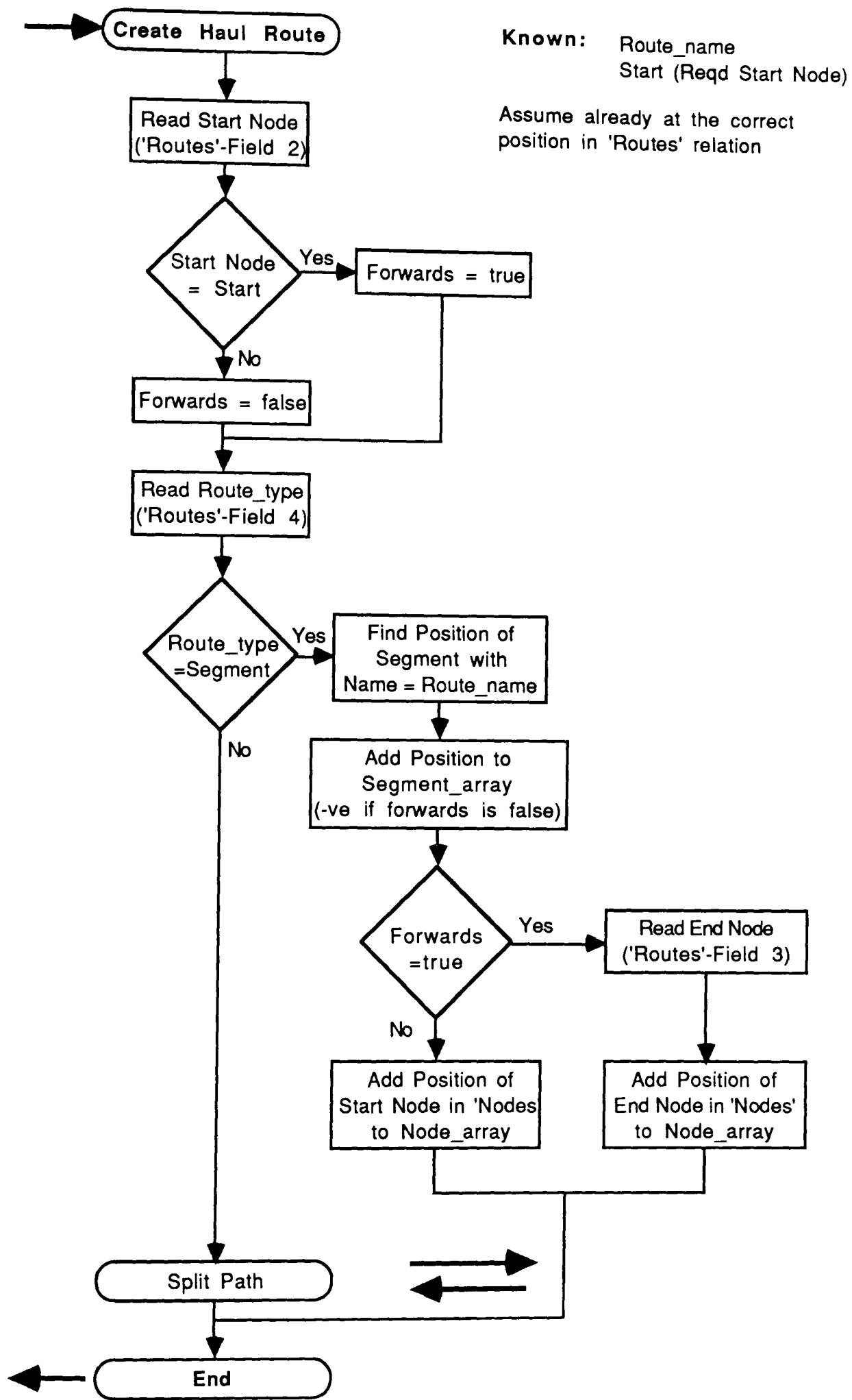


Figure 6.11a Create Haul Route Flowsheet

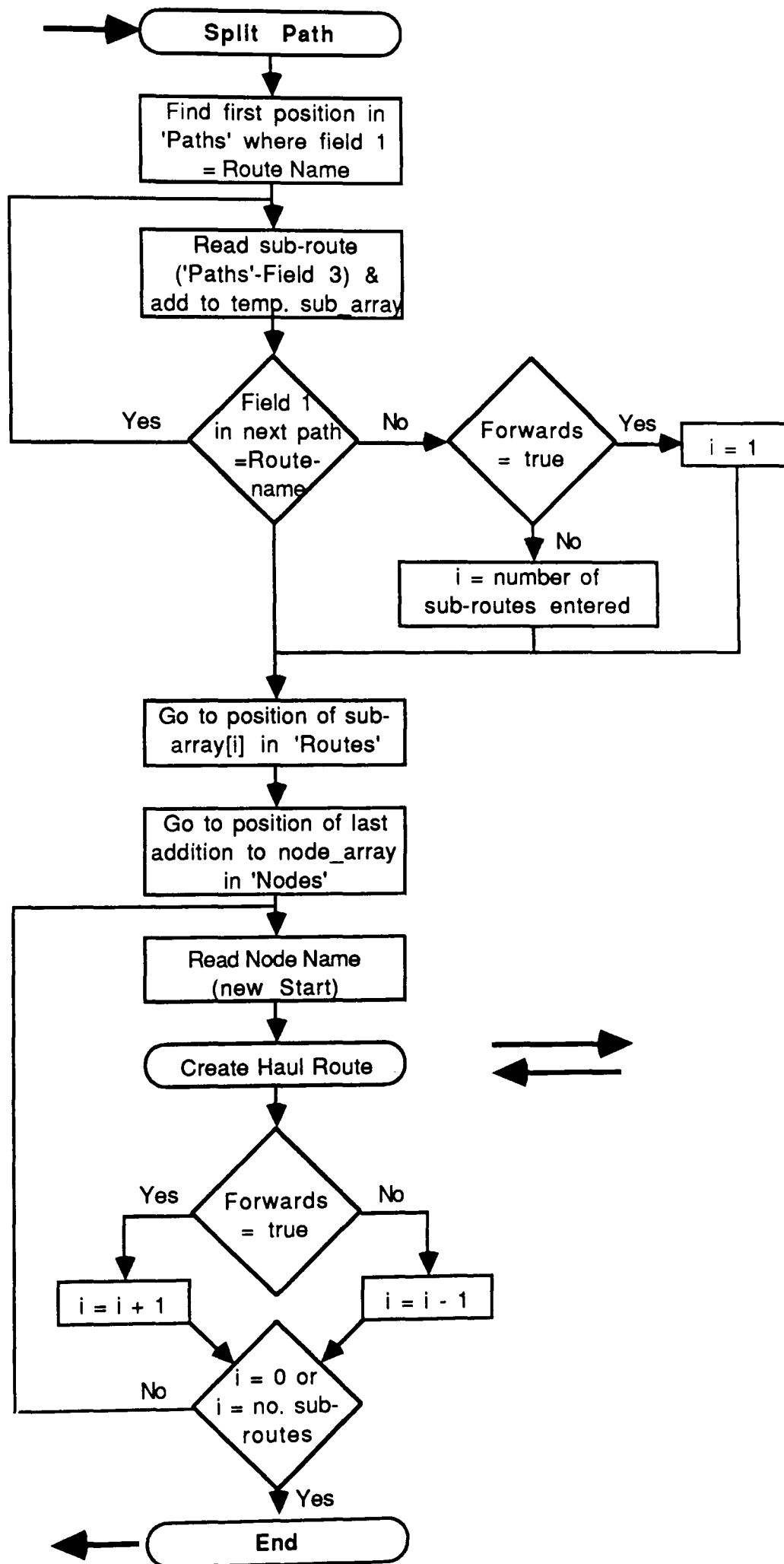


Figure 6.11b Split Path Flowsheet

6.3 Truck Selection by Linear Programming

It can be assumed that, by this stage, preliminary truck selection of the type described in Section 3.3.2 will have been completed. The planning engineer will already have some idea of the range of truck models suitable for the mine in question and will have compiled the performance and retarder curves for these into the required format. The next objective would be to establish haul times for each truck model along each valid route.

6.3.1 Route Haul Times

The main requirements here are the distance, gradient and rolling resistance of each segment which makes up the route. These are established by reconfiguring the segment list using exactly the same routines as described above, by locating the position of each in the segments relation and then reading their characteristics into a 'haul' array.

The haul time is calculated in a similar manner to the way described in Section 2.2 except that the necessary retarding distances for each downhill (sharply curved) road are calculated before the performance simulation begins. To illustrate this, consider the route taken from Dp1-Ld3 in the quarry example. The route profile is predominantly downhill (Fig. 6.12) which means that a large proportion of the array is subject to a speed limitation some way below the capability of a truck. In addition, the uphill stretch between nodes 9 and 8 is limiting with regards available truck power.

A braking period must be allowed wherever it is possible that the truck, on entry to a following segment, may be travelling too fast. Typical areas are immediately preceding downhill segments and in the final stretch during which the truck must be brought to rest. A mean deceleration rate of 1.5 ms^{-2} for downhill and 2 ms^{-2} for uphill stretches is assumed.

Next, the acceleration time must be calculated. As mentioned in Section 2.7, this is achieved dynamically by assuming a constant acceleration over the period of time that the truck remains in a particular gear (Equations 2.5). The distance covered is also calculated based on the equations of motion where v and u are taken as the maximum speed for the gear and that of the previous gear respectively. If the speed for the gear exceeds the segment speed limit, the acceleration period is reduced accordingly. (Fig 6.13 (Segment Ld1-9))

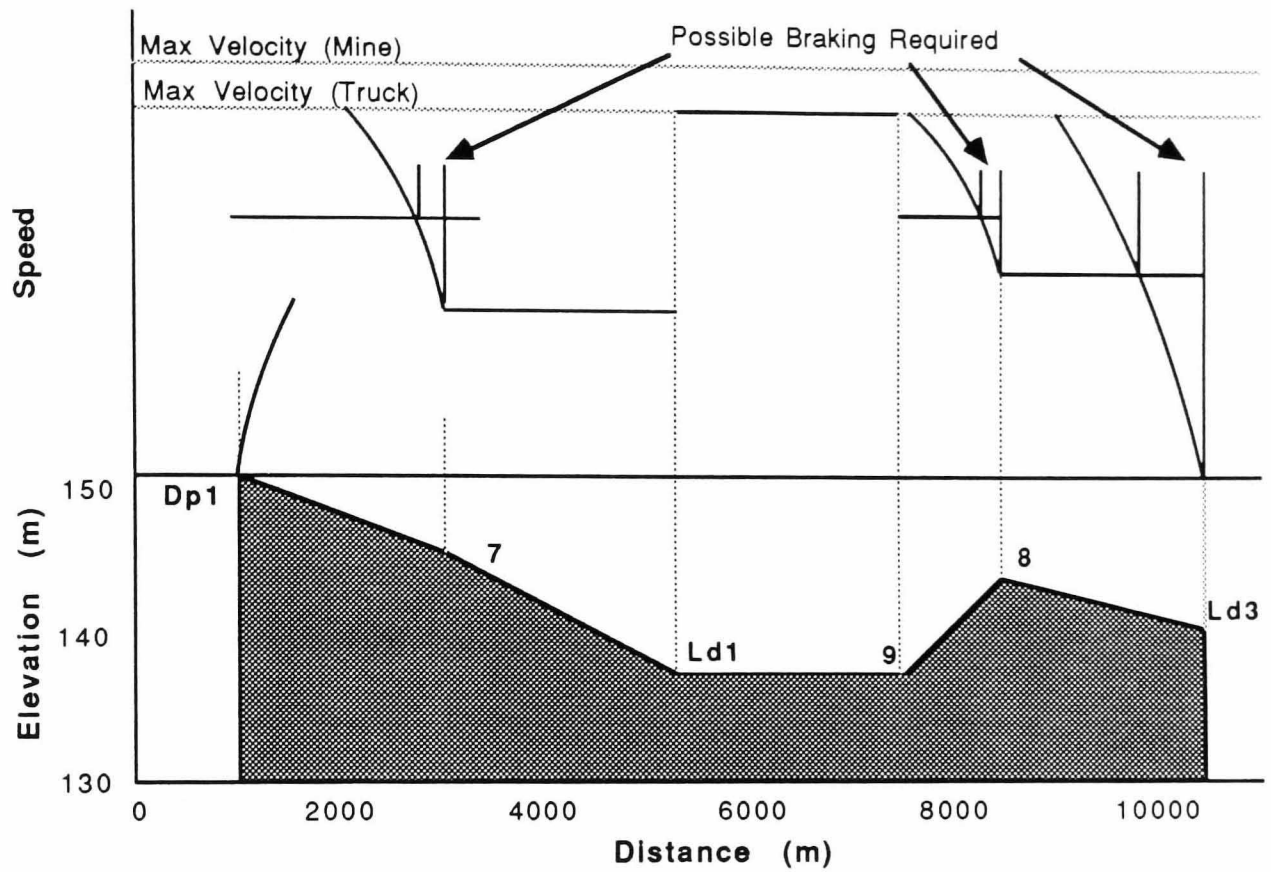


Figure 6.12 Route Profile and preliminary speed considerations for Dp1-Ld3

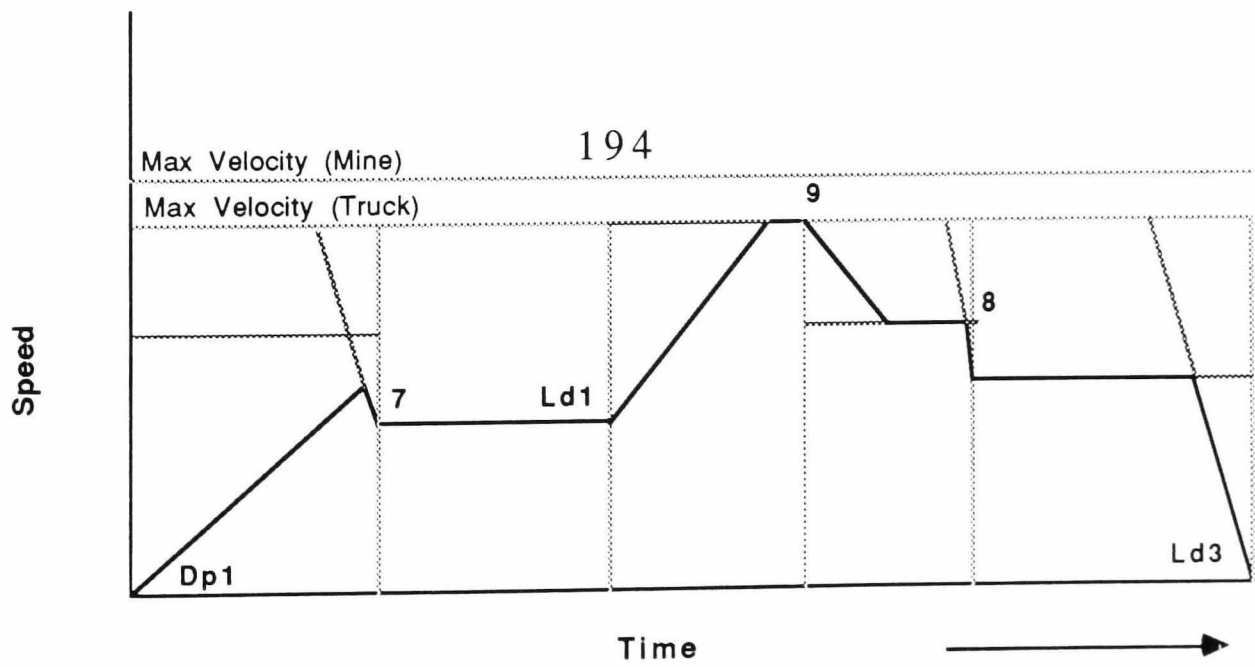


Figure 6.13 Speed versus Time for Dp1-Ld3

However, if, having calculated the distance covered in a particular gear, the total distance (accelerating and braking) is found to exceed that of the whole segment (Segment Dp1-7), the maximum velocity obtained must be interpolated in order to recalculate the time and distance spent in each state (Fig. 6.14(Pt. a)). The same applies for accelerating alone if no braking is required (Pt. b).

In segment 14 (9-8), the impact of the steep upward slope causes the truck to slow down. This is evaluated in the same way as above but by swapping the values used for v and u .

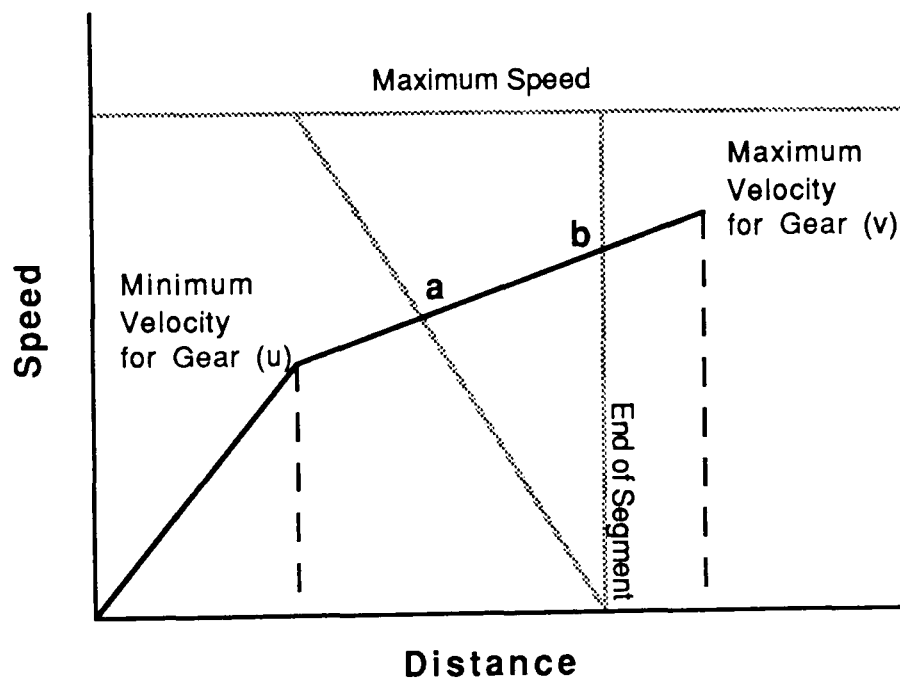


Figure 6.14 Interpolation of velocities at critical points

6.3.2 Validation of the Haul-Time Calculation Routine

In the absence of any real data upon which to base validation of the haul-time calculation routine, comparisons were made between the values achieved for a number of trucks traversing a pre-defined route using the NUmine system and those achieved for the same trucks over an identical route using existing software packages. This information was obtained from work carried out by Srajer (1987) which formed the basis of his PhD thesis. The haul route characteristics listed in Table 6.1 were inserted into a segments relation and read directly.

Figures 6.15 and 6.16 show the times calculated for the haul and return sections of the test route respectively using a) Srajer's model and b) the NUmine approach. It can be immediately seen that a significant degree of correspondance has been achieved for all the trucks tested especially in the empty return phase. The results obtained for two Wabco models have been plotted against the only information available for a Wabco truck the identity of which is ambiguous in the source text.

The reason for the varying degrees of discrepancy between the two sets of haul time results was initially diagnosed as being due to the assumption that each truck had been filled to its payload capacity. In fact, a comparison of the overall times achieved for each truck (Figs. 6.17 a & b) show that the NUmine system produces results which are, on average, closer to the 'safe' estimates provided by the various truck manufacturers than the model data. In any case, for the purposes of illustrating the principle of haul time calculation within the context of an integrated planning package, these results are close enough to suggest that only site-specific modifications need be made.

6.3.3 Linear Programming Formulation

NUmine's linear programming module is based on the simplex method (Press *et al*, 1986). As described in Chapter 2, the problem of minimising flow rates along each haul route are limited by the constraints of continuity and production. These, together with the objective function itself, must be configured into a tabular format acceptable to the LP module as follows:-

□ Row 1

This contains the objective function i.e. the time taken for a truck to travel along each path listed in order so that each column represents a path.

Route Name	Distance m	Gradient Grads	Rol. Res	
HAUL	1	320	-3.0	3.5
	2	80	-3.0	3.5
	3	70	-0.6	2.0
	4	450	-0.6	2.0
	5	305	3.9	2.0
	6	115	-0.2	2.0
	7	465	-0.2	2.0
	8	515	0.2	2.0
	9	140	0.2	2.0
	10	85	0.1	2.0
	11	350	0.1	2.0
	12	255	0.0	2.0
	13	245	0.0	2.0
	14	400	0.0	2.0
	15	950	0.1	2.0
	16	100	-0.9	2.0
	17	470	-0.9	2.0
	18	150	-0.9	2.0
	19	520	-0.9	2.0
	20	80	-0.9	2.0
	21	125	0.2	2.7
	22	40	0.2	2.7
	23	15	0.2	2.7
	24	80	-3.0	2.7
RETURN	25	55	2.6	2.7
	26	60	2.6	2.7
	27	20	-0.6	2.7
	28	80	-0.6	2.7
	29	15	0.2	2.7
	30	40	0.2	2.7
	31	125	0.2	2.7
	32	80	-0.9	2.0
	33	520	-0.9	2.0
	34	150	-0.9	2.0
	35	470	-0.9	2.0
	36	100	-0.9	2.0
	37	950	0.1	2.0
	38	400	0.0	2.0
	39	245	0.0	2.0
	40	255	0.0	2.0
	41	350	0.1	2.0
	42	85	0.1	2.0
	43	140	0.2	2.0
	44	515	0.2	2.0
	45	465	-0.2	2.0
	46	115	-0.2	2.0
	47	60	0.5	3.5
	48	500	0.5	3.5
	49	20	0.5	3.5
	50	240	0.0	5.0
	51	110	0.0	5.0
	52	190	0.0	5.0

Table 6.1 Test Haul Route Characteristics

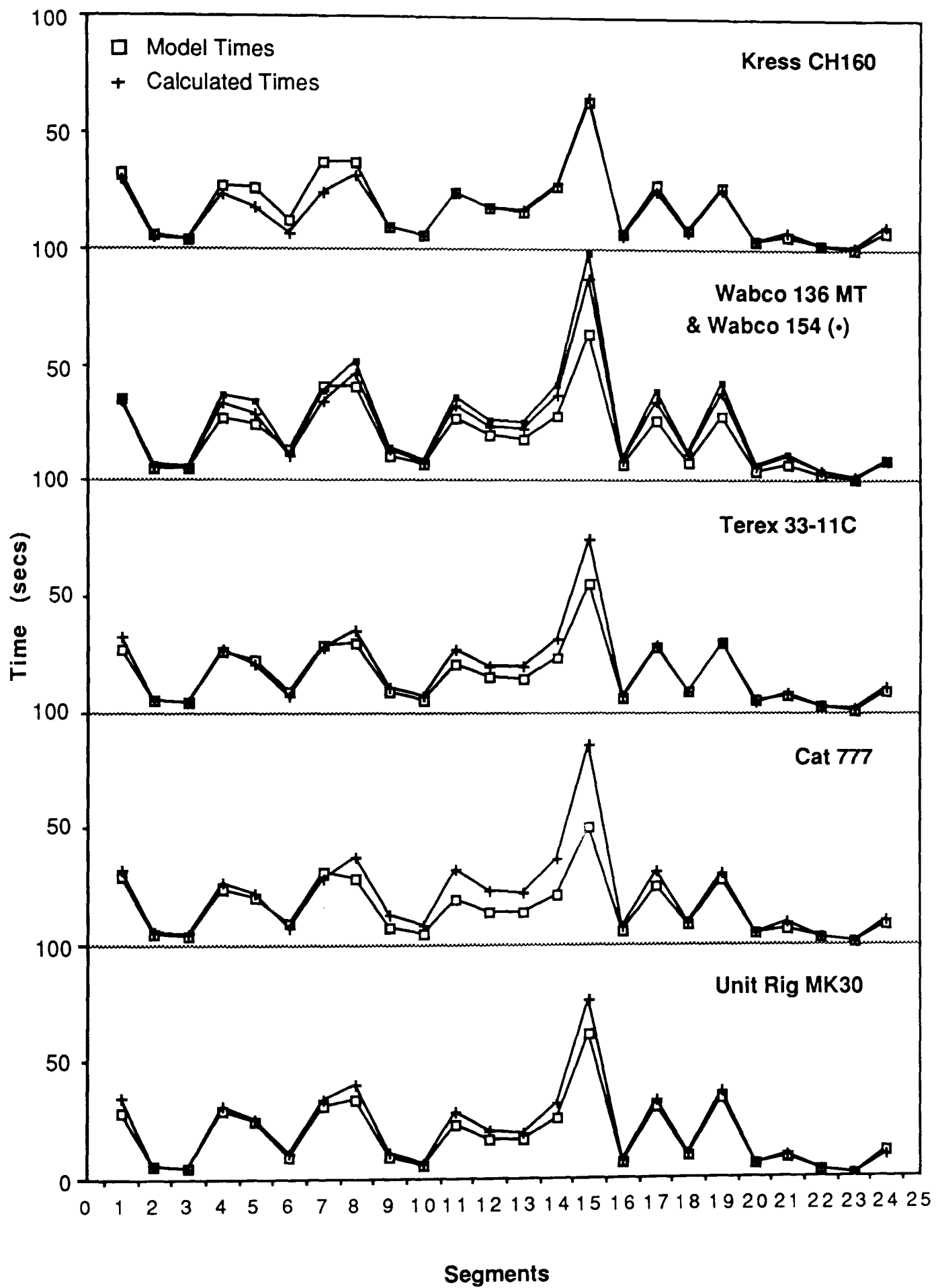


Figure 6.15 Segment Times for 6 different trucks hauling material along the test route

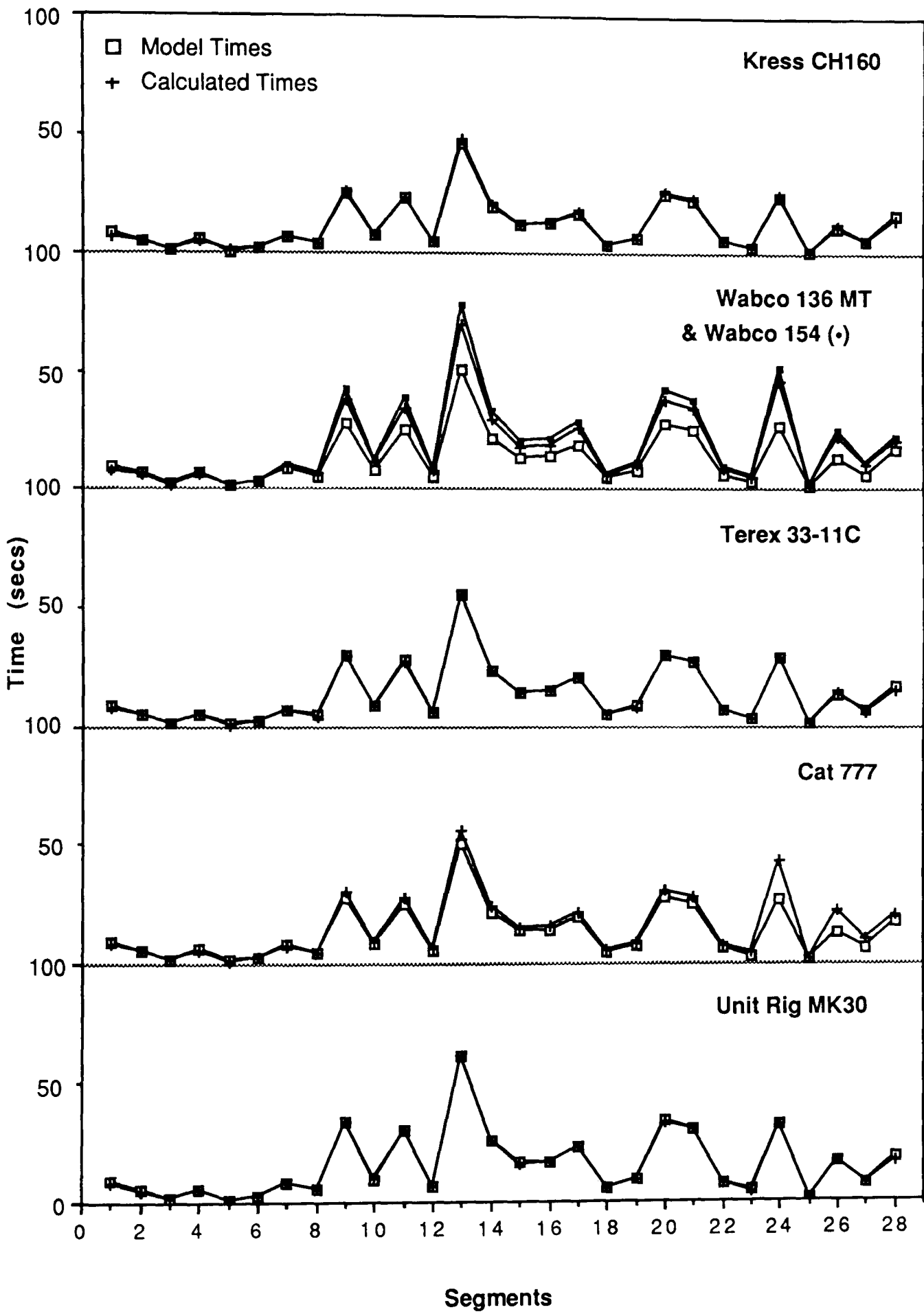


Figure 6.16 Segment Times for 6 different trucks returning empty along the test route

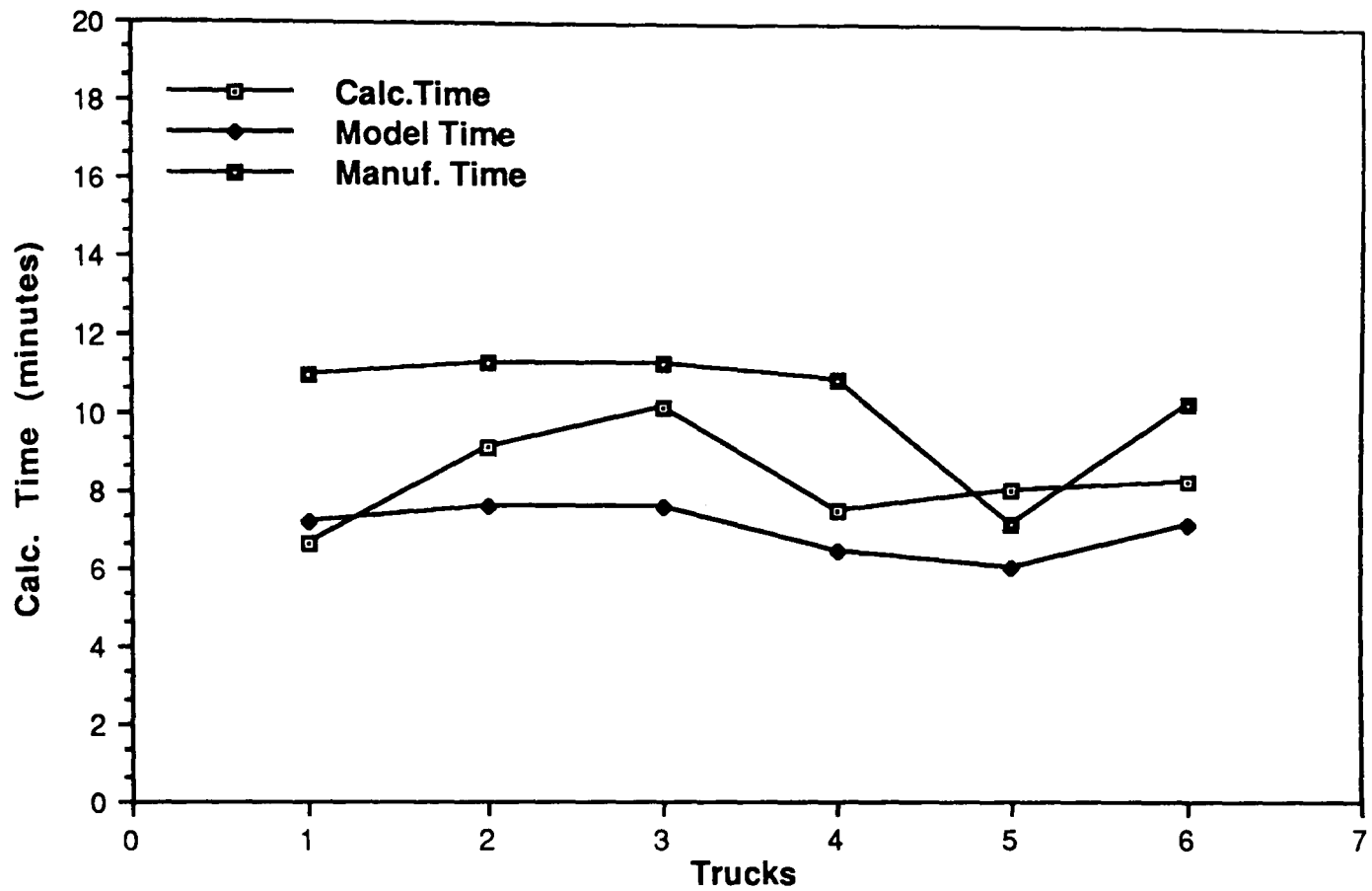


Figure 6.17a Comparison of Calculated times, Model times and Manufacturer Specified times for Hauling

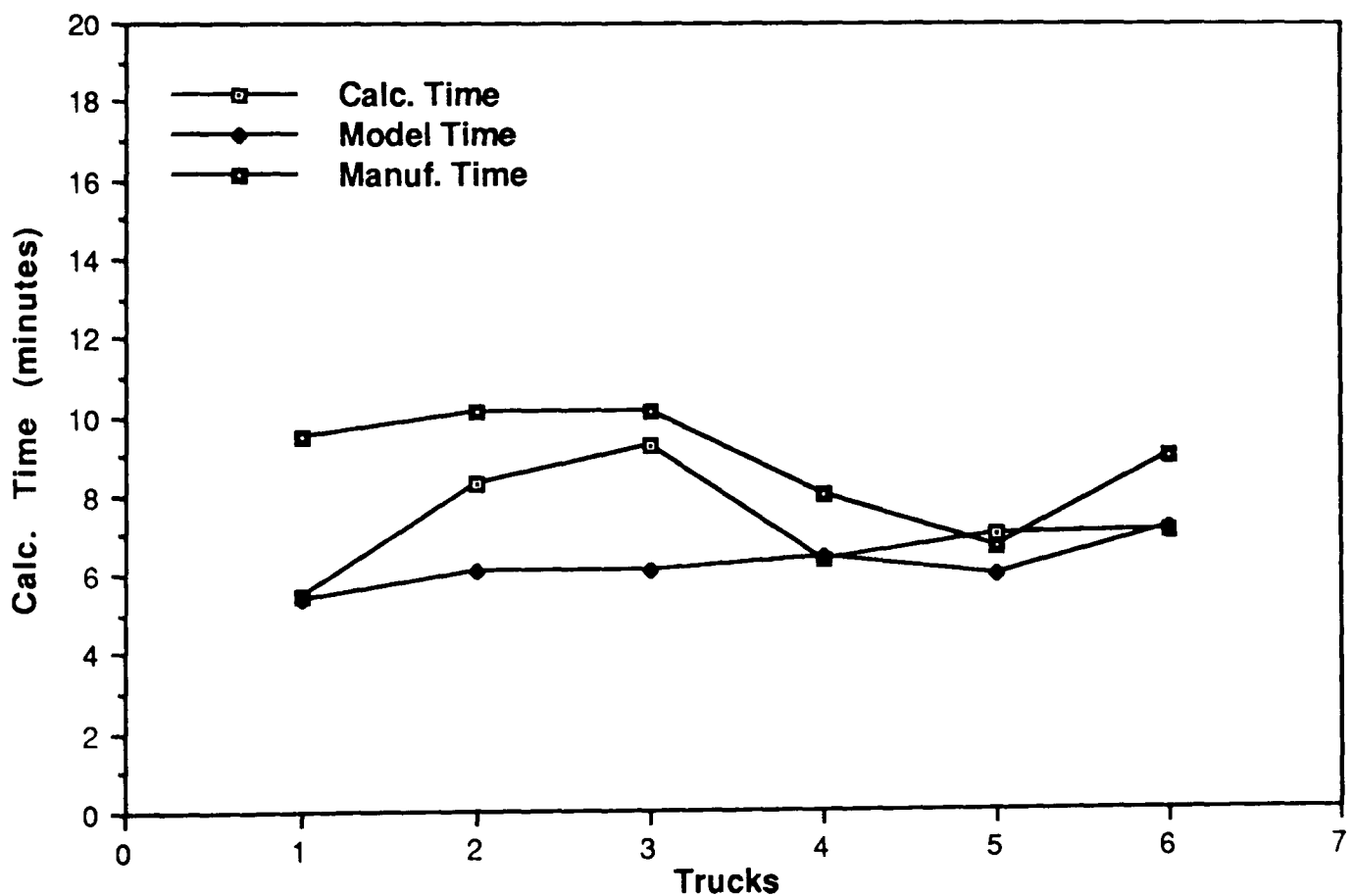


Figure 6.17b Comparison of Calculated times, Model times and Manufacturer Specified times for Returning Empty

□ Rows 2 - NumNodes+1

These contain the continuity information for each load and dump node in the network - each row represents a node.

For each row: a 1 is placed in each column representing a path flowing into the node;
a -1 is placed in each column representing a path flowing out of the node;
a 0 is placed in the objective column.

□ Rows NumNodes+2 to NumNodes+2+NumProductiveNodes

These contain the productivity information for each node requiring a certain input/output.

For each row: a 1 is placed in each column representing a path flowing into the node;
the required production is placed in the objective column.

The production required at each node must be typed into the production relation. If no production is specified for a particular node, the node will be omitted from the productivity rows of the input tableaux and its value will be calculated according to the 'optimisation' procedure. This is generally the case for the dump points.

Care must be taken to ensure that the total output from a system or sub-system for which all the productivities have been declared, equates to its total input. Otherwise, the simplex routines are unable to produce a solution.

6.3.4 Two Types of Truck

The above method produces results analogous to those derived from Equation 2.12 in Chapter 2. The truck allocations (per hour) for each path are determined by dividing the required flow rate by the payload of the truck under examination. However, this method fails to take into account the situation where a combination of two or more truck types are either already in use or are considered as a viable alternative. Here, one column must be set aside for calculating the flow rate of each type of truck along each valid path. Hence, the examination of two truck types will require twice as many columns.

In general, output from the linear programming module will be biased towards the quickest trucks - no account is taken at present of the capital and operating costs of each truck type. Therefore, if no limit is placed on the number of units of each type which can be used, the quickest trucks will be allocated. Placing a value in the second field of the truck relation (Fig 6.18) counters this. It adds one extra row per limited truck to the input tableaux to ensure that the potential productivity of these are fulfilled before the remaining trucks are considered (see Section 6.5).

For each additional row (each row represents one limited truck type), the travel times for each haul route are placed in the appropriate columns and the total potential allocation is placed in the objective column. This is based on the following calculation:

The potential productivity for n_j trucks in one hour is n_j truck hours

$$\Rightarrow P_{rj} = a_{mj} \times t_{mj} \quad \dots (6.1)$$

where: a_{mj} = allocation of trucks of type m to path j
 t_{mj} = travel time of truck of type m along path j
 P_{rj} = productivity required for allocation a_{mj}

$$\Rightarrow P_m = \sum_{i=1}^{n_j} a_m \times t_{mj} \leq P_{pn} \quad \dots (6.2)$$

where P_m = total productivity required
 n_j = total number of paths
 P_{pn} = potential productivity

and, for potential productivity to be met,

$$\Rightarrow P_{pn} = \sum_{i=1}^{n_j} a_m \times t_{mj} \quad \dots (6.3)$$

In the input tableaux, each side of the above equation is divided by the truck payload to conform with other entries.

Trucks			
Truck Type	Number	Weight	Payload
Wabco 136MT	2	82.40	136.00
Terex 33-11C		59.70	77.00

Figure 6.18 A 'Trucks' Relation

6.4 Simulation Network Formulation

The examples described in Chapter 4 illustrate how diverse three simulation networks created for the analysis of similar problems can be. Different node types may be introduced to examine the effect of, say, breakdowns, maintenance schedules, dozer-assisted loading etc. For this reason, interaction between the truck/shovel program and the simulation module is far less rigid than that which links the program to the LP module previously described. In fact, much of the network formulation is still carried out within the NUsim application itself.

Although graphical editing helps a great deal, a substantial amount of the network creation involves typing in probability function and assignment statements and this takes time. A number of labour saving routines were developed to speed up the process.

6.4.1 Converting the Route Network into a Simulation Network

This routine produces a basic network which can be modified within the simulation module as required.

At each load point, the following set of nodes are entered:

- ❑ A Queue with name '**Queue L_n** ';
- ❑ An Activity with name '**Load L_n** '
& duration as a function of the required loading time;
- ❑ An Assignment which records the total loaded at '**Load L_n** ',
the total loaded altogether
and sets the truck load to a random number
approximately equal to its payload,

At each mineral/coal feed dump:

- ❑ A Queue with name '**Queue D_n** ';
- ❑ An Activity with name '**Dump D_n** '
& duration as a function of the required dumping times;
- ❑ An Assignment which records the total amount dumped,
the number of completed trips
and resets the current truck load to zero.

At each waste dump:

- An Activity with name '**Dump D_n** '
& duration as a function of the required dumping time;
- An Assignment which records the same as for the dumps above

Between each load assignment and the queue associated with its allocated dumping station:

- n_t activities with names '**Haul $L_n t$** ',
durations as functions of their respective haul times
& proborconds as '**atrib[2] = t**';

Between each dump assignment and the queue associated with each loading point:

- n_t activities with names '**Return $L_n t$** ',
durations as functions of their respective return times
& proborconds as '**(atrib[6] = n) and (atrib[2] = t)**';

In the above statements,

n = node index,

t = truck index,

n_t = number of trucks.

atrib[2] refers to the truck type

atrib[6] refers to the route selected

In addition, dispatching points may be automatically created by inserting a continue node at a specified time after the dump assignments (The return times are reduced accordingly).

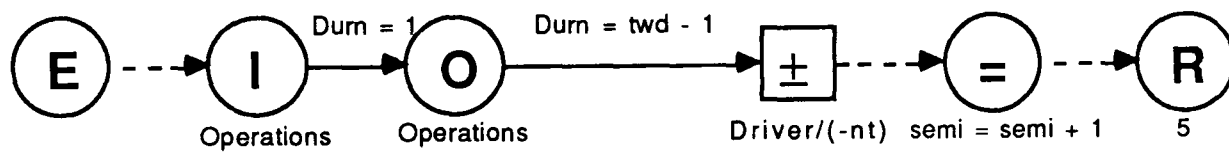
Before the conversion process can begin, the user must specify whether the analysis is concerned with waste haulage or otherwise. He can also specify the distributions round which each load, dump, haul or return times vary.

6.4.2 Shift Patterns

One factor which has a substantial affect on the productive capacity of an operating mine is the daily work regime. A subsidiary shift pattern network can be built in by specifying:

- ❑ the time between driver arrivals (random) - t_{da} ,
- ❑ the time in each shift when work will start to wind down prior to a break - t_{wd} ,
- ❑ the time when work is to recommence - t_{wr} ,
- ❑ the number of trucks in the 'entities' relation - n_t .

Throughout the subsequent case studies, simulation runs of 1 day have been used. The following network simulates a two shift day based on the above parameters (Fig. 6.19). 'Wait' and 'Free' nodes are inserted at appropriate places in the main network to simulate driver availability.



Time Between Entries
 $= 210 + 60 \times$
 $(semi \text{ mod } 2 = 0)$
 (example of t_{wr})

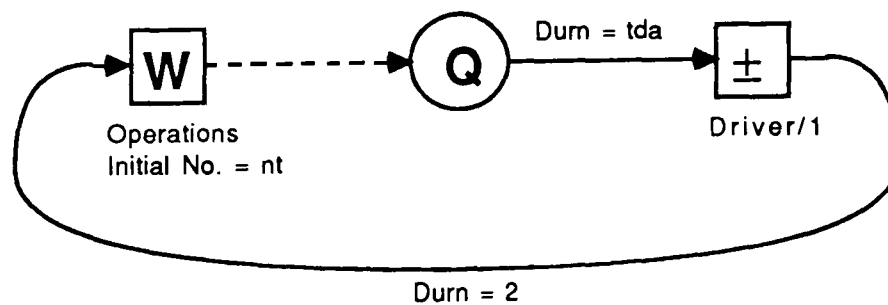


Figure 6.19 Operating Schedule for a two-shift day

6.4.3 Dispatching Algorithms

The dispatching policies which can be automatically included are as follows:

❑ Fixed Allocation

Here, attribute 6 in each truck entity is defined as the variable 'route' and assigned a constant value from within the 'entities' relation. This reference will indicate the truck's destination on each occasion the truck reaches a dispatch node.

❑ Maximise Shovels

This policy is stored as a user-defined function. The loading activity which has been idle the longest will be returned unless all the loading points are busy in which case, the queue and all the return activities leading to the load point are interrogated to find the shovel which is expected to become available soonest. The following algorithm illustrates this:-

For each Load Activity

 If activity is currently loading:

 set expected_time to end of event time for the truck being loaded
 + mean time for loading x number of trucks queueing

 else set expected_time to current time.

 Find the first event which involves the completion of a return activity leading to the load point.

 If the scheduled event completion (arrival) time exceeds expected_time:

 set expected-time to the scheduled event completion (arrival) time.

 Add to expected_time the mean loading time for the truck.

 Find the next similar event and repeat the previous two instructions.

 Compare the expected_time for each load activity and select the minimum for dispatch allocation.

❑ Maximise Trucks

The second user-defined policy also uses the above algorithm but the mean haul time for each possible route is added to the total expected waiting time to determine where the truck is likely to be loaded soonest.

❑ Blending Schedule

Here, the function returns a reference to the loading activity which is furthest behind a specified target value taking into account the total payload of the trucks en route. The target is stored in the 'Variables' relation as a constant with the same name as the load point to which it refers.

❑ Match Factor

This user-defined function calculates the total number of trucks being loaded, queueing or on the way to the loading activity and compares the current match-factor for each cycle by multiplying by the ratio of loader cycle time/hauler cycle time which is also stored as a 'target' variable.

The above policies all involve conditional branching from the dispatching point. The maximise truck policy, for example, would require each 'Return' activity to be given a 'ufn(2) = 1' type condition in addition to the 'atrib[2] = 1' condition required for establishing the truck type (Fig. 6.20).

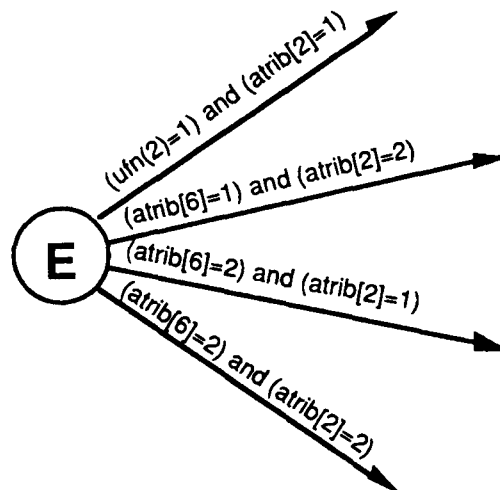


Figure 6.20 Conditional Branching from the Dispatch Point

However, it was not necessary to incorporate the user-defined function into all the activities since an identical calculation would be found to take place as each alternative route was checked. Instead, only the condition of the first activity leading from the dispatching point was altered, atrib[6] for the truck entity is set equal to the selected loading node index within the function, and the remaining activities check for the latter with a considerable time saving.

6.4.4 Dynamic Allocation

Dynamic allocation of the type described in Section 2.7.2 must be allowed to take place after a specified time increment. A further subsidiary network can be automatically added to facilitate this (Fig. 6.21).

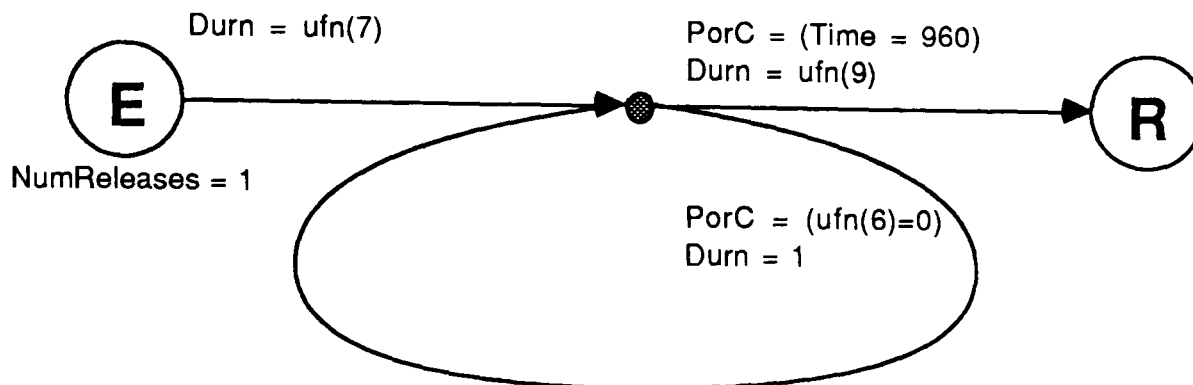


Figure 6.21 Subsidiary Network for Dynamic Allocation

The network consists of:

- an enter node for the creation of a single entity;
- an activity with duration defined as 'ufn(7)' which is always returned as 0 but, at the same time, initialises the array of routes which are used to store information required by the dispatching process;
- a continue node from which only the single entity can be routed;
- an activity with duration defined as 'ufn(8)' and a proborcond of 'Time = 960' which again always returns 0 but, in the process, frees the storage space allocated for the array of routes at the end of the simulation period;
- a remove node which follows it; and
- a looping activity (duration = 1) along which the entity passes by default since the condition 'ufn(6)=0' is always met.
(User-defined function 6 is the dynamic allocation function itself)

As a truck passes through the assign node which immediately follows the dumping activity, its sixth attribute is reset to 0 indicating that it is available for reallocation.

The load points are interrogated in order starting with the one with the current minimum value of X_i as stipulated in Equation 2.21 and stored in the route-arrays described above. The available truck which is expected to arrive at the dispatching point serving its current region closest to this time is allocated to the load point in question by setting its 'atrib[6]' to the node's reference.

Trucks which reach the dispatching point without an allocation are routed through an activity immediately preceding it with duration specified as 'ufn(5)'. This assigns the truck to the load point with the minimum value of X_i taking into account the cross-region damping factor (Equation 2.22). None of the activities branching from the dispatch node need be altered.

Validation of the (semi-)automated simulation network formulation routines can only be carried out by applying actual examples and will therefore be covered in the next section.

6.5 Canadian Test Examples

It would be impossible to qualitatively assess the performance of the haulage analysis module as a means of satisfying both project objectives without referring to specific test cases. Two working areas were chosen - primarily because of their respective geometries. The first is elongated with a large number of loading points and a central dumping area, the second is more compact with loading and dumping points scattered throughout.

It would also be impractical to rigorously analyse a vast quantity of alternative truck fleet and scheduling combinations. Particularly without an effective means of comparing them financially. To simplify matters, it was decided to base all experimental work on the truck fleets currently operating at each mine and to measure suggested improvements in terms of the potential reduction in fleet size (and hence operating expenditure) required to maintain the existing productive capacity.

The example data was collected at two operating open-cast coal mines in western Canada during the years 1984-85 as a means of validating the deterministic simulation model developed by Srajer (1987). A more detailed description of these and the other six mines which were surveyed, can be found in Srajer's Ph.D. thesis.

6.5.1 Quintette - Summary of Operations

Quintette Mine lies in the Rocky Mountain Foothills region of northeastern British Columbia. Two major rivers transect the property creating a natural split into two overall operating regions both of which are subdivided into two coal producing areas. The first of these, the McConkey pit, has four seams with an aggregate thickness of 18m and a stripping ratio of 3.17 bm^3/t , The second, the Frame pit contains 5 seams with an aggregate thickness of 16.6m and a much higher stripping ratio of 7.23 bm^3/t . These are the areas which will be investigated further throughout this section. Figure 6.22 shows the principal haul routes which exist in the region.

Production during 1984 consisted of 3.48 million tonnes of metallurgical coal, 1.79 million tonnes of thermal coal and 34.01 million bank m^3 of overburden material. The mine already employs a dispatching system to help maintain the correct blend of material extraction but no account of this was taken during the original study which was mainly concerned with the factors influencing haul times.

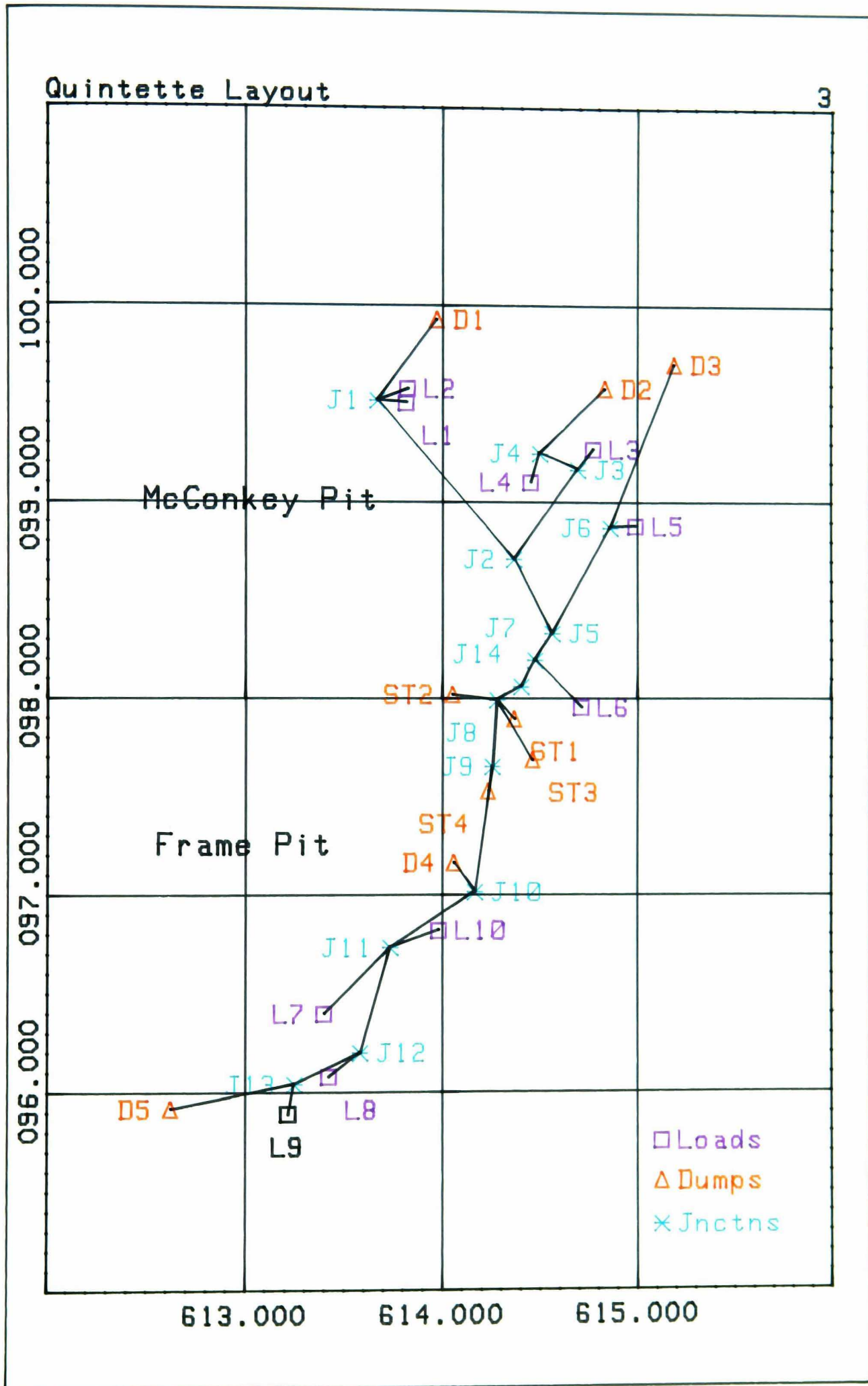


Figure 6.22 Quintette - Principal Haul Routes

The operations taking place at Quintette at the time the study was carried out are summarised in Tables - 6.2a and 6.2b. The four P&H cable and the five Demag H241 electric excavators were all digging waste material as was one of the Demag diesel machines. This material was being hauled, by Wabco 154 and occasionally Terex 33-11C haulage trucks, to waste dumps within the vicinity of each loading area. Figure 6.23 illustrates the valid routes for waste haulage in network terms as described in the previous sections.

At the same time, coal was being excavated from all but one of the mine's 10 loading stations by the remaining Demag loaders and a total of 7 Letourneau machines. This was hauled, principally by the Wabco 136-MT trucks, to one of four central stockpiling points for conveyance to the processing plant. Three of these are set aside for the dumping of the metallurgical coal which forms approximately 70% of total coal production in this region, the fourth is used for stockpiling the thermal coal which makes up the remainder.

In Table 6.2a, the theoretical production figures (in bank m³ per hour) are calculated from the following equations:

$$Q_t = q_c \times n_c \times a_s \quad \dots (6.4)$$

where: Q_t = theoretical quantity (bm³/hr)
 q_c = quantity per cycle (bm³)
 n_c = theoretical cycles per hour
 a_s = scheduled availability (working hours / shift time)

q_c is subject to a bucket fill factor:

$$q_c = c_d \times f_b \quad \dots (6.5)$$

where: c_d = dipper capacity
 f_b = bucket fill factor

n_c is subject to a swing factor:

$$n_c = \frac{3600 \times f_s}{t_c} \quad \dots (6.6)$$

where: f_s = swing factor
 t_c = cycle time (secs)

Loading Equipment	Type	Units	Loading	Dipper (bm ³)	Cycle Time (secs)	Theoretical Production (bm ³ /h)	Budgeted Production (bm ³ /h)	Load Points
P&H 2800-XP	Cable Excavator	4	Waste	22.9	30.0	2289	1029	L1, L3, L9, L10
Demag H241-E	Shovel (electric)	5	Waste	14.5	37.0	1176	450	L2, L5, L6, L7, L8
Demag H241-D	Backhoe(diesel)	1	C+W	14.5	36.0	1208	408	L4 (waste)
Demag H241-D	Shovel(diesel)	2	C+W	14.5	36.0	1208	486	L5 (coal), L8 (coal)
Letourneau L1000	Loader	4	Coal	10.0	13.0	2307	446	L2, L4, L6, L7
Letourneau L1000	Loader	3	Coal	8.0	9.9	2424	275	L1, L3, L10

Table 6.2a Quintette Loading Equipment

Hauling Equipment	Type	Units	Hauling	Weight (tonnes)	Payload (tonnes)	Availability (%)	Utilisation (%)
Wabco 154	Haulage Truck	41	Waste	90.9	154	83	100
Terex 33-11C	Haulage Truck	7	C+W	59.7	77	83	100
Wabco 136-MT	Haulage Truck	5	Coal	82.4	136	83	100

Table 6.2b Quintette Hauling Equipment

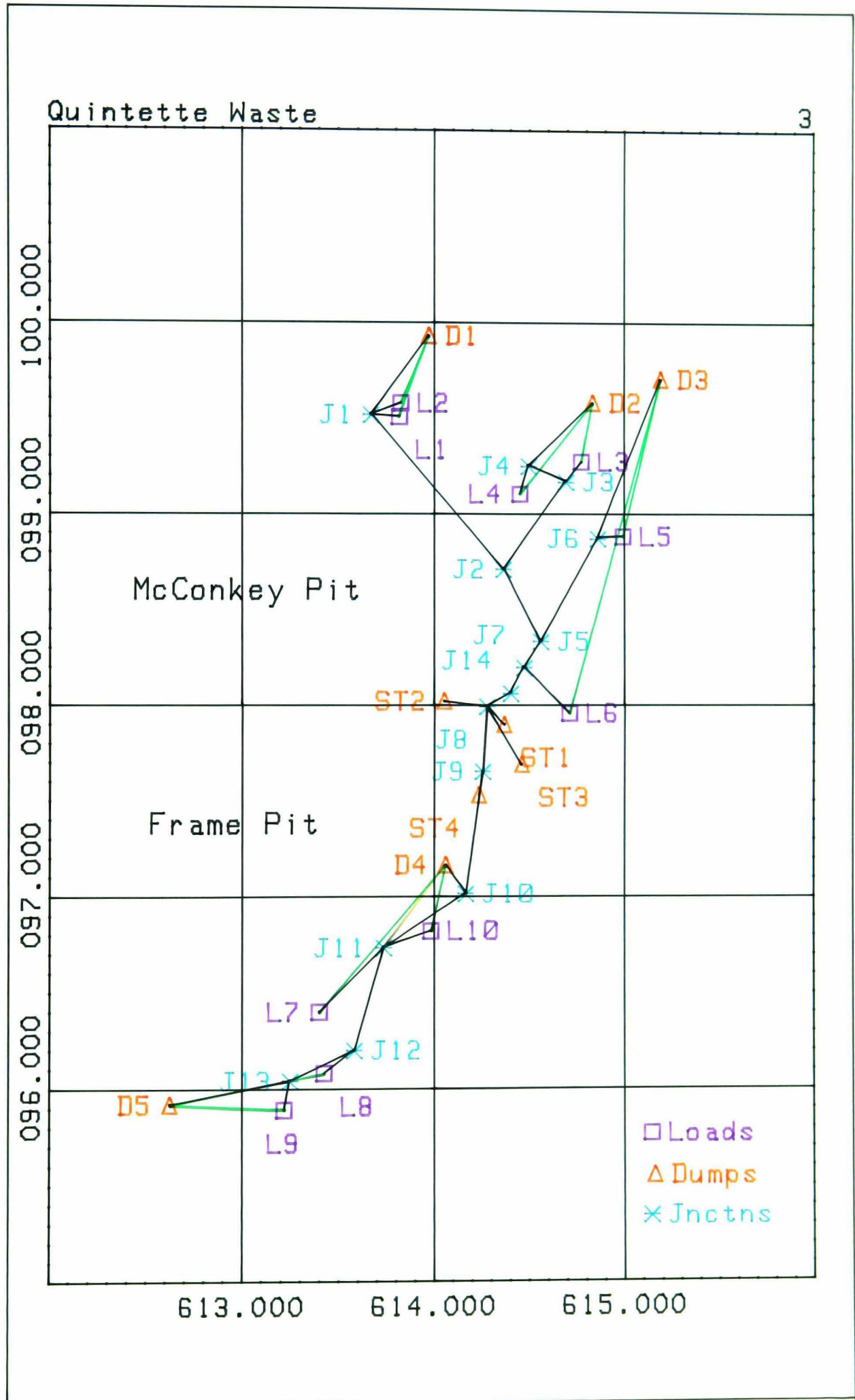


Figure 6.23 Quintette - Waste Haulage

If these values are compared to the actual production capacity (bm³/hr) found to exist during the study (taken as a mean for each equipment type over a period when they are actually operating), it is clear that a significant potential capacity could be harnessed. The large discrepancy between theoretical and actual production values calculated for the haul trucks is indicative of the extreme under-trucking which prevails here as will be seen in Section 6.5.6.

Tables 6.3 and 6.4 show the characteristics and quantities of material being excavated from each load point. The quantities (in tonnes/hour) are calculated using the equation:

$$P = Q_a \times d_b \quad \text{.. (6.7)}$$

where: P = actual production (tonnes/hr)
 d_b = bank density (tonnes/m³)

No information regarding the source of the thermal coal was available. It was assumed, for the purposes of the exercise, that this material was being excavated from two load points, namely L4 and L5, which together constitute approximately 30% of the total coal output.

Again, for coal, a large degree of incompatibility is evident between the total productive capacity and the budgeted target value. This is because the quoted total assumes that all 9 coal loaders will be operating simultaneously. In fact, very low utilisation factors are associated with the coal shovels which are continuously being moved from site to site. This has the knock on effect of reducing overall availability due to breakdowns and other non-mechanical delays but also reduces the amount of under-trucking.

6.5.2 Coal Mountain - Summary of Operations.

The coal reserves at the Coal Mountain mine, in southeastern British Columbia, are classified as medium volatile bituminous. The main, 'Mammoth' seam varies in depth from several to 214 metres and the total production figures for 1984 were 1.5 million tonnes of raw coal and 0.53 million tonnes of high-ash coal. During the same year, 3.13 million bank m³ of waste material was removed.

Materials	Bank Density (t/m ³)	Loose Density (t/m ³)	Swell factor
Waste	2.600	1.926	1.35
Metallurgical Coal	1.550	1.192	1.30
Thermal Coal	1.420	1.092	1.30

Table 6.3 Quintette Material Characteristics

Load Point	Waste (t/h)	Met. Coal (t/h)	Thermal Coal (t/h)
L1	2675	426	
L2	1170	691	
L3	2675	426	
L4	1061		633
L5	1170		690
L6	1170	691	
L7	1170	691	
L8	1170	753	
L9	2675		
L10	2675		
Total	17611	4104	1323
Target	10093	1503	413

Table 6.4 Quintette Production

The area is a steep-sided ridge divided longitudinally into two structured blocks with coal and waste being mined from both areas simultaneously. Figure 6.24 illustrates the principal haul routes. Tables 6.5a and b, 6.6 and 6.7 summarise the operations taking place at the time of the study. Metallurgical coal was being hauled from load points L1 and L2 to dump D1 while high-ash coal was transported from L1 to either D2 or D3. The other seven dump points were all being used for waste disposal and it is assumed that an even distribution of waste was required not only for geotechnical reasons but also to avoid truck congestion in the central area. The values listed in Table 6.5a were derived in exactly the same way as those for Quintette above.

6.5.3 Quintette LP Input

Before embarking on any optimisation exercise, the potential for improvement by applying a particular technique should be either apparent or indeterminable without carrying out the exercise. In the case of waste haulage at the Quintette mine (Fig. 6.23), the proximity of the dumping sites to the loading points makes for an efficient system on which little improvement can be made. Any reallocation by linear programming may increase productivity slightly but it will also offset the rate of dumping at each waste tip. As a result, only coal haulage has been considered.

In Figure 6.25, the green lines indicate all the possible paths which can be taken by the coal trucks. Since all the stockpiling areas lie within the central region, it was possible to combine the 3 metallurgical coal dumping points into a single node (ST2) to which all the trucks hauling this material are directed. The other valid hauling routes lead from the thermal coaling areas (L4 and L5) to dump ST3. In the closed out configuration, trucks must return along the same paths from which they came. However, in the synergised configuration, trucks should be able to return to any load point after dumping. Therefore, in the diagram, routes to all 9 load points emerge from each dump point.

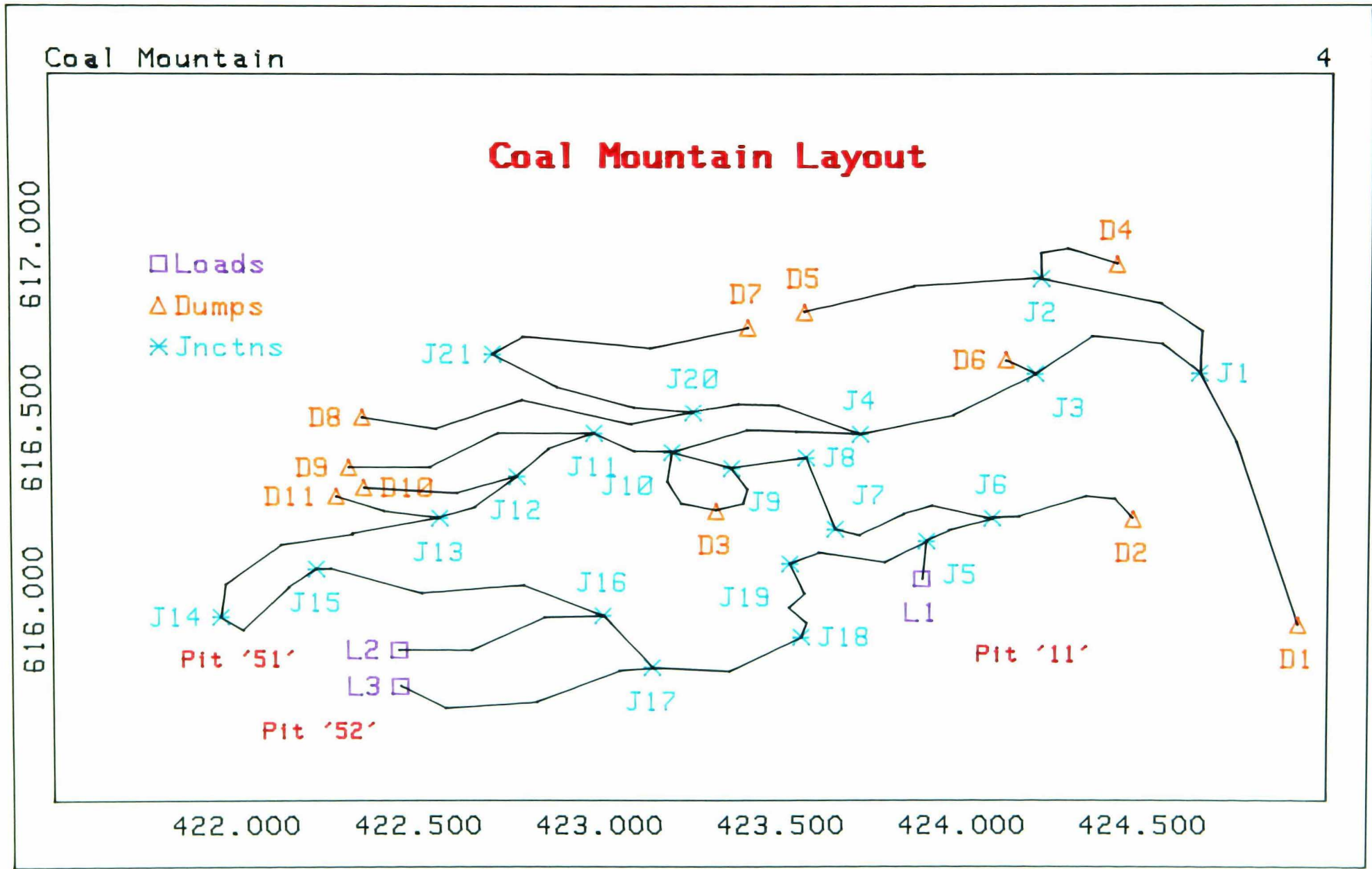


Figure 6.34 Coal Mountain - Principal Haul Routes

Loading Equipment	Type	Units	Loading	Dipper (bm3)	Cycle Time (secs)	Theoretical Production (bm ³ /h)	Budgeted Production (bm ³ /h)	Load Points
Cat 992-C	Loader	2	C+W	9.5	42.0	679	404	L2a (c),L2b (w)
Cat 992-B	Loader	3	C+W	14.5	42.0	1036	506	L1b,L1c (c), L3 (w)
Demag H241-D	Shovel (diesel)	1	C+W	14.5	36.0	1208	486	L1a (waste)

Table 6.5a Coal Mountain Loading Equipment

Hauling Equipment	Type	Units	Hauling	Weight (tonnes)	Payload (tonnes)	Availability (%)	Utilisation (%)
Cat 777	Haulage Truck	13	Waste	58.9	77	75.1	52.6
Unit Rig MK30	Haulage Truck	5	Coal	75.8	108.8	76.1	59.0

Table 6.5b Coal Mountain Hauling Equipment

Materials	Bank Density (t/m ³)	Loose Density (t/m ³)	Swell factor
Waste	2.090	1.492	1.40
Metallurgical Coal	1.330	0.985	1.35
Thermal Coal	1.330	0.985	1.35

Table 6.6 Coal Mountain Material Characteristics

Load Point	Waste (t/h)	Met. Coal (t/h)	Thermal Coal (t/h)
L1a	1016		
L1b			673
L1c		673	
L2a		537	
L2b	844		
L3	1057		
Total	17611	4104	1323
Target	10093	1503	413

Table 6.7 Coal Mountain Production

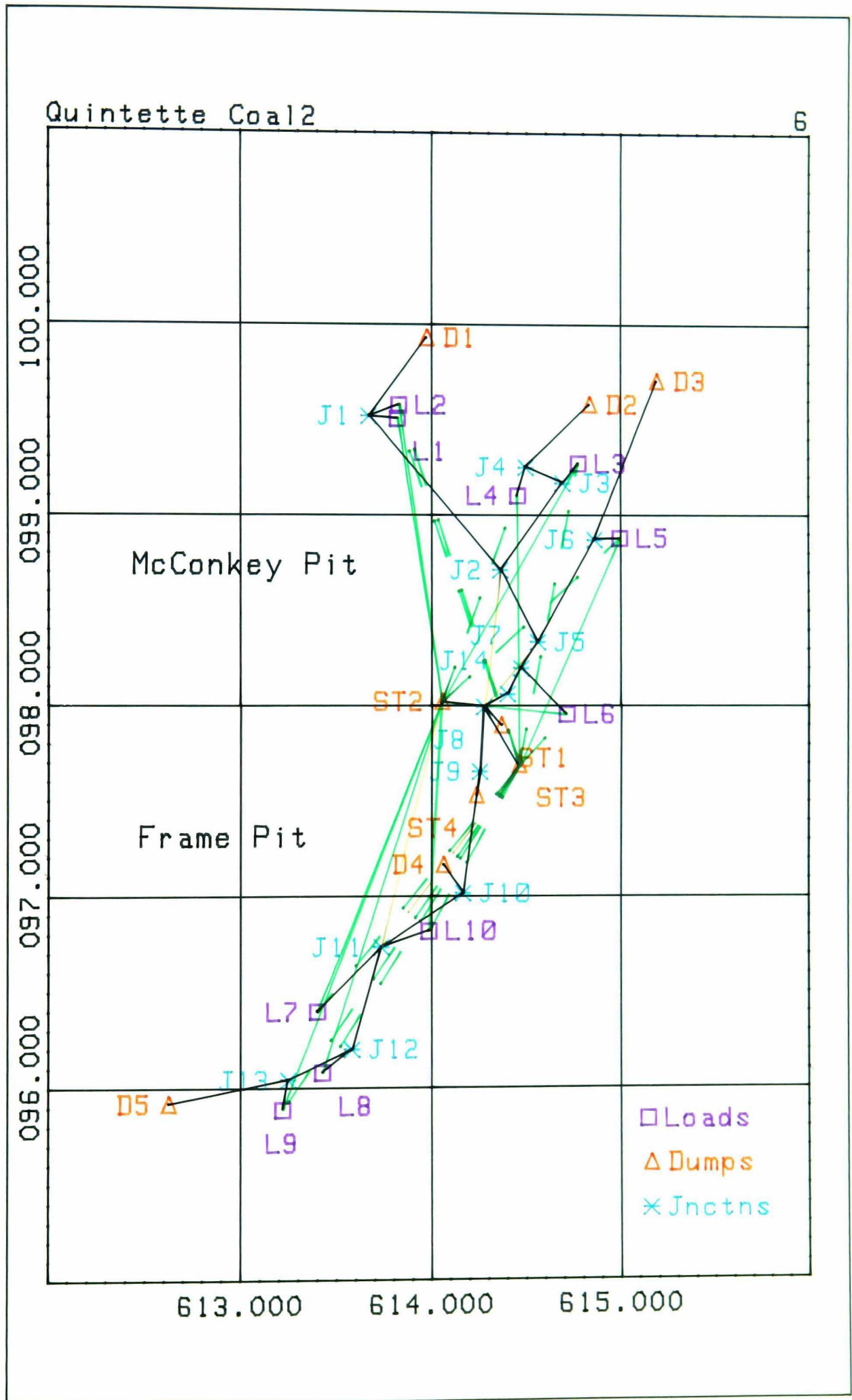


Figure 6.25 Quintette - Coal Haulage

6.5.4 Quintette LP Results

Fig 6.26 compares the calculated times taken for the two types of coal hauler used at Quintette to traverse each route used in fixed allocation mode (routes 1 to 8 refer to load points L1 to L8, route 9 refers to L10). It illustrates two things:

- ❑ The relative speeds of the two truck types, (Wabco slower than Terex)
- ❑ Those haul routes which take the longest time to traverse (1,2,4,7,8 & 10).

Tables 6.8 and 6.9 show the required allocation of trucks to shovels based on these times if a fixed regime was utilised ('number of trucks'). The total capacity figures indicate how much of each truck's potential is being wasted (compare to required production). This is the result of the overtrucking required to match equipment in a fixed regime. Assuming that the loaders are operating close to their limit and stockpiling, since with only one serving truck there would otherwise be an unacceptably long idle period during each cycle, the overtrucking would cause an increase in truck waiting at each load point.

The figures at the bottom of the 'required trips per hour' columns are the minimum objective functions which can be achieved with 'perfect' dispatching. The tables also include calculated values for the objective functions if the total capacities data had been applied to the linear programming module. While the objective functions are less for the smaller and therefore 'easier to match' Terex machines than for the Wabco's, there is a significant difference of at least 4 trucks between each pair of independent solutions. It is apparent that none of the solutions can be met using the existing trucks individually (i.e. 5 Wabco's or 7 Terex's) and that a combination of the two is therefore required.

Tables 6.10a and 6.10b show the objective functions and numbers of each type of truck required assuming 'perfect' dispatching in a closed out and a synergised configuration respectively. They illustrate that when the linear programming criteria were applied to this particular network, only a tiny improvement could be achieved. This was put down to the proximity of the two dumping stations in the centre of the mine and the minimal difference between their respective distances from the various load points.

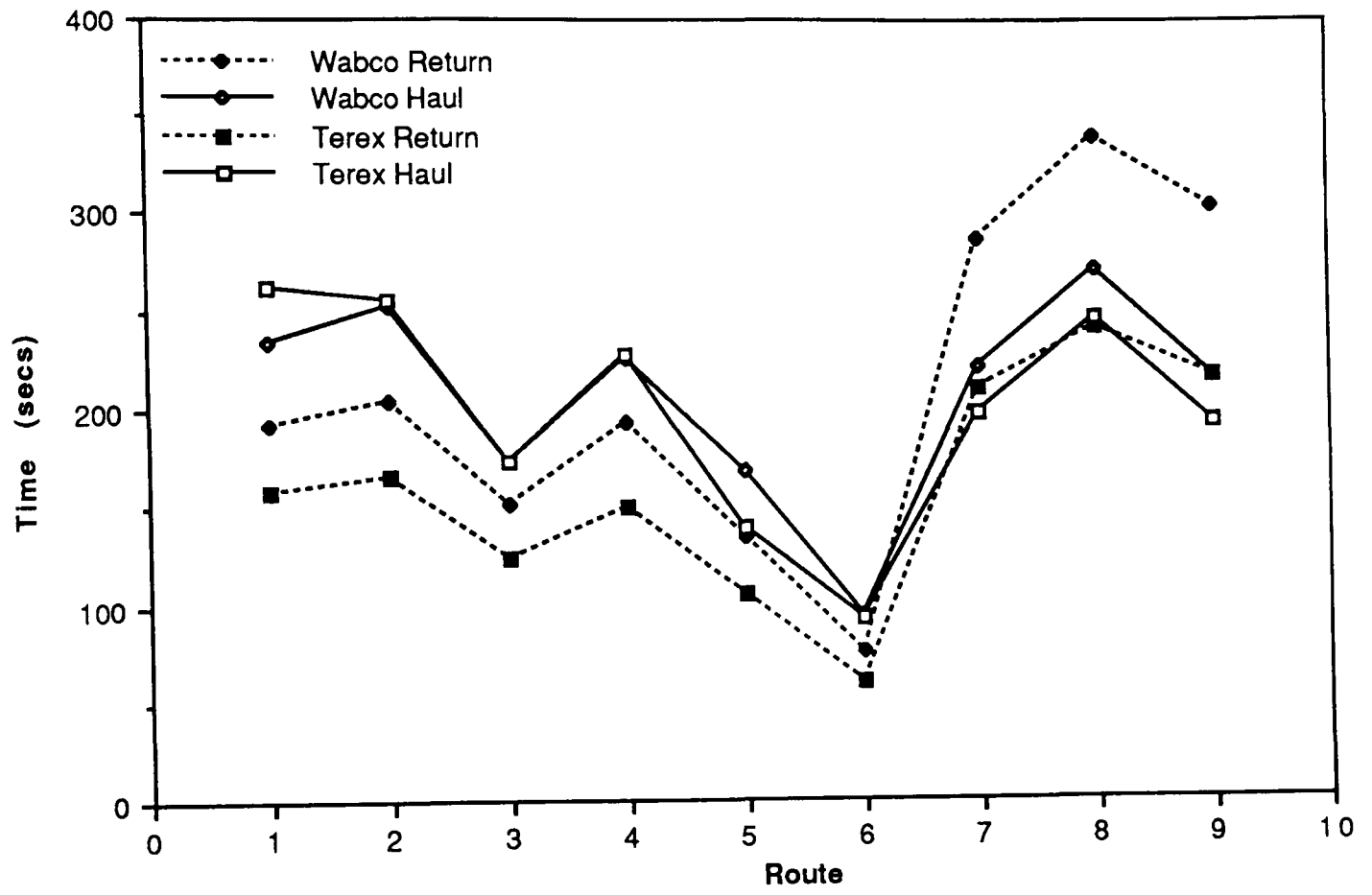


Figure 6.26 Comparison of Haul and Return times for 'Fixed' routes

Route	Travel Time (secs)	Loading & Dumping (secs)	Cycle Time (secs)	Trips/hr for 1 Truck	Required Production Tonnes/hr	Required Trips per hr	Number of Trucks	Total Capacity
ST2-L1	426.74	150	576.74	6.24	426	3.13	1	848.91
ST2-L2	457.92	150	607.92	5.92	691	5.08	1	805.37
ST2-L3	325.80	150	475.80	7.57	426	3.13	1	1029.00
ST3-L4	421.02	150	571.02	6.30	633	4.65	1	857.41
ST3-L5	303.79	200	503.79	7.15	690	5.07	1	971.83
ST2-L6	171.22	150	321.22	11.21	691	5.08	1	1524.19
ST2-L7	508.68	150	658.68	5.47	691	5.08	1	743.30
ST2-L8	610.99	200	810.99	4.44	753	5.54	2	1207.41
ST2-L10	521.10	150	671.10	5.36	426	3.13	1	729.55
				Total Number of Trucks		6.44	10	
				Objective Function		875.33		1360.00

Table 6.8 Quintette - Wabco Potential in a fixed regime assuming perfect dispatching

Route	Travel Time (secs)	Loading & Dumping (secs)	Cycle Time (secs)	Trips/hr for 1 Truck	Required Production Tonnes/hr	Required Trips per hr	Number of Trucks	Total Capacity
ST2-L1	419.24	150	569.24	6.32	426	5.53	1	486.97
ST2-L2	422.41	150	572.41	6.29	691	8.97	2	968.54
ST2-L3	297.54	150	447.54	8.04	426	5.53	1	619.39
ST3-L4	378.76	150	528.76	6.81	633	8.22	2	1048.49
ST3-L5	246.27	200	446.27	8.07	690	8.96	2	1242.30
ST2-L6	153.10	150	303.10	11.88	691	8.97	1	914.55
ST2-L7	408.37	150	558.37	6.45	691	8.97	2	992.89
ST2-L8	488.57	200	688.57	5.23	753	9.78	2	805.15
ST2-L10	410.78	150	560.78	6.42	426	5.53	2	988.62
				Total Number of Trucks		10.19	15	
				Objective Function		784.44		1155.00

Table 6.9 Quintette - Terex Potential in a fixed regime assuming perfect dispatching

Combination	Objective Function	No. Wabcos	No. Terex
Wabcos Only	875.34	6.44	0.000
5 Wabcos	844.63	5	2.138
4 Wabcos	824.08	4	3.637
3 Wabcos	807.01	3	5.182
2 Wabcos	797.21	2	6.821
Unlimited (all Terex)	784.58	0	10.189

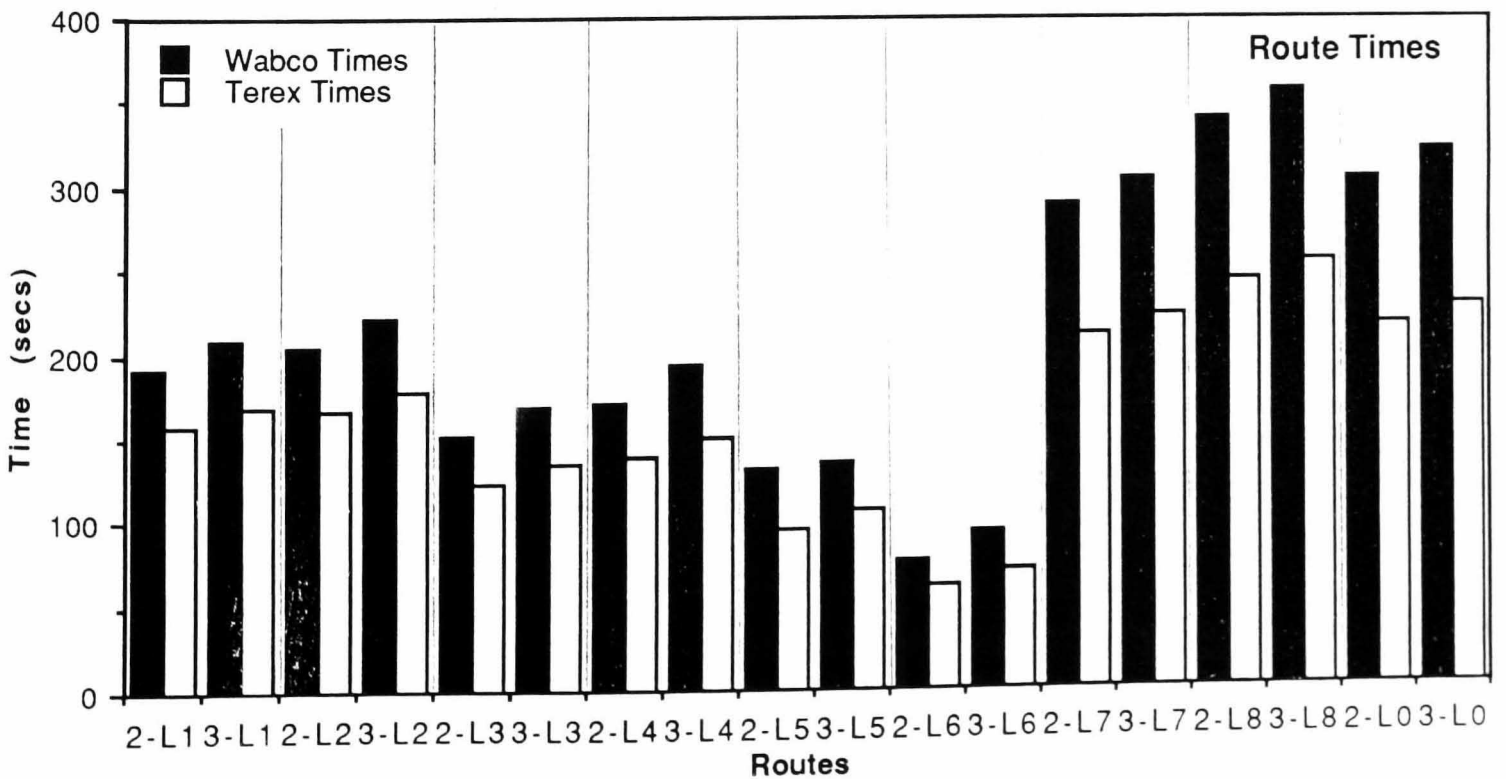
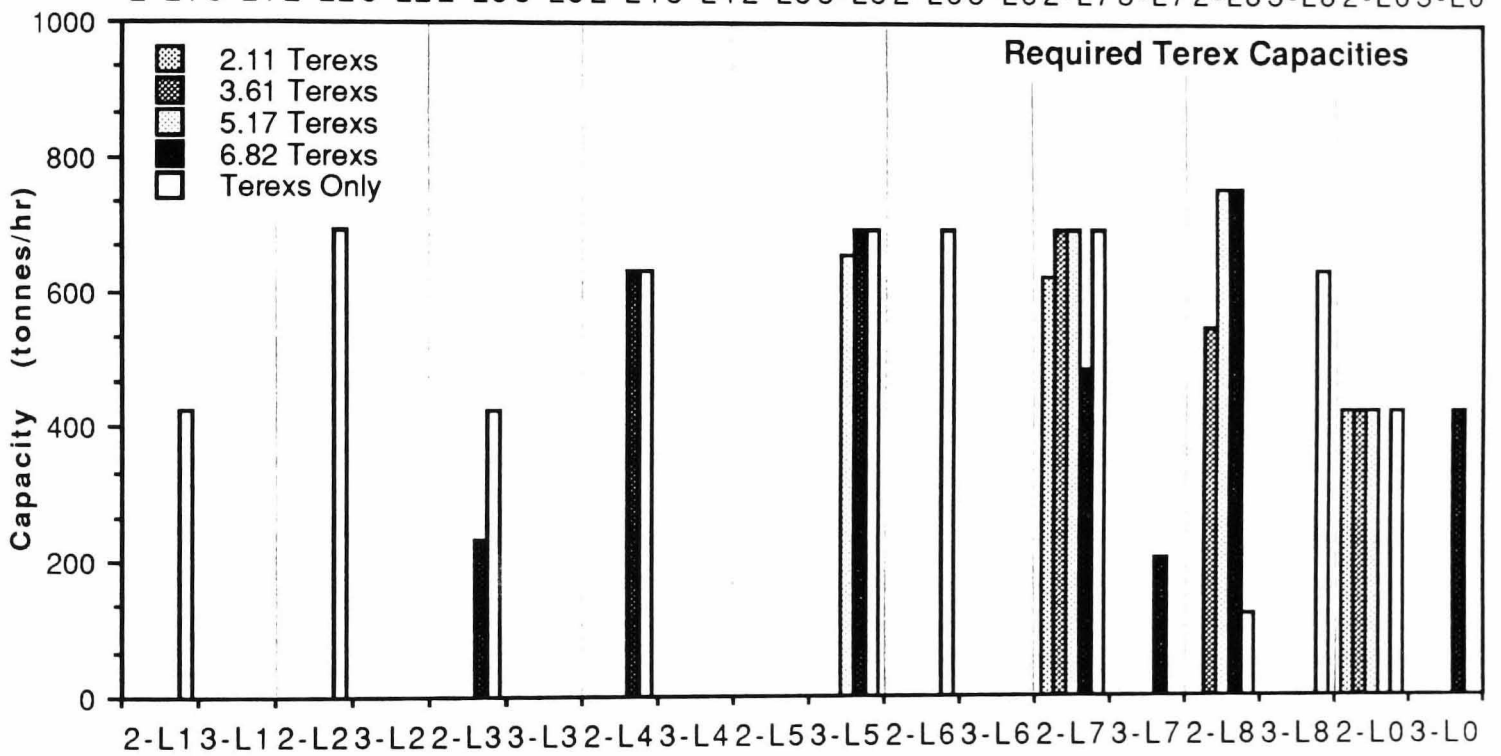
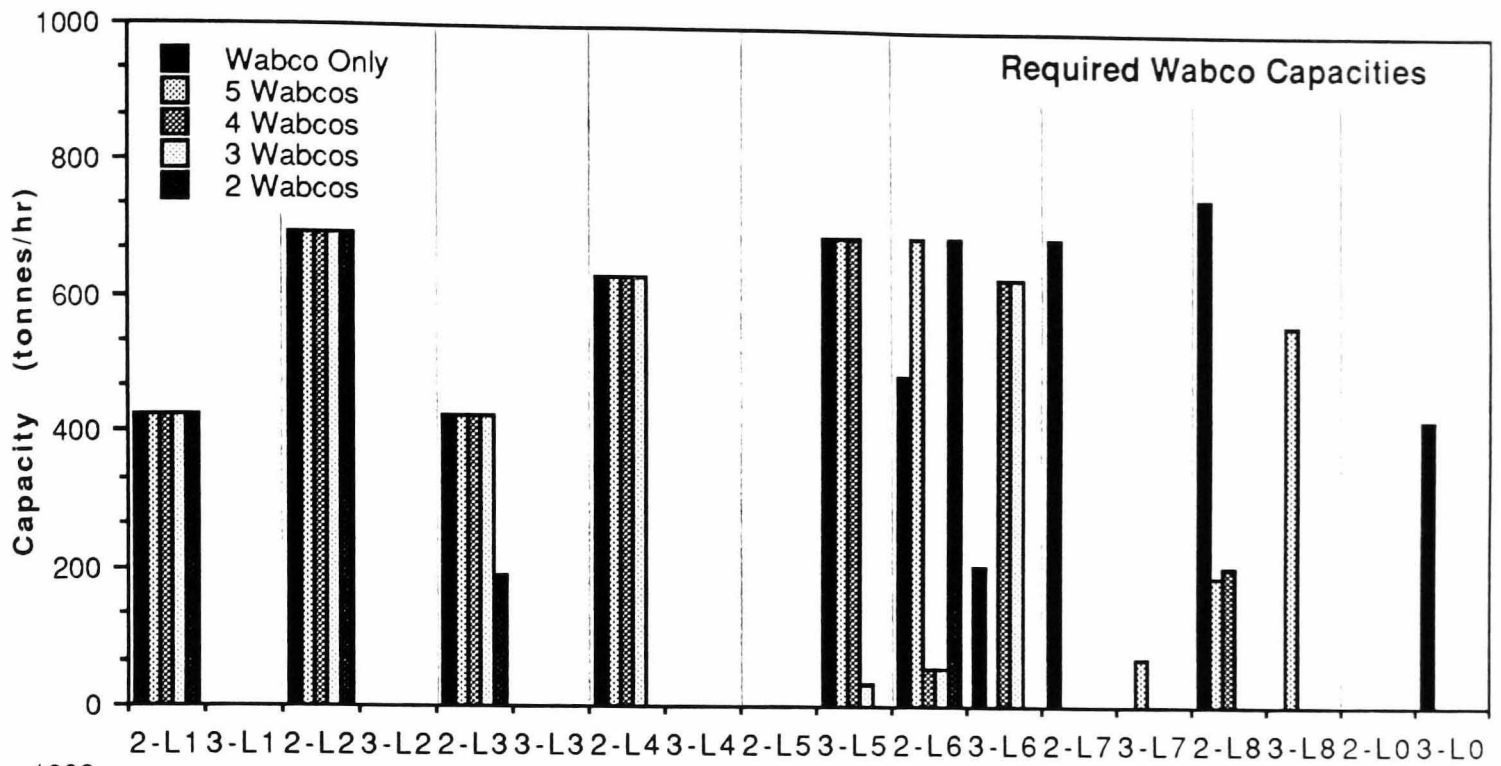
Table 6.10a Required trucks and corresponding objective functions when in a closed-out configuration.

Combination	Objective Function	No. Wabcos	No. Terex
Wabcos Only	873.14	6.42	0.000
5 Wabcos	842.77	5	2.110
4 Wabcos	822.22	4	3.613
3 Wabcos	806.85	3	5.177
2 Wabcos	797.14	2	6.820
Unlimited (all Terex)	784.51	0	10.189

Table 6.10b Required trucks and corresponding objective functions when in a 'synergised' configuration.

Nevertheless, Figure 6.27 shows how the flow rates calculated for each route using the linear programming module are related to the travel times along them. Whereas in a closed out configuration, trucks allocated to L4 would normally be routed from L3, the proximity of ST2 to L4 makes the use of this route more suitable. Figure 6.27 also shows how varying the number of each type of truck affects the derived solution. As more Terex machines are introduced, they are initially allocated to the longer routes due to the absolute difference in travelling times. However, if the production required at a remote loading point is great (e.g. L1-L4 & L8), the larger Wabco's are retained.

The difference in calculated travel times actually cause the solutions for the two truck types operating independently to differ from each other.



Throughout the above exercises, the number of Wabco trucks was varied from 2 to 5 and the number of Terex's needed to make up the rest of the required productive capacity was calculated. Though we are not specifically interested in the financial implications of the derived solutions, this information could be used to calculate an 'optimum' combination. For example, if the numbers of each type listed in Tables 6.10a and 6.10b were multiplied by an operating cost and divided by availability and utilisation factors, a curve in the form illustrated in Figure 6.28 could be produced with the optimum cost being a horizontal line through the curve's lowermost point.

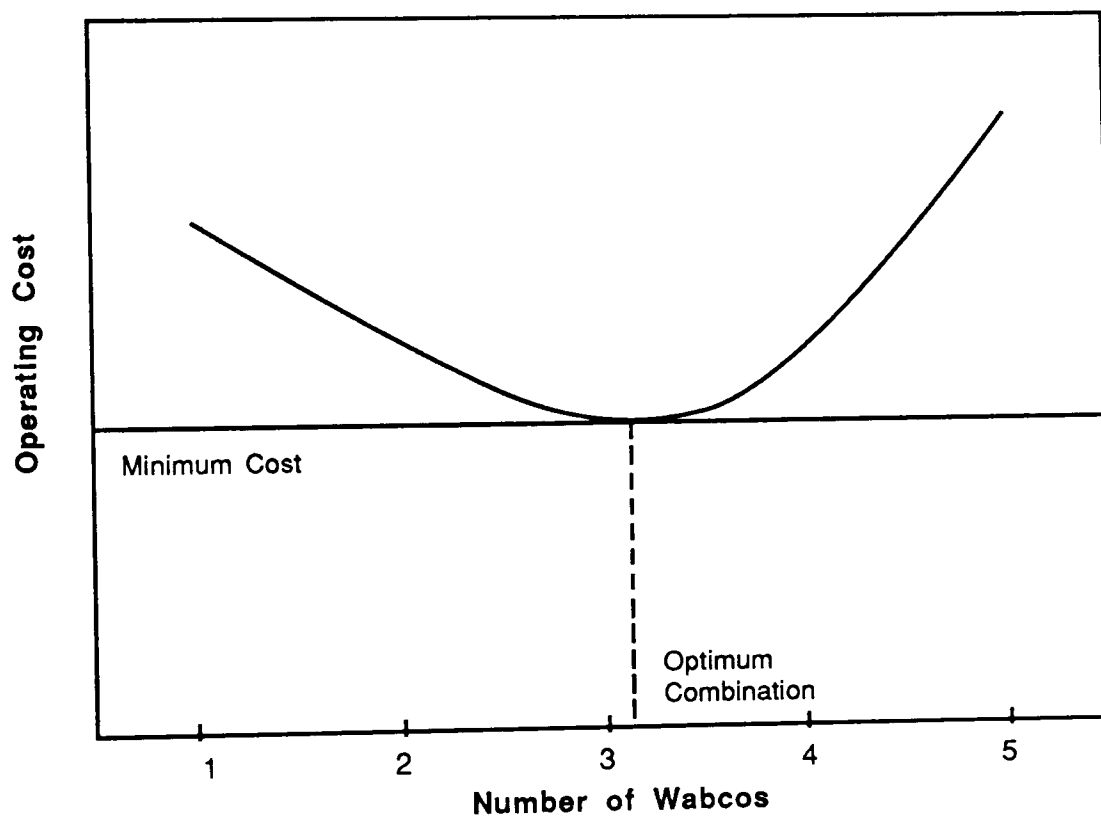


Figure 6.28 Selection of best solution by cost.

Figure 6.27 (Previous Page)
 Relationship between allocated capacities and times for optimal route selection with various truck combinations

6.5.5 Quintette Simulation Input

None of the preceding results would be of any great significance if the correlation with what could be achieved in reality was actually known. All the mathematically derived solutions assume perfect dispatching and no account is taken of the interference that exists between different items of equipment. This is where the second objective becomes important and simulation is the only practical method of achieving it.

Validation of the simulation network, in particular decisions relating to the length and number of individual runs, were carried out using Wabco's only in a fixed orientation. Subsequent dispatching policy tests were assumed to be fundamentally similar and the same experimental itinerary was therefore applied.

6.5.6 Quintette Simulation Results

To determine the required number and length of simulation runs, a total of 20 runs of one week in length were initially carried out with intermittent recording of shift by shift and daily totals. It was ascertained that two shift days produced a much greater degree of correspondance with each other than single shifts taken individually and that hour-long simulation runs of one week were simply too long for practical analysis. Using a 5% confidence limit, it was determined that 10 runs were sufficient to produce mean values for the comparison of mathematically derived and dispatch-affected solutions.

Figure 6.29 illustrates how, in a fixed orientation, truck productivity relates inversely to the cycle-time of the route to which it has been allocated. The result is that there is no correspondance to required production. It is also clear that, with as many as 9 Wabco's, overall productivity falls well short of the total required. Consequently, in the following examples, the maximum number of Wabco's available (5) are retained, while the number of Terex's are increased from 3 up to 7 thus providing an immediate comparison with the 2.11 Terex's deduced mathematically in Table 6.10b. Ten runs were carried out for each combination.

Figure 6.30 shows how the 'maximise shovels' policy produces a more even spread of productivity which is related directly to the cycle times of the various routes. Again, there is no correlation to required production and overall capacity falls well below the total requirement.

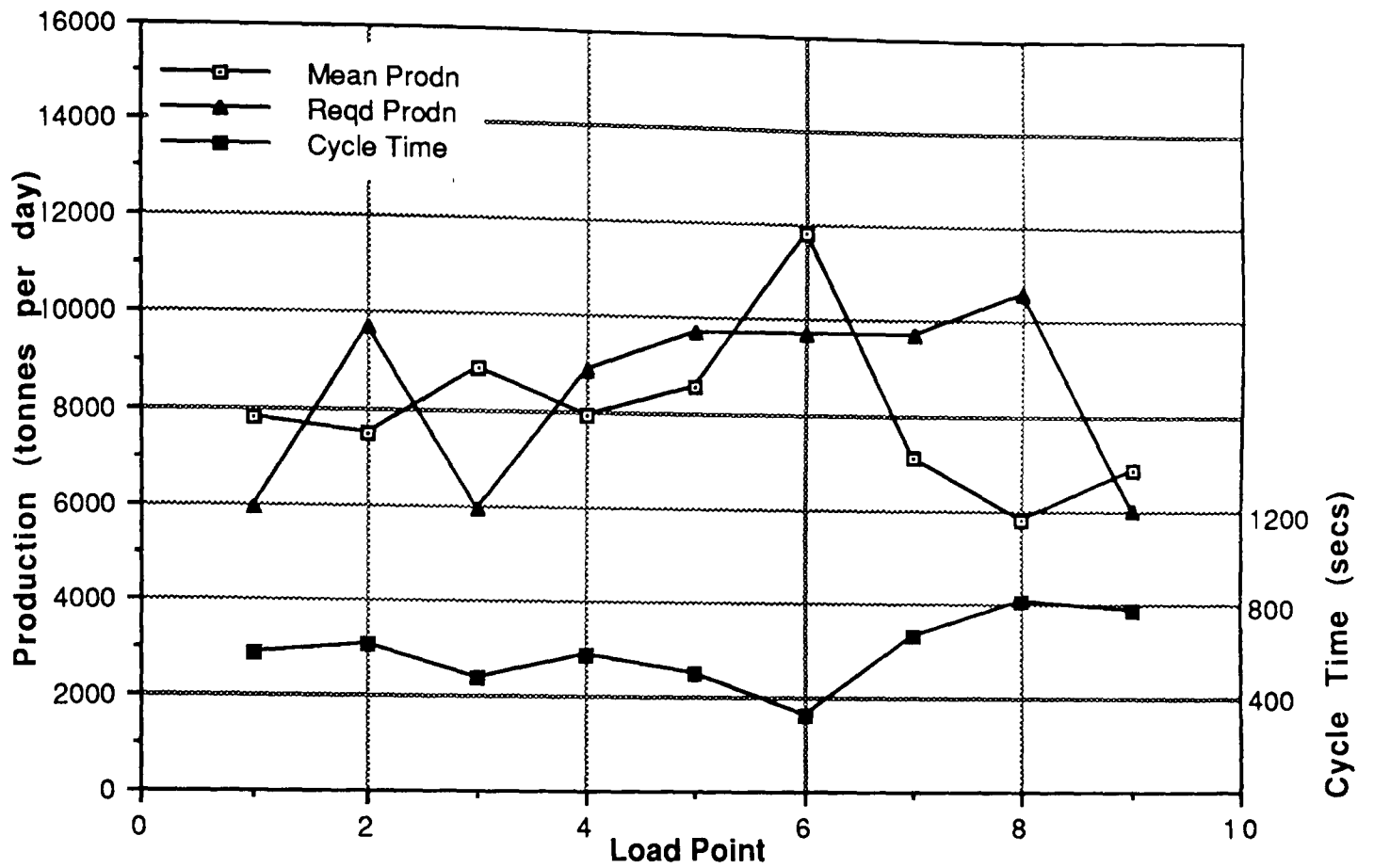


Figure 6.29 Quintette - Simulated production for fixed allocation (one Wabco to each load point)

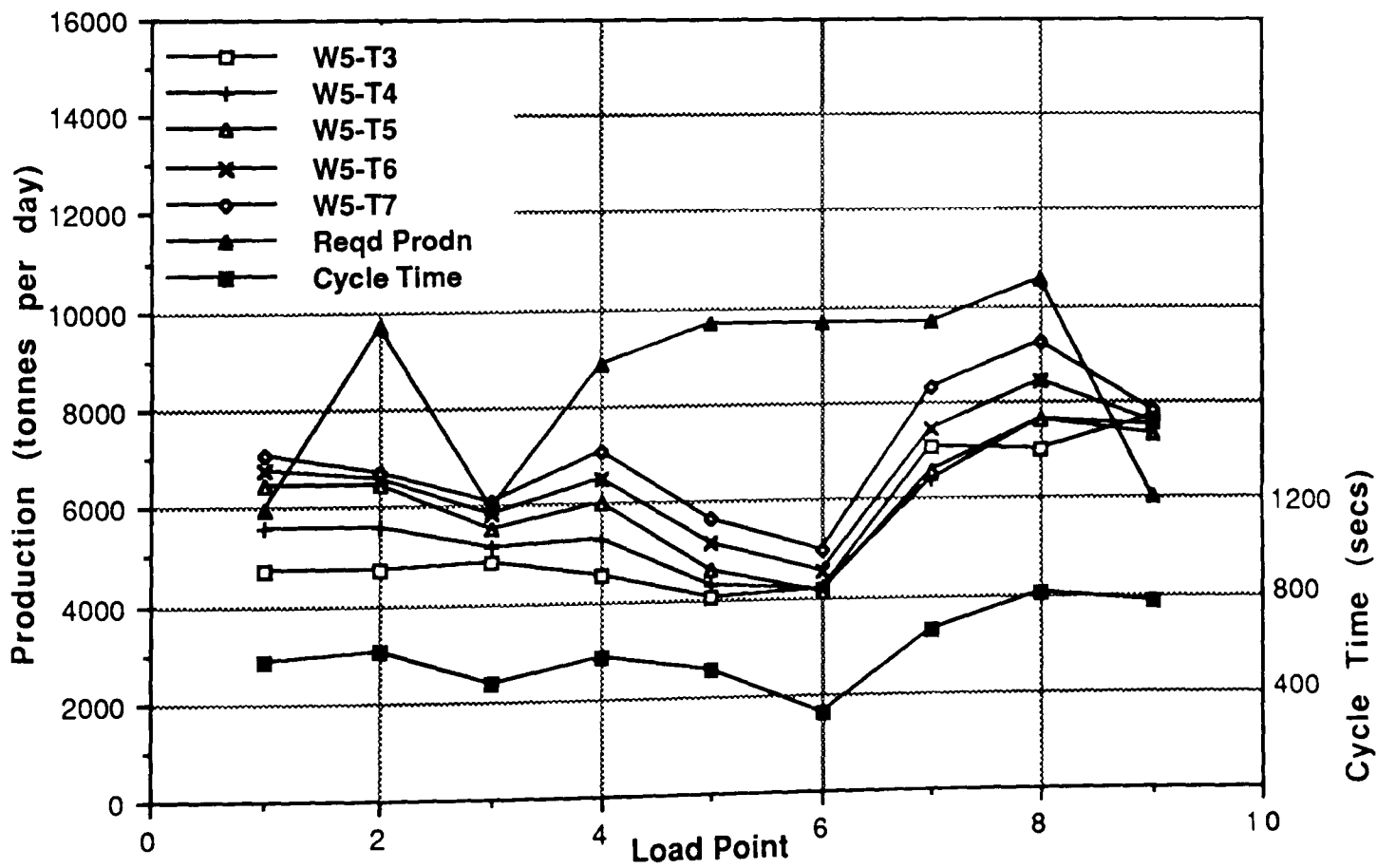


Figure 6.30 Quintette - Simulated Production for various truck combinations using the maximise shovels policy

Figure 6.31 illustrates the degree of undertrucking which would be encountered if a 'maximise trucks' policy were employed at Quintette. Most of the routes considered first in the dispatching routine, particularly those close to the dispatching point, achieved productivities in excess of what is required but the policy is such that those considered last (L10) and those which are far away (L7 and L8) receive no allocations.

Figure 6.32 shows how, as expected, the 'behind schedule' policy produces results which are in direct proportion to what is required but totals which are still too low since no account is taken of relative distances and interference.

The 'match factor' policy, (Fig. 6.33) produces a more even spread of productivities along similar lines to the 'maximise trucks' policy. Only the last point considered (L10) receives no allocations due to undertrucking. There is no correlation with required production.

Figure 6.34 shows how the dynamic allocation of trucks produces results which correspond approximately to required production and go some way towards achieving the overall production goal. However, total capacity is less than that achieved using the 'behind schedule' policy which suggests that there is some potential which can still be harnessed.

A series of runs were carried out to try to establish a weighting factor to be applied to the blending requirement portion of Equation 2.21 (Section 2.7.2) i.e. H_{i0}/P_i so that it would assume either greater or less importance compared to the 'synergy' portion ($L_i - T_i > 0$). Hence the new dynamic allocation equation became

$$\text{Minimise } X_i \quad \text{where } X_i = L_i + F \times H_{i0} / P_i - T_i \quad \dots (6.8)$$

Figure 6.35 shows the results obtained for 5 Wabco's and 7 Terex's for a range of values of F. It is difficult to determine an optimum value since none of the graphs exceed required production for all the load points but if the total production figures obtained are plotted against F (Fig. 6.36), an optimum value of 0.6 is established for this particular configuration. It must be remembered, however, that the match with required production is less pronounced as a result.

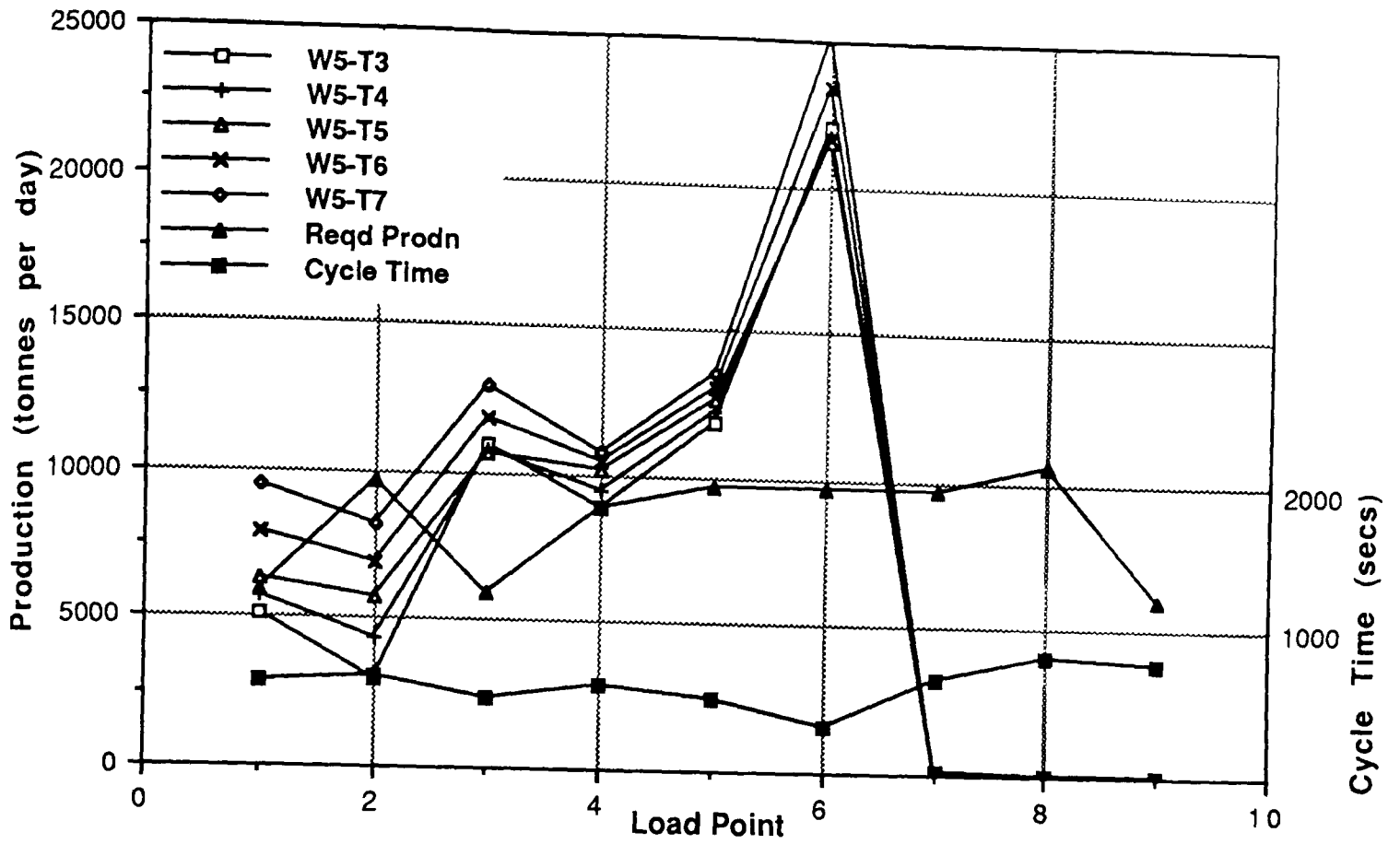


Figure 6.31 Quintette - Simulated Production for various truck combinations using the maximise trucks policy

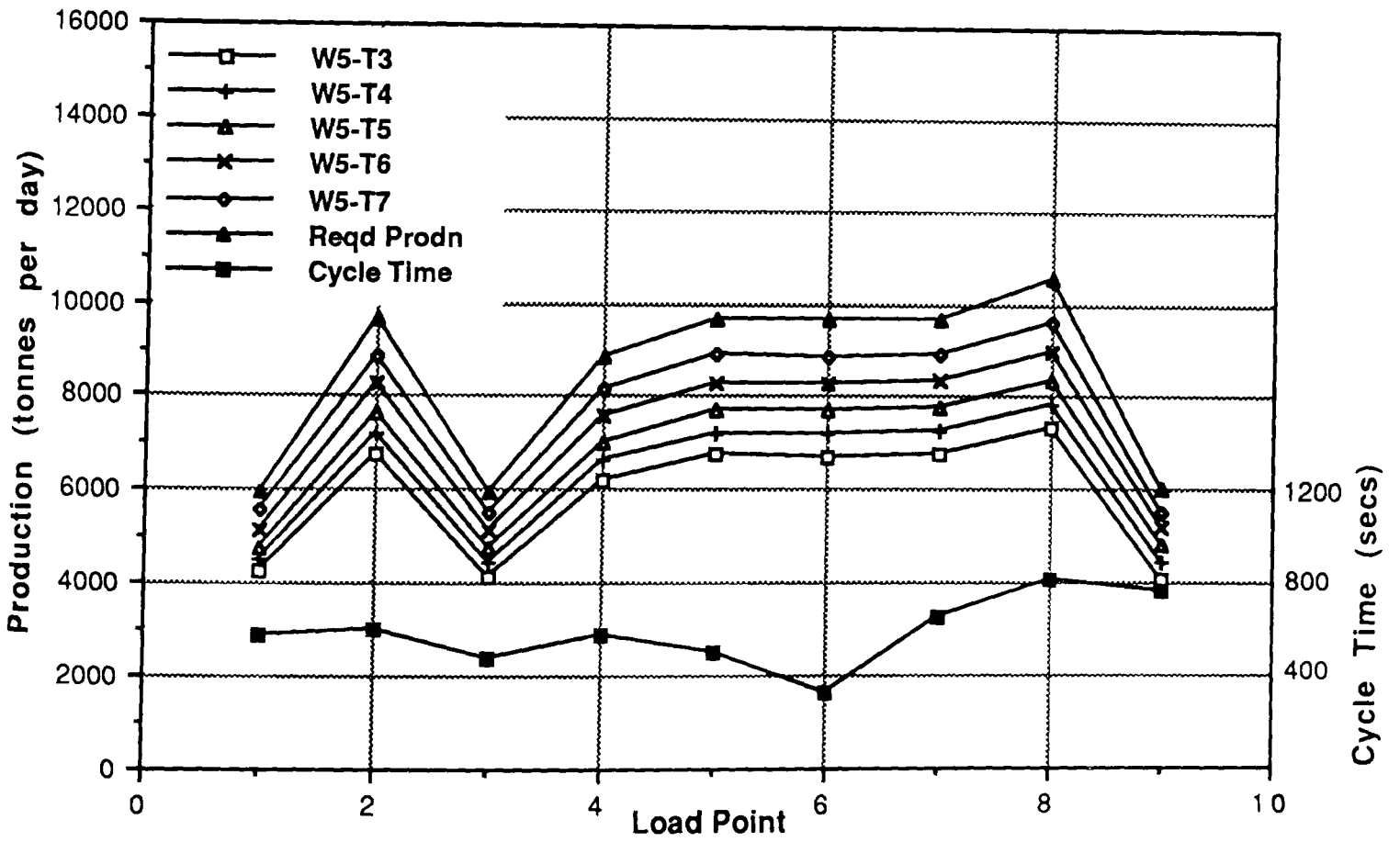


Figure 6.32 Quintette - Simulated Production for various truck combinations using the "behind schedule" policy

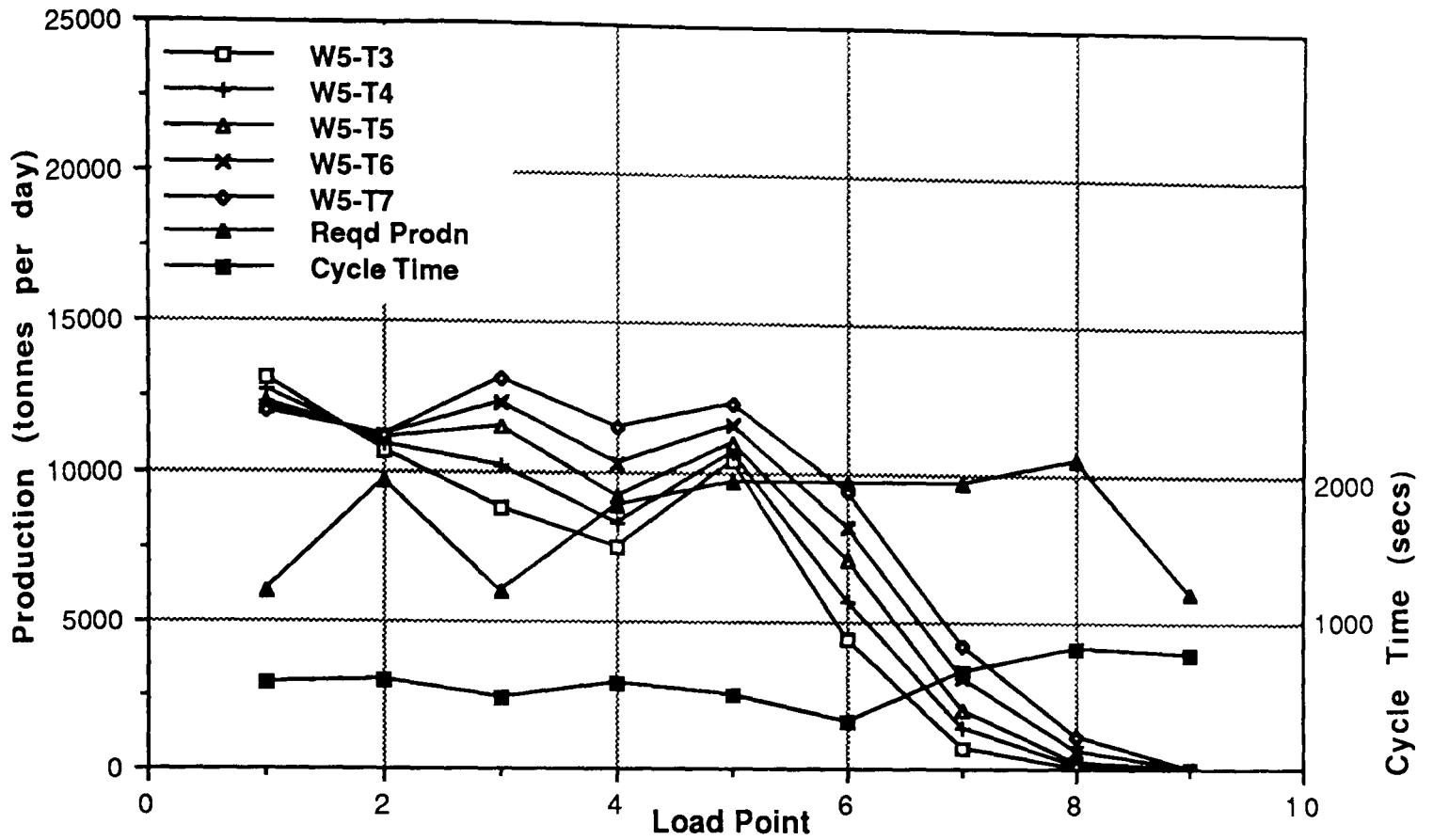


Figure 6.33 Quintette - Simulated Production for various truck combinations using the 'match factor' policy

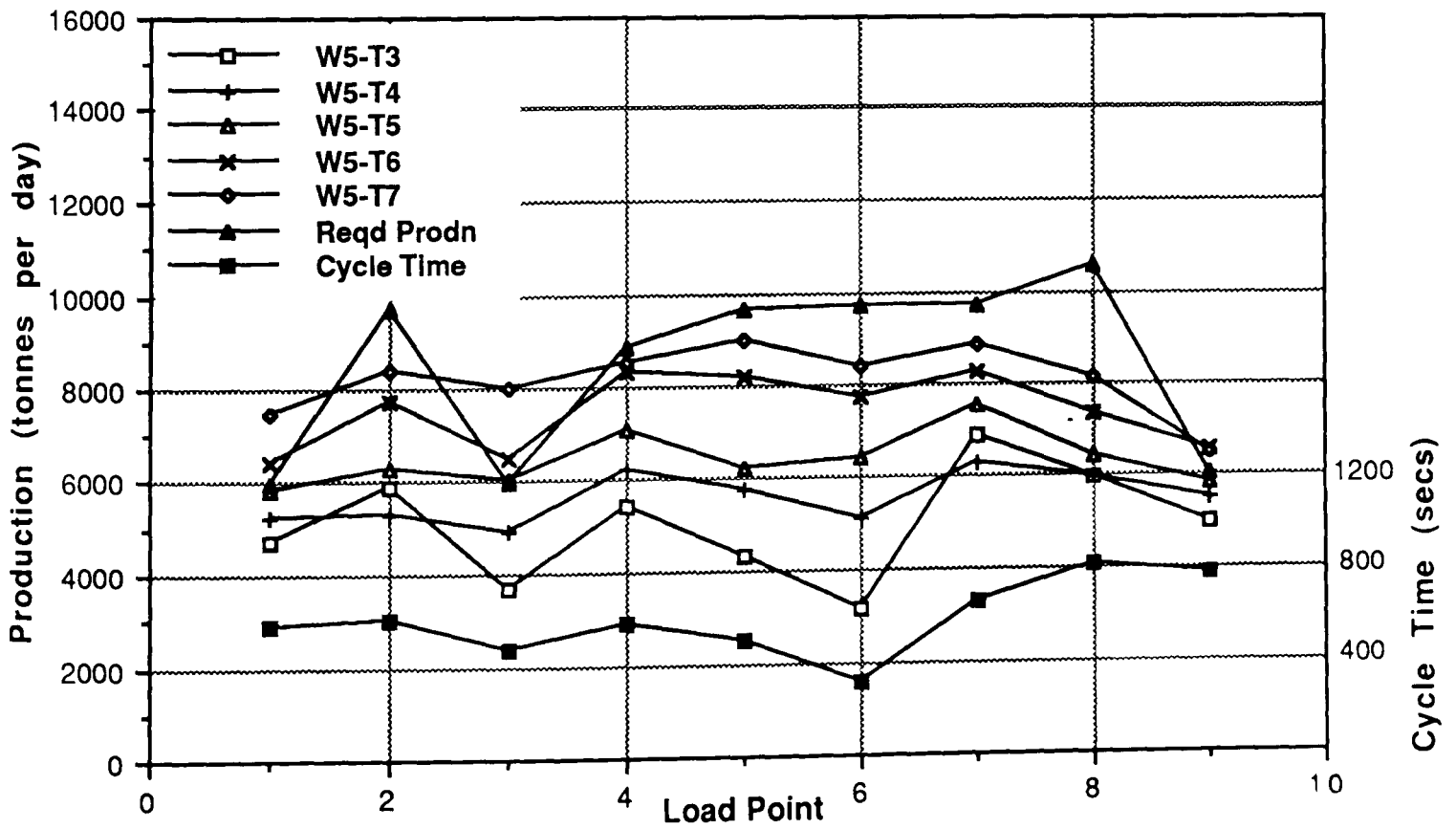


Figure 6.34 Quintette - Simulated Production for various truck combinations using 'dynamic allocation' with factor F = 1

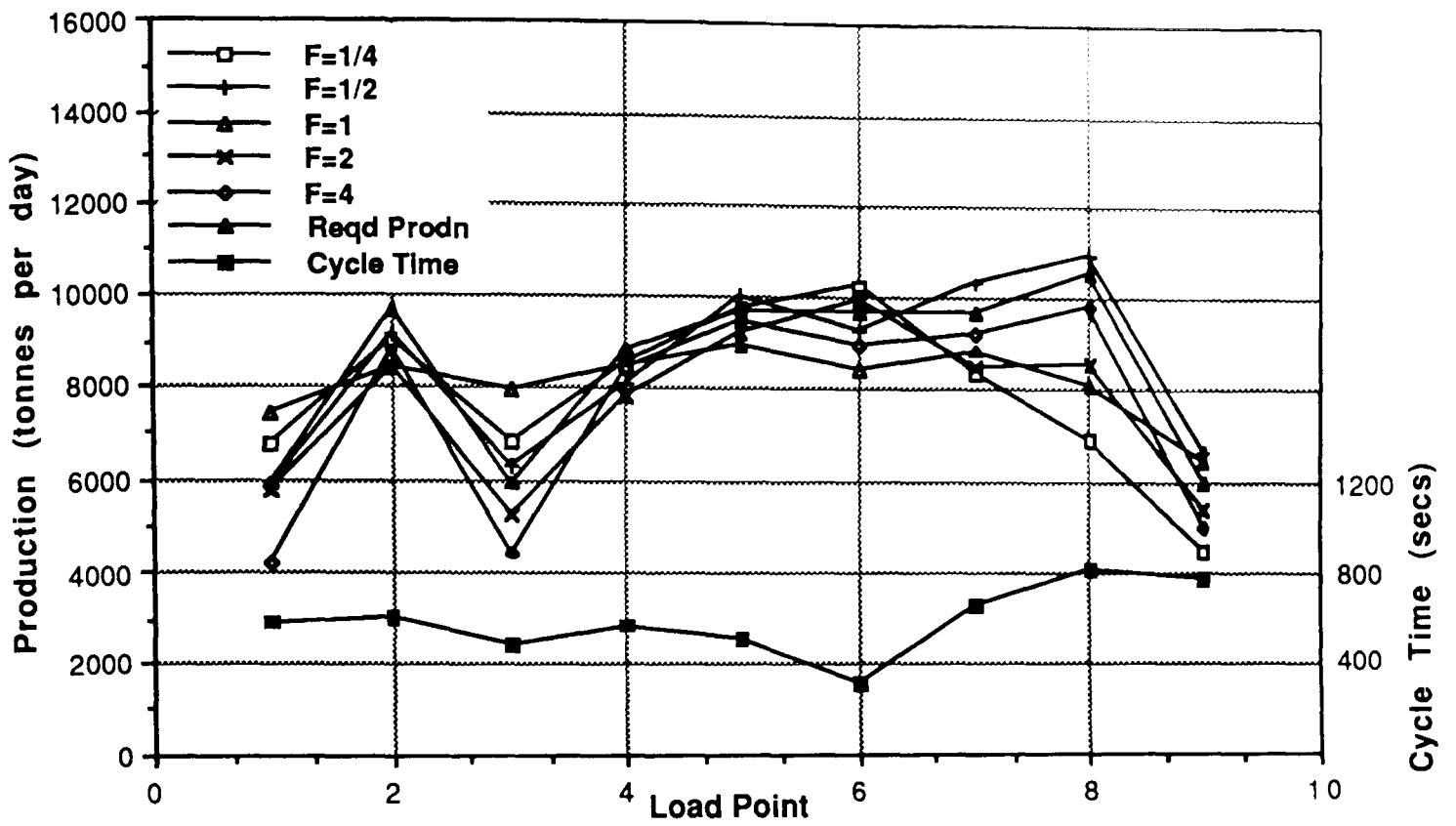


Figure 6.35 Quintette - Simulated Production for 5 Wabco's & 7 Terex's using dynamic allocation over a range of factors (F)

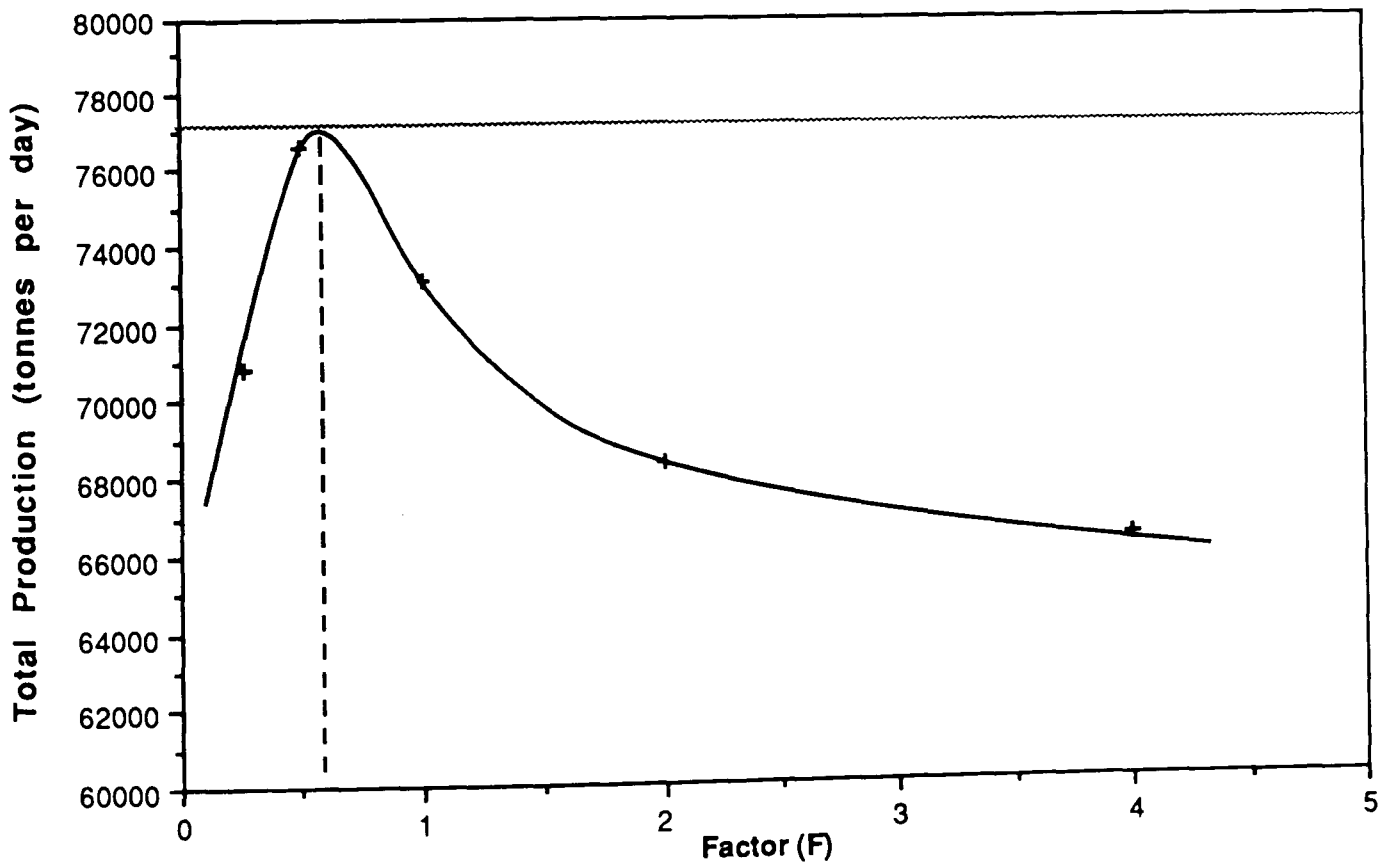


Figure 6.36 Optimum F for 5 Wabco's & 7 Terex's at Quintette

6.5.7 Coal Mountain LP Input

The haulage of waste at the Coal Mountain Mine would, in practice, be a fundamental exercise in dumpsite selection according to proximity and availability at the time. In fact, it is unlikely whether the allocation of more than one dumping area for each loading source would be of any great benefit.

However, the geometry of the mine is ideal for the examination of the effect upon the selection of dispatching policy of having a compact layout with randomly distributed nodes. Also, the shorter distances mean that the fleet, which is similar in size to that which exists at Quintette, has almost twice the productive capacity and hence sufficient flexibility to demonstrate dispatching potential to a greater extent. For these reasons, it was assumed that all 8 waste dump points were being used simultaneously. Figure 6.37 shows all the possible waste haulage routes

6.5.8 Coal Mountain LP Results

Without going into too much detail, the linear programming module produced a 'synergised' route plan as illustrated diagrammatically in Fig 6.38. It is perhaps surprising that, in order to maintain uniformity of dumping, trucks need to be dispatched from L2 all the way across to D4 but the relatively low level of waste production at L1 makes this necessary.

The resulting objective function in terms of number of Caterpillar 777's required is 13.26 and although this exceeds the number in the fleet, it must be remembered that the simulation is a purely hypothetical one (the fixed allocation policy would have produced a fleet requirement of at least 16 trucks (2 per route) and a great deal of enforced shovel idle time).

6.5.9 Coal Mountain Simulation Input

It is apparent from Figure 6.38 that in order to apply the 'synergised' route configuration, four dispatching points must be set up - three to select the routes to be taken from each load point and one to select the return route from D6. In fact, in formulating the simulation network, dispatching from load points L2 and L3 was combined and took place at the latest possible moment i.e. at J13 while J10 was used for similar reasons in the sector containing L1. Dispatching at the latter had to take place in two directions.

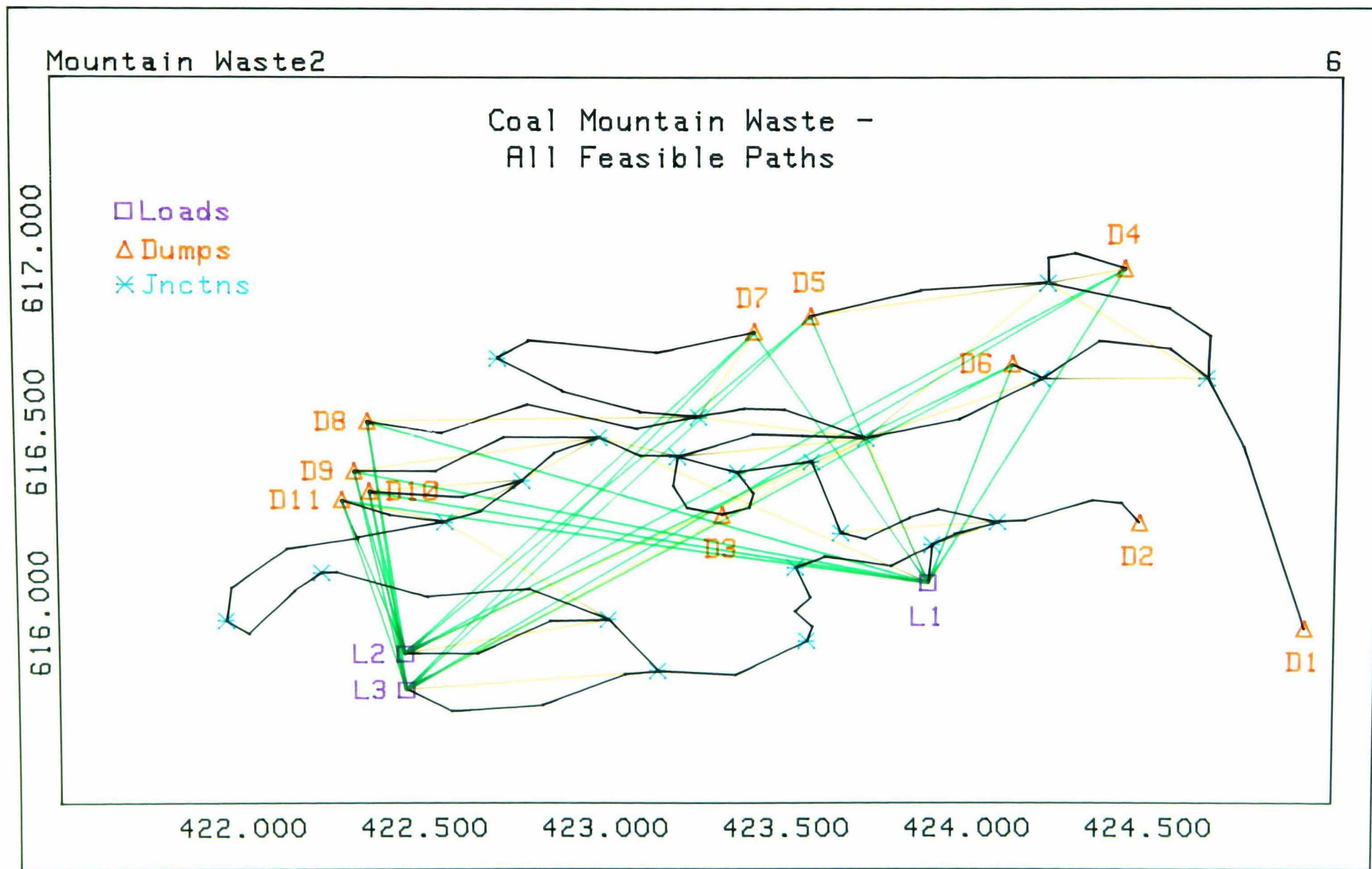


Figure 6.37 Coal Mountain - All Possible Waste Haulage Paths

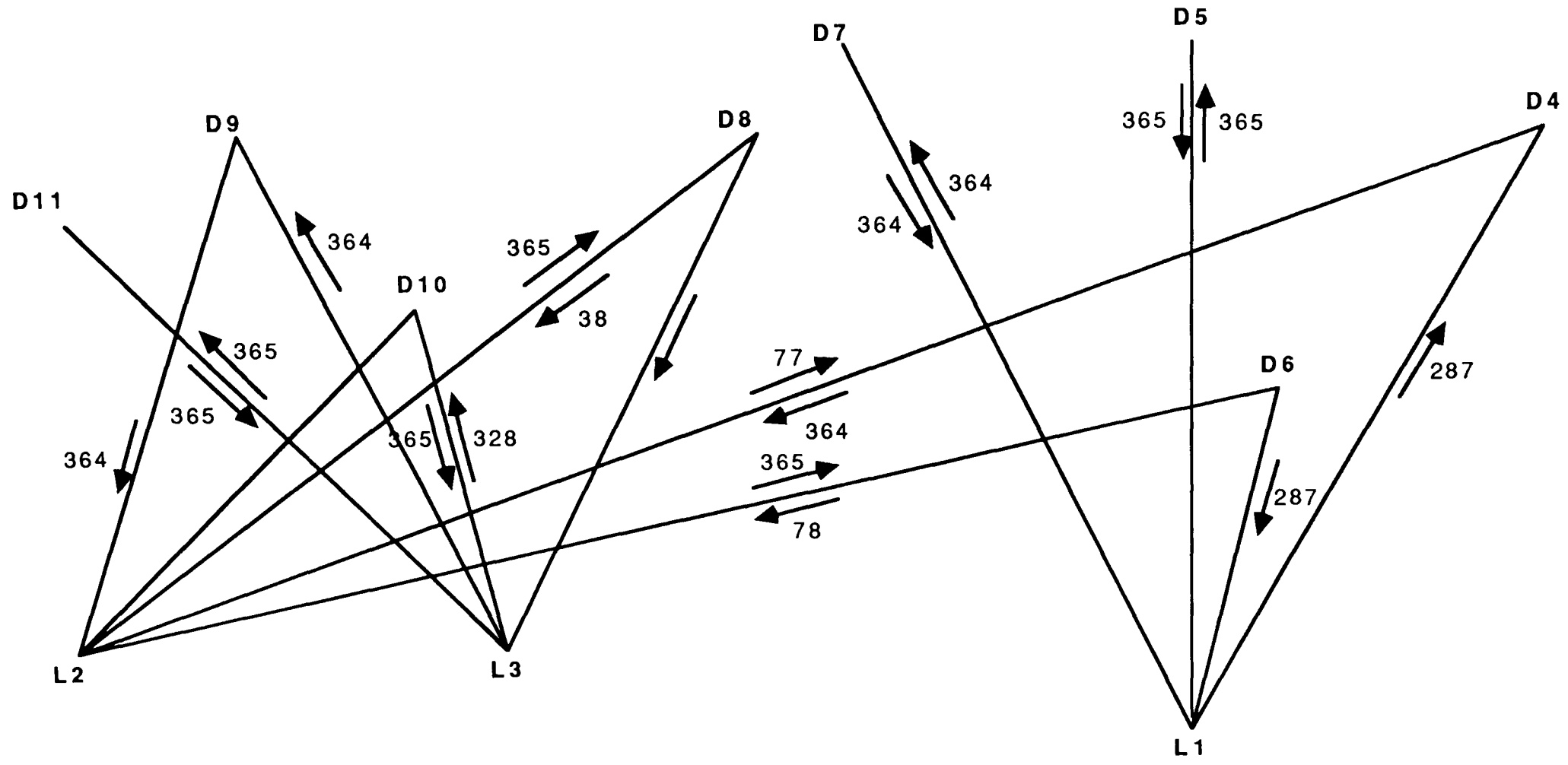


Figure 6.38 Coal Mountain - Diagrammatic Representation of LP Solution to Waste Haulage

The objectives here were to determine the value and effect (if any) of applying the correct cross region damping factor to this particular set of circumstances and the closest value to the objective function of 13.26 trucks which could be achieved.

6.5.10 Coal Mountain Simulation Results

Since it is impossible for the objective function ever to be achieved in reality, Coal Mountain simulation exercises were carried out using 14 trucks to see whether this number would be sufficient. It was assumed that, based on Quintette's results, the increased flexibility brought about by the shorter haul and return distances and smaller trucks would lead to a blending requirement weighting factor close to unity.

The first set of runs were used to determine the cross-region damping factor (33 secs) which resulted in the highest overall production (Fig. 6.39). The second set indicated how close to unity F would actually be if this value was implemented (Fig. 6.40). This second graph also shows that, for a narrow range of F , the required level of overall productivity can actually be achieved using 14 trucks but that, again, individual dump point quotas are not always met.

6.5.11 Test Example Conclusions

The most striking conclusion which can be drawn from the two test examples is that a mine's configuration as well as its size has an affect on its suitability for the implementation of a dispatching system. Both the examples portrayed a degree of undertrucking which enabled operational improvements to be measured in terms of truck productivity or, in other words, the potential to achieve the same objectives using fewer trucks.

With the longitudinal nature of the Quintette mine, linear programming suggests that only a very small improvement (less than 0.02 Terex trucks) can be made by adopting a synergised route plan. Also due to the long haul distances and the use of a relatively small number of large truck units, the dynamic dispatching policy leaves the productivity attainable short of the mine's requirements even with 5 additional Terex's. However, the overall production can be improved by applying a weighting factor to increase the importance of synergy at the expense of blending targets if required.

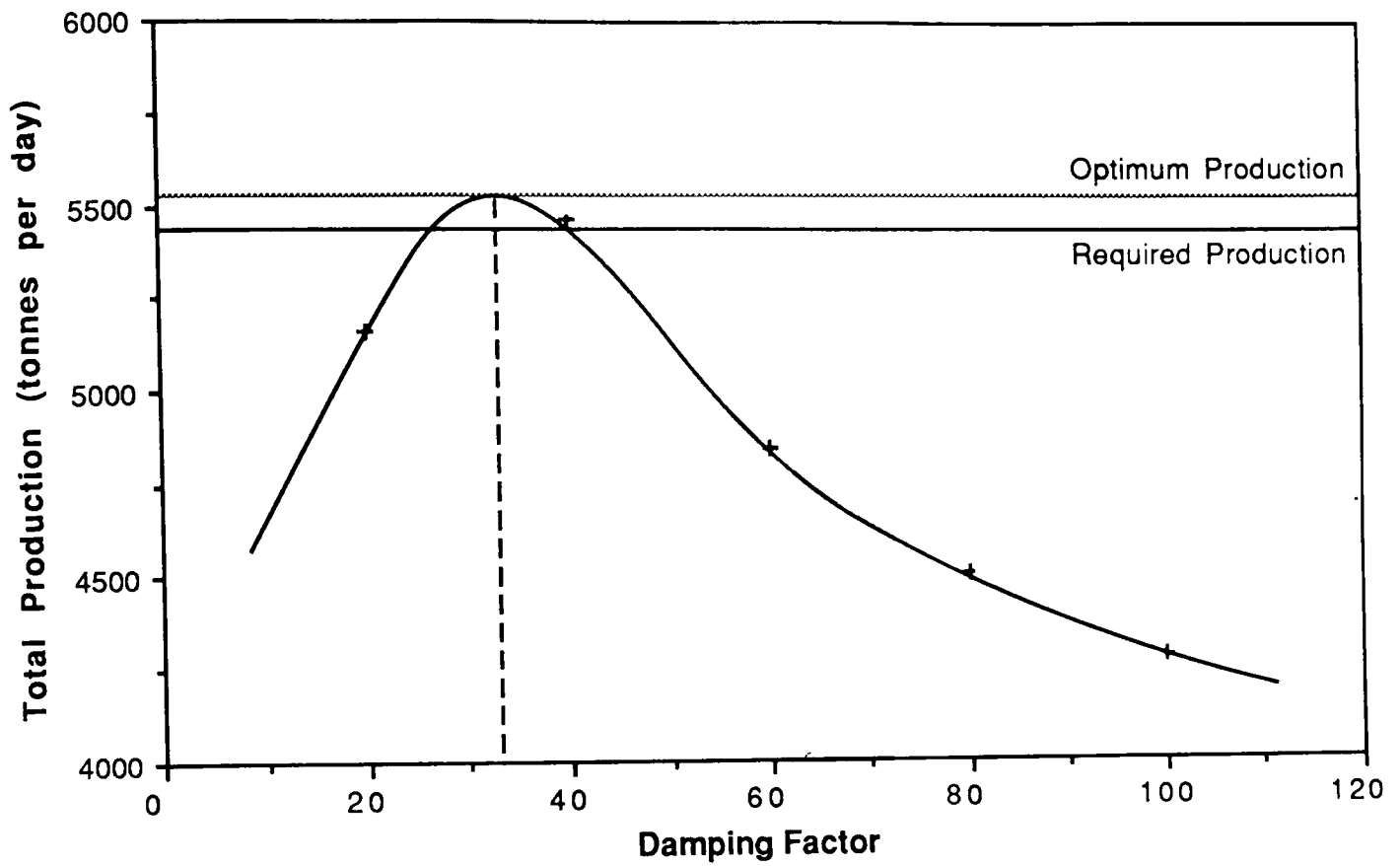


Figure 6.39 Optimum Damping Factor using 14 Caterpillar 777's at Coal Mountain where Factor (F) = 1

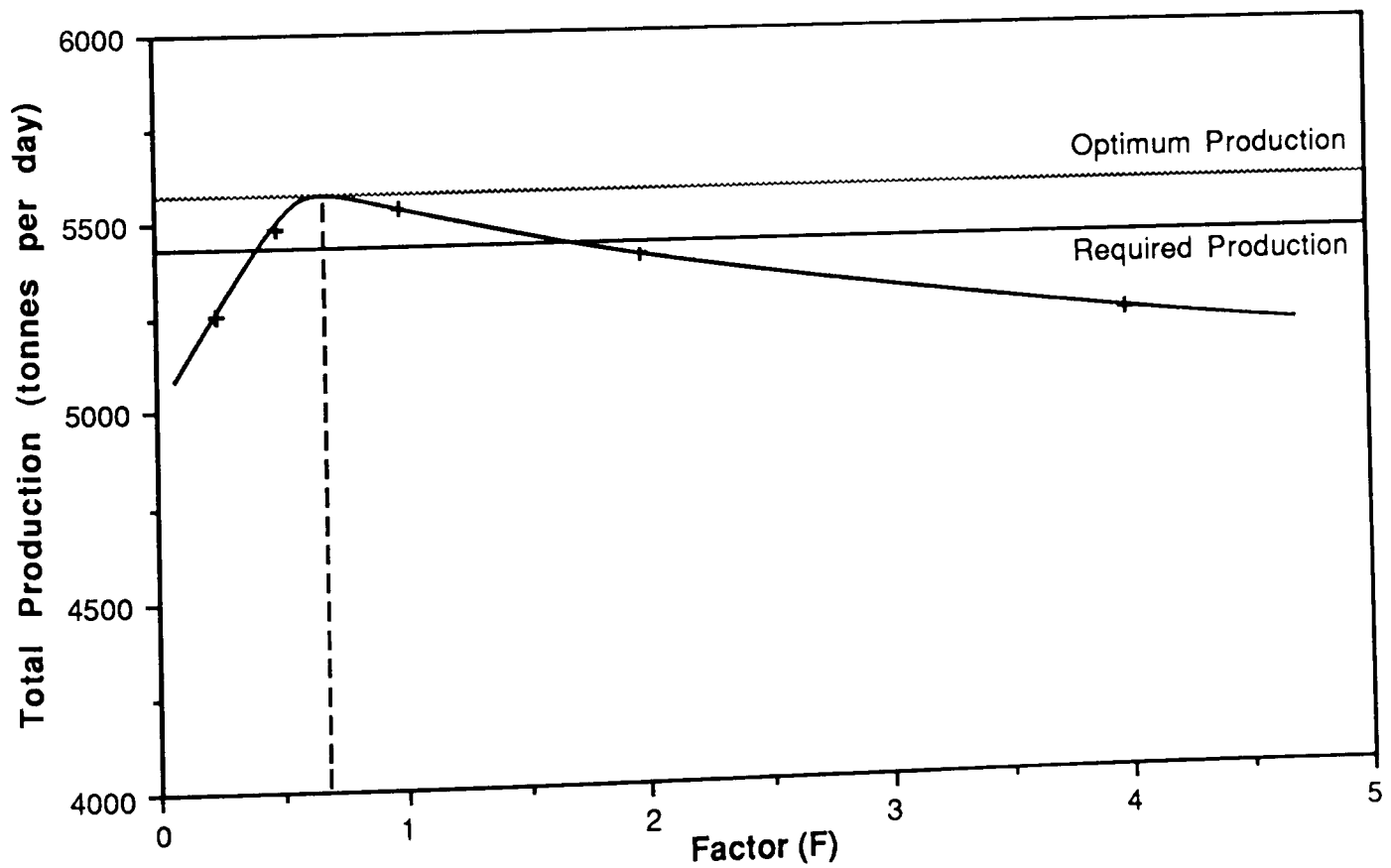


Figure 6.40 Optimum F using 14 Caterpillar 777's at Coal Mountain where Damping Factor = 33 secs

The compact geometry of Coal Mountain, on the other hand, makes the synergised pit configuration far more profitable. Though it is impossible to establish a base case when dispatching is possible from the dumping as well as the loading points, the productivity levels associated with the LP objective function of 13.26 trucks can be achieved, in reality, using just 14. This is by applying a cross region damping factor of 33 seconds to reduce the large degree of haul-time uncertainty associated with longer routes and a weighting factor of 0.67 which again may have some detrimental affect on blending targets. In this case, waste production targets have been given disproportionate importance.

6.6 Conclusions

One of the main aims of the NUmine system was that, by making everything fully compatible, its whole should be of greater value than the sum of its separate components. One of the ways in which two completely distinct modules such as graphics and simulation can be forced to be compatible with each other is to provide the user with a rigid interface which not only carries out the necessary conversion process automatically but also allows him to gain access to both simultaneously. The module described in this chapter does just that.

The advantages can be summarised as follows :-

- ❑ The memory requirement is reduced because the network configurations are always stored in the most efficient manner.
- ❑ graphical formulation of a network is easier to understand than a mathematical description
- ❑ As a result of the above items, the time taken to build a model of the transportation system at anything other than a very simple mine is reduced considerably.

(During testing it took approximately 4 hours from configuring the digitising table to reach the first Quintette simulation run and it is likely that this could be speeded up with practice)

After formulation, the ease of data flow between the various modules used by the selection and scheduling program allowed a large number of tests to be completed in a relatively short space of time. The tests which were carried out on data obtained from two Canadian mines show how the introduction of dispatching policies can affect a mine's productivity and its ability to uphold mathematically derived 'synergised' operations.

The effect of dynamic allocation in particular is to allow total productivity requirements to be met without falling too far out of line with the needs of the mine's constituent load and dump points taken individually. However, the effect can vary according to the geometry of the mine at which it is applied and simulation is one way of measuring and improving the exact characteristics of the policy to be adopted, at little cost.

Chapter 7

Conclusions and Recommendations for Further Work

Chapter 7

Conclusions and Recommendations for Further Work

7.1 Introduction

Increased efficiency in mine transportation has become essential. Mining companies have realised that it is not only minimisation of the fleet size which is important in terms of expenditure, but also the way in which it is utilised. Until recently most planning work has concentrated on fleet size and a number of computer software packages exist for this purpose. However, as computing advances have taken place, the mechanism for predicting and formulating scheduling systems for improved utilisation has emerged and since this can also have a profound effect on the number of trucks required, it should be incorporated in any feasibility study.

Hardware advances during the same period have allowed individual mines and quarries to monitor and control truck movements throughout the course of an operating shift. This has resulted in the successful introduction of truck dispatching where haulage units can be rerouted on a trip by trip basis according to a particular decision-making policy. Over recent years, both the policies used and the controlling mechanisms have become more sophisticated. Consequently, substantially more planning work is necessary much of which involves simulation.

7.1.1 The Need for Rapid Modelling

Simulation modelling, like many other mine planning activities involves uncertainty, and a large amount of both data manipulation and numerical calculation - all features which make this application suitable for a computing environment. Simulation languages and specific modelling packages have been used but have proved insufficient to deal with the complexities of modelling a complete system. They are generally rigid, unintegratable, difficult to learn and use and as a result, have been relatively slow to catch on.

Simulation is time consuming and complicated but can be speeded up using computer aided design techniques in particular automatic network formulation through graphical representation and by the use of a central database to access as much information which already exists about the mine as possible. The NUMine system has been developed with both these points in mind. It is micro-based for flexibility, economy and ease of availability, fully integrated for common utility access and yet modular for multi-level development purposes. It also provides a general purpose user-interface for the benefit of both programmer and planning engineer.

7.1.2 Correspondence with Reality

One of the techniques used for fleet size determination (linear programming) assumes that truck movements can be synergised according to a mathematically derived route network. The concept of dynamic truck allocation allows this to take place to a certain extent but, for any transport option, it is important to try to establish the degree of correspondence with what can actually be achieved. In so doing, the planner not only gets a feel for the range of initial fleet alternatives which then need to be analysed further using simulation, he may also be able to establish the exact dispatching policy suitable for the mine in question.

The latter undoubtedly depends on much more than the network size and node distribution factors which have been discussed, by way of case study, in Chapter 6. There may, for example, be a very tight blending requirement in one mine where, in another, the main operational restrictions are on the total extraction of one area in advance of the one next to it. Planning transportation systems is, in fact, a site-specific problem which means that it can only be carried out within a general-purpose computing environment like the one provided by NUMine if the ease of formulating and using a model of the site using the range of CAD tools available exceeds that of building a stand-alone system for the mine from scratch.

7.2 Recommendations for Further Work

The NUmine system is an on-going project with scope for further research and development in the field of discrete mine transportation. One area which has been neglected somewhat during the three year period since its inception has been the all important link between equipment selection decisions and cost. This, is a general recommendation which must be implemented if the system is to be considered complete. Other more specific enhancements to the modules covered in this thesis are described below.

7.2.1 Modular Simulation

An initial aim of the NUsim module was to be its ability to accept user-defined sub-networks as independent parametric node items. These would be available for use in a similar way to that in which graphical symbols are used in drafting packages. For example, a three way network junction consisting of several nodes and branches configured to represent a particular 'right of way' policy may be pre-defined and stored separately from the network itself. This could then be duplicated by adding a single node with the appropriate name and a list of parameters or simply referenced by the node so that the predefined network itself is used in lieu during a simulation run.

In the event, the module proved to be insufficiently flexible and this facility could not be accommodated without changing the system code and recompiling. However, it may have been possible if a lower level language such as 'C' had been used and would seem to be the ideal mechanism for customising simulation languages to particular applications.

7.2.2 Expert Systems

One area of computing which has received a great deal of interest in the last few years has been artificial intelligence. One branch of this is the expert system which is a mechanism for storing and using knowledge about a particular subject on the computer. The storage facility or 'Knowledge Base' consists of a number of rules which may represent the combined knowledge of one or more experts but may also be derived semi-automatically based on the outcome of a number of exclusive trials.

In Chapter 2 it was mentioned how equipment selection can be based on experience. The role of an expert system in this process may be in evaluating the performance of particular truck or shovel types in the conditions which prevail at the mine or may be in obtaining a realistic estimate of the effect of vehicle interaction and/or breakdowns without carrying out simulation.

The more obvious application in the field of mine transportation however, would be in assisting the fleet controller to make quick decisions when breakdowns, shovel moves or other changes to the network configuration occur. For example,

if a truck breakdown occurs
and its destination is a high priority shovel
and there is another truck in the vicinity assigned to a low priority shovel
then re-route the second truck to the high priority shovel

Computerised dispatching systems have allowed the controllers role to be primarily concerned with management by exception, expert systems may narrow the range of exceptional circumstances further.

7.2.3 Further NUmine Development

From the outset, development of NUmine was intended to be completely in-house. No existing software packages other than those supplied in library form to run the Hewlett Packard Workstation operating system were to be used and the various general purpose utility modules would be added as research requiring their use was carried out. In the event, this philosophy proved to be partially justified since the team working on the system were able to fully understand the workings of the core modules, were able to identify all the features which would be of benefit to their particular areas of study thereby expanding the generality of the overall system, and, if necessary, make changes themselves without having to consult with software suppliers and wait for version updates.

However, a great deal of time and effort was spent developing and subsequently maintaining the database and graphics facilities at a level which could be favourably compared with the ever-improving commercial product. Much of this could be put down to the limitations of the hardware and its in-built library functions which became outdated during the 3-year period since the start of the project.

Recently, the decision was made to transfer the system to the Apple Macintosh II computers which provide an in-built menu-driven windows environment and a number of high quality graphics features. This will provide an opportunity to streamline the system and make use of external software where problems have previously arisen. In addition, at a time when Apple Macintosh is gaining acceptance in industry for its document-processing abilities, the move should increase NUmine's potential as being more than an academic exercise.

Further hardware and system software improvements must be anticipated.

7.3 Conclusions

Throughout this thesis, an attempt has been made to explain the need for a quick and easy method of building models of discrete mine transportation networks to assist the planning engineer in his task of evaluating new systems or changes to existing ones. Attention has been drawn to the fact that a number of optimising techniques now exist which can be used to increase the productive capacity of a working mine and that their effect must be taken into account in the equipment selection process. Also, since they usually differ from one mine to another, the optimising techniques themselves must be analysed against each other with careful regard to the degree of correspondence between the mathematical intents of the policy and what can be achieved in reality.

The general purpose 'total' CAD system described herein provides an ideal environment for rapid transportation network modelling using graphics linked to a central database. As a result, not only is the quantity of alternative network configurations and fleet compositions which can be analysed in a specified time increased, but their accuracy is also improved.

References

REFERENCES

- Adam, N.R., Dogramaci, A., 1979 (a),**
Issues in Simulation Languages; A Brief Overview,
Current Issues in Computer Simulation (Eds. Adam & Dogramaci),
Academic Press, Ch.1, pp 101-107.
- Adam, N.R., Dogramaci, A., 1979 (b),**
Applications of Simulation,
Current Issues in Computer Simulation (Eds. Adam & Dogramaci),
Academic Press, Ch.1, pp 3-13.
- Aiken, G.E., 1980,**
Computerised Trucks: The Future Look of Mine Truck Dispatching,
Proc. Conf. Coal Mine Electro-Technology,
University of West Virginia, pp 12.1-12.9.
- Arnold, M.J., White, J.W., 1983,**
Computer-based Truck Dispatching,
World Mining, April 1983, pp 53-57.
- Atkinson, T., 1982,**
Truck Haulage,
Report, Department of Mining Engineering, University of Nottingham.
- Batchelor, D.H., 1987,**
The Implementation of a Computerised Truck Dispatch System at
Palabora,
Proc. Twentieth Int. Symp. Application of Computers and Mathematics
in the Minerals Industry. Volume 1, SAIMM, 1987, pp 389-401.
- Baron, J., 1977,**
Automatic Truck Dispatching. Lake Jeannine Operations (Paper no. 1),
First Open Pit Operators Conference, C.I.M., May, 1977
- Bauer, A., Calder, P.N., 1972,**
Planning Open Pit Mining Operations Using Simulation,
Proc. Symp. Application of Computers and Operations Research in the
Minerals Industry, 1972, pp 273-278.
- Benson, 1984**
Benson 6301 Digitiser Board Operating Manual
Doc No. 404220350, Nov. 1984.
- Bobillier, P.A., Kahan, B.C., Probst, A.R., 1976,**
Simulation with GPSS and GPSS V,
Prentice-Hall.
- Bohnet, E.L., 1984,**
Mine Equipment Fleet Size Determination,
International Mining, December 1984, pp 13-20.
- Bonates, E., Lizotte, Y., 1988,**
A Combined Approach to Solve Truck Dispatching Problems,
Proc. Symp. Computer Applications in the Mineral Industry
(Eds. Fytas, Collins & Singhal), ISBN, Rotterdam.

- Donner and Moore Associates Inc., 1977,**
Economies of Large Scale Surface Coal Mining Using Simulation Models,
Report for U.S. Energy Research and Development Administration,
March 1977.
- Boulton, C.B., Blair, J.R., 1980,**
A Performance Simulator For Heavy Dump Trucks,
Proceedings of IFAC Symposium on Mining, Mineral and Metal Processing, 1980, Vol. 3, pp 65-77.
- Borland International, 1986,**
Turbo Pascal For The Mac - User's Guide and Reference Manual.
- Brake, D.J., Chatterjee, P.K., 1979,**
Evaluation of Truck Dispatching and Simulation Methods in Large-Scale Open-Pit Operations,
Proc. of 16th Int. Symp. on Application of Computers and Operations Research in the Minerals Industry, 1979, pp 375-383.
- Brown, D.J., Ashton, A.R., Croghan, J.A., Johnson, S.M., 1988,**
Concepts in Computer Aided Mine Design and Planning,
Mining Science and Technology, Vol. 7, 1988, pp 99-119.
- Buxton, J.N., 1968,**
Simulation Programming Languages,
Proc. of I.F.P. Working Conference on Simulation Programming Languages, North Holland Publishing Co.
- Caterpillar Tractor Co., 1968,**
Fundamentals of Earthmoving.
- Caterpillar Tractor Co., 1980,**
Caterpillar Performance Handbook.
- Caterpillar Tractor Co., 1984,**
Travel Time and Earthmoving Production Vehicle Simulation Program.
- Chatterjee, P.K., Brake, D.J., 1981,**
Truck Dispatching and Simulation Methods in Open-Pit Operations,
C.I.M. Bulletin, Nov. 1981, Vol. 74, No. 835, pp. 102-107
- Clarke, M.P., 1989,**
Decision Support System for Strip Mine Design,
PhD. Thesis, in preparation, Department of Mining Engineering,
University of Nottingham.
- Clevenger, J.G., 1983,**
Dispatch Reduces Mining Equipment Requirements,
Mining Engineering, Sept. 1983, 1277-1280.
- Crawford, J.T., 1979,**
Shovel and Haulage Truck Evaluation,
Open Pit Mine Planning and Design (Editors-Crawford and Hustrulid).
A.I.M.E., New York, 1979, pp 258-270.
- Croghan, J.A., 1989,**
Recursive Computer Modelling for Mine Design,

PhD. Thesis, Department of Mining Engineering,
University of Nottingham.

- Cross, B.K., Williamson, G.B., 1969,**
Digital Simulation of an Open-Pit Truck Haulage System,
A Decade of Digital Computing in the Mineral Industry (Ed. Weiss, A.),
A.I.M.E., 1969, pp 385-400.
- Crosson, C.C., Tonking, M.J.H., Moffat, W.G., 1977,**
Palabora System of Truck Control,
Mining Magazine, Feb. 1977, pp 74-82.
- Farrell, T.R., 1988,**
Computerised Truck Dispatching at Quintette Coal Limited,
Proc. Symp. Computer Applications in the Mineral Industry
(Eds. Fytas, Collins & Singhal), ISBN, Rotterdam.
- Ferguson, G.A., 1985,**
Computer-Based Mining Design,
Mining Magazine, May 1985, pp 403-408.
- Fitzgerald, B.W., Blair, J.R., 1977,**
A Computer Study of a Single Track Iron-ore Railway Operation,
Proc. 15th Int. Symp. Application of Computers and Operations
Research in the Minerals Industry, pp 39-48.
- Fytas, K., Calder, P.N., 1984,**
Optimisation of Open Pit Loading and Hauling Systems,
Symposium on Surface Mining, Hydrology, Sedimentology and
Reclamation, University of Kentucky, December 1984, pp 69-75.
- Fytas, K., Wilson, E.P., Singhal, R.K., 1985,**
Comparison of Fortran and GPSS Languages in the Simulation of a
Shovel-Truck Open Pit Operation,
Report, Laval University, Quebec, Canada.
- Graham, R., 1983,**
Practical Pascal for Microcomputers,
John Wiley & Sons.
- Hancock, M.W., Lyons, D.N.G., 1984,**
Operational Research in the Planning of Underground Transport,
Proc. Int. Symp. Application of Computers and Operations Research in
the Minerals Industry, pp 389-399.
- Hanson, B.D, Selim, A.A., 1975,**
Probabilistic Simulation of Underground Production System,
Transactions, Society of Mining Engineers, Vol. 258, pp 19-24.
- Hastings, N.A.J., Peacock, J.B., 1975,**
Statistical Distributions,
Butterworth.
- Hauk, R.F., 1979,**
Computer-Controlled Truck Dispatching in Open-Pit Mines,
Computer Methods for the 80's in the Minerals Industry,
A.I.M.E., pp 735-742.

- Haycocks, C., Kumar, A., Unal, A., Osei-Agyemana, S., 1984,**
Interactive Simulation for Room and Pillar Mines,
Proc. Int. Symp. Application of Computers and Operations Research in
the Minerals Industry, pp 584-590.
- Hempenstall, J., Hill, R., 1980,**
Bougainville Increases Efficiency with Computers,
World Mining, Feb. 1980, pp 54-56.
- Hewlett-Packard, 1985 (a),**
Pascal 3.1 Graphics Techniques for the HP 9000 Series 200/300
Computers.
- Hewlett-Packard, 1985 (b),**
Pascal 3.1 Workstation System for the HP 9000 Series 200/300
Computers.
- Himebaugh, 1980,**
Computer-Based Truck Dispatching in the Tyrone Mine,
Mining Congress Journal, Vol. 66, No. 11, pp 16-21.
- Holzmann, E.G., Haefner, K.B., 1978,**
Modelling and Control of Robotic Mine Haulage Systems
Annual Simulation Symposium.
- Jolley, D.R., 1970,**
Nature & Usefulness of Simulation as it Applies to Equipment Selection
at an Open-Pit Mine,
C.I.M. Bulletin, October 1970, pp 1176-1180.
- Katsabanis, P., Michalopoulos, N.,
Panagolopoulos, C., Economopoulos, J., 1984,**
Simulation of Room and Pillar Mining Systems: an Application at the
Bauxite Mines of Parnassos, Greece),
Proc. Int. Symp. Application of Computers and Operations Research in
the Minerals Industry, pp 591-597
- Kim, Y.C., Ibarra, M.A., 1981,**
Truck Dispatching by Computer Simulation,
Bulk Solids Handling, Feb. 1981, Vol. 1, No. 1, pp 137-147.
- Korn, G.A., Wait, J.V., 1978,**
Digital Continuous-System Simulation,
Prentice-Hall.
- Krakiwski, E.J., Karimi, H.A., Kempe, A.B.,
Srajer, V., Lockhart, T., Wade, R., 1988,**
An Automatic Vehicle Status, Location, and Allocation System for
Surface Mines,
Proc. Symp. Computer Applications in the Mineral Industry
(Eds. Fytas, Collins & Singhal), ISBN, Rotterdam.
- Kullberg, H.F., Wright, K.W., 1968,**
Viewpoints on Haulage - Truck Size Selection,
Mining Congress Journal, May 1968, pp 50-51.
- Lehman, R.S., 1977,**
Computer Simulation and Modelling,

John Wiley & Sons.

- Lizotte, Y., Bonates, E., Leclerc, A., 1987,**
Design and Implementation of a Semi-automated Truck/Shovel
Dispatching System,
Proc. of 20th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, SAIMM, 1987, pp 377-387.
- Lizotte, Y., Scoble, M., Bonates, E., 1985,**
Application of Simulation to Assess Truck/Shovel Dispatching Policies,
Presented at Mining Equipment Symposium, University of Calgary,
Nov. 1985.
- Manula, A.E.K., Ramani, R.V., 1977,**
Application of a Total System Surface Mine Simulator to Coal
Stripping,
Pennsylvania State University, October 1977.
- Manula, A.E.K., Ramani, R.V., Falkie, T.V., 1975,**
A General Purpose Systems Simulator For Coal Mining
Mining Congress Journal, March 1975.
- Markowitz, H.M., 1979,**
Simscrip; Past, Present and Some Thoughts about the Future,
Current Issues in Computer Simulation (Eds. Adam & Dogramaci),
Academic Press, Ch.3, pp 27-59.
- Medland, A.J., Mullineux, G., 1988,**
Principles of CAD,
Kogan Page Ltd.
- Meyer, H.I., 1979,**
Truck Allocation to Shovels in an Open-Pit Mine - A Case Study on the
Initial Attempt,
Computer Methods for the 80's in the Minerals Industry,
A.I.M.E., 1979, pp 637-646.
- Morgan, W.C., Peterson, L.L., 1968**
Determining Shovel-Truck Productivity,
Mining Engineering, Dec. 1968, pp 77-80.
- Mueller, E.R., 1977,**
Simplified Dispatching Board Boosts Truck Productivity at Cyprus
Pima,
Mining Engineering, August 1977, pp 40-43.
- Naplatanov, N.D., Sgurev, V.S., Petrov, P.A., 1977,**
Truck Control at Medet,
Mining Magazine, July 1977, pp 12-18.
- Nenonen, L.K., 1982,**
Interactive Computer Modelling of Truck Haulage Systems,
C.I.M. Bulletin, Vol. 75, No. 847, Nov. 1982, pp 84-89.
- Newman, W.M., Sproull, R.F., 1981**
Principles of Interactive Computer Graphics,
McGraw-Hill.

- Jewman, W., Stephens, N., Sweetman, D., 1985,**
A Window Manager with a Modular User Interface,
People and Computers: Designing the Interface,
Cambridge University Press, Ch. 38, pp 415-426.
- O'Neil, T.J., Manula, C.B., 1967,**
Computer Simulation of Materials Handling in Open Pit Mining,
Transactions - Society of Mining Engineers, June 1967, pp 137-146.
- Oberg, B., 1980,**
Computer-Controlled Ore Transportation at the Kiirunavaara Iron Mine
in Kiruna, Sweden,
ASEA Journal, Vol. 53, No. 3, pp 35-40.
- Pincock, Allen & Holt inc., 1982a,**
Truck Dispatch by Computer.
C.I.M. Bulletin, Vol. 75 No. 844, Aug. 1982, pp 42-43
- Pincock, Allen & Holt inc., 1982b,**
Computer-directed Dispatch System and Traffic Simulator for Open
Pits.
Mining Magazine, May 1982, pp 407-409
- Press, W.H., Flannery, B.P.,
Teukolsky, S.A., Vetterling, W.T., 1986**
Numerical Recipes,
Cambridge University Press.
- Price, C.M., Szabo, L.J., 1970,**
Planning Conveyor Systems by Computer,
The Mining Engineer, Nov. 1970, pp 85-99.
- Pritsker, A.A.B., 1979,**
GASP; Present Status and Future Prospects,
Current Issues in Computer Simulation (Eds. Adam & Dogramaci),
Academic Press, Ch.4, pp 61-69.
- Pritsker, A.A.B., Pegden, C.D., 1979,**
Introduction to Simulation and SLAM,
John Wiley & Sons.
- Reid, P., 1984,**
Work Station Design, Activities and Display Techniques.
Fundamentals of Human-Computer Interaction (Editor - Monk, A.),
Academic Press, 1984, Ch. 8, pp 107-126.
- Rossouw, P.A., 1986,**
Truck Allocation and Equipment Control as Applied at the Sishen Iron
Ore Mine,
The Planning and Operation of Open-Pit and Strip Mines,
(Ed. Deetlefs, J.P.), SAIMM, 1986, pp 319-324.
- Ryder, P., Szabo, L.J., 1976,**
Recent Developments in the Design and Control of Underground
Conveyor Belt Bunker Coal Clearance Systems,
Proc. of 14th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, 1976, pp 140-145.

- Ryder, J.A., 1976,**
Transim II - A New Generalised Underground Transport Simulator,
Proc. of 14th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, 1976, pp 105-111.
- Sadler, W.M., 1988,**
Practical Truck Dispatch - A Micro Computer Based Approach,
Proc. Symp. Computer Applications in the Mineral Industry
(Eds. Fytas, Collins & Singhal), ISBN, Rotterdam.
- Sassos, M.P., 1984,**
Automatic Dispatch System Improves Production, Optimises Equipment
Selection,
Engineering and Mining Journal, October, 1984.
- Schneider, G.M., Weingart, S.W., Perlman, D.M., 1982,**
An introduction to Programming and Problem Solving with Pascal
(second edition), John Wiley & Sons.
- Singhal, R.J., Fytas, K., 1985,**
Off-Highway Truck Selection in Surface Coal Mining,
Symposium on Surface Mining, Hydrology, Sedimentology and
Reclamation, University of Kentucky, December 1985, pp 61-67.
- Singhal, R.J., Fytas, K., 1986,**
Optimization of Loading and Hauling Equipment in Surface Mining,
Report: Canmet Coal Research Laboratory, Calgary, Canada.
- Srajer, V., 1987,**
Optimization for Maximum Production of Truck/Shovel Mining System,
Ph.D. Thesis, Department of Mining Engineering, University of
Nottingham.
- Steiker, A.B., 1982,**
Simulation of an Underground Haulage System,
Proc. of 17th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, 1982, pp 599-613.
- Sturgul, J.R., 1987,**
Should the Mining Engineer Learn a Special Computer Simulation
Language?,
Report, South Australia Institute of Technology.
- Sturgul, J.R., Singhal, R.K., 1988,**
Using the Personal Computer to Simulate Mining Operations,
Proc. Symp. Computer Applications in the Mineral Industry
(Eds. Fytas, Collins & Singhal), ISBN, Rotterdam, pp 439-442.
- Suboleski, S.C., Lucas, J.R., 1968,**
Simulation of Room and Pillar Face Mining Systems,
A Decade of Digital computing in the Mineral Industry (Ed Weiss),
A.I.M.E., 1969, pp 385-400.
- Sutherland, I.E., Hodgman, G.W., 1974,**
Reentrant Polygon Clipping,
Communications of the Association for Computing Machinery,
January 1974, Vol. 17, No. 1.

- Galbot, K., 1977,**
Simulation of Conveyor Belt Networks in Coal Mines,
Proc. of 15th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, 1977, pp 297-304.
- Gouwen, F.H. Joughin, N.C., 1972,**
The Simulation of Underground Stopping and Transport Operations in
Gold Mining,
Proc. of Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, 1972, pp 231-236.
- Tu, J.H., Hucka, V.J., 1985,**
Analysis of Open-Pit Truck Haulage System by Use of a Computer
Model,
C.I.M. Bulletin, July 1985, Vol. 78, No. 879, pp 53-59.
- Vagenas, N., 1988,**
Profitas; A CAD Simulator for Automated, Guided LHD in
Underground Mining,
Proc. Symp. Computer Applications in the Mineral Industry
(Eds. Fytas, Collins & Singhal), ISBN, Rotterdam, pp 413-422.
- White, J.A., Schmidt, J.W., Bennett, G.K., 1975,**
Analysis of Queuing Systems,
Academic Press Inc.
- White, J.W., Arnold, M.J., Clevenger, J.G., 1982,**
Automated Open-Pit Truck Dispatching,
Engineering and Mining Journal, Vol. 183, No. 6, pp 76-84.
- White, J.W., Olson, J.P., 1986,**
Computer-Based Dispatching in Mines with Concurrent Operating
Objectives,
Mining Engineering, November 1986, pp 1045-1054.
- White, J.W., Zoschke, L.T., 1987,**
The Development of a Computerised truck Dispatching System,
Mining Magazine, December 1987, pp 558-562.
- Wilke, F.L., 1973,**
Simulation Studies on Computer Controlld Traffic Underground in
Large Coal Mines,
C.I.M. Special, Vol.12, pp 344-350
- Wilke, F.L., Heck, K., 1982,**
Simulation Studies on Truck Dispatching,
Proc. of 17th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, 1982, pp 620-626.
- Wilson, J.W., 1984,**
Simulation of Ore Transport on Surface Rail at Impala Platinum Ltd.,
Proc. of 18th Int. Symp. on Application of Computers and Operations
Research in the Minerals Industry, pp 414-418.
- Zhongzhou, L., 1984,**
Computer Simulation of Railway Systemsd in Surface Mines,
Proc. of 18th int. symp. on Application of Computers and Operations
Research in the Minerals Industry, pp 419-425.

Appendices

Appendix I**NUsim Functions**

Variable	Type	Meaning
TIME	real	the current time.
ATRIB[i]	array[1..20] of real	an entity attribute.
GLOBAL[i]	array[1-100] of real	a global variable.
INQ(n)	integer function	returns number of entities currently residing in queue node n.
INA(a)	integer function	returns number of entities currently being processed by activity a.
CPA(a)	integer function	returns number of entities which have completed activity a.
LASTR(n)	real function	returns last release time of node/activity n.
NEXTR(n)	real function	used to schedule a future event at the time of the next release of node/activity n.
UNRSC(r)	integer function	returns the current number of available units of resource r
SWST(s)	boolean function	returns true if switch s is currently open - false if not.
UFN(f)	real function	returns a real value from user defined function f.
RND(s)	random function	returns a real value between 0 and 10000 using random number stream s.
NRM(m,d,s)	random function	returns a real value from a normal distribution with mean m, standard deviation d, and using random number stream s.
TRG(m,l,u,s)	random function	returns a real value from a triangular distribution with mean m, lower limit l, upper limit u and using random number stream s.
UFM(l,u,s)	random function	returns a real value from a uniform distribution with lower limit l, upper limit u and using random number stream s.
EXD(m,s)	random function	returns a real value from an exponential distribution with mean m and using random number stream s.

Variable	Type	Meaning
LNM(m,d,s)	random function	returns a real value from a lognormal distribution with mean m, standard deviation d, and using random number stream s.
WBL(α,β,γ,s)	random function	returns a real value from a weibull distribution with shape parameter α , scale parameter β , location parameter γ , and using random number stream s.
BET(α,β,s)	random function	returns a real value from a beta distribution with shape parameters α and β , and using random number stream s.
ERL(α,β,s)	random function	returns a real value from an erlang distribution with shape parameter α , scale parameter β , and using random number stream s.
PSN(m,s)	random function	returns a real value from a poisson distribution with mean s, and using random number stream s.
GAM(α,β,s)	random function	returns a real value from a gamma distribution with shape parameter α , scale parameter β , and using random number stream s.
MCO(c,s)	random function	returns a real value from predefined cumulative distribution curve c using random number stream s.

All the standard PASCAL functions are also available.
 In addition, variables can be defined either as individual items e.g. global[9]
 or as combinations of items e.g. atrib[2]+nrm(10,2,1).
 These can be referred to, thereafter, by name. (see variable definition)

integer parameters:

















- i integer index
- s random number stream
- c curve index
- n node index
- a activity index
- r resource index
- s switch index
- f user defined function index

real parameters:

- m mean
- d standard deviation
- l lower limit
- u upper limit
- α shape parameter
- β scale parameter
- γ location parameter

Appendix II

NUsim Symbols

	Activity		Collect		Alter
	Branch		Stats		Wait
	Enter		Continue		Off
	Queue		Decision		On
	Remove		Match		Free
	Assign				