RISK ANALYSIS IN CONSTRUCTION

by

Saleh Bashir Awad Gaderbouh

A thesis submitted for the degree of

Master of Science

Department of Civil Engineering

University of Glasgow

January 1993

© Gaderbouh January 1993
ACKNOWLEDGEMENTS

This work reported in this thesis was carried out in the Department of Civil Engineering University of Glasgow.

The author wishes to express his special thanks and appreciation to his supervisor Dr. K. J. Williams, for his valuable guidance and constant encouragement throughout this work. Thanks should also go to him for undertaking much of the 'graphic' work in PERTRA1 and PETRA2.

The author is also grateful to professor Muir Wood (Cormack professor and Head of the Department) and professor D. Green (previous Head of the Department) for allowing the use of the Department facilities.

The author is grateful to his fellow researchers Dr. Zhang, Dr. H. Jasem, Dr. Mesavi, Dr. O. Famiyesin, M. Abdulkader, A. Bensalem, K. Belkhair, A. Hamidon, George Frangopoulos, Giovainni Schlipoti, P. Addison and X. Leng for their helpful discussions and constant encouragement.

The author is grateful to Ras Lanuf Oil and Gas Processing Company and to Jawaby Oil Company for the financial support during the course.

Finally my thanks reserved for my Mother, my Father, my Sister my Brothers, my Wife Taisser and my Daughters Hind and Orayda for their patience and constant encouragement throughout my studies.
SUMMARY

The work reported in this thesis centres around the development of a risk analysis computer software package which operates in conjunction with PERTMASTER ADVANCE, a widely-used commercial project planning software package.

The risk analysis technique used herein is the so-called 'Monte Carlo simulation technique' whereby a number of project simulations, each of which makes use of a set of project data values (such as task durations) chosen randomly from within specified probability distributions, is carried out with a view to producing a particular solution (such as the project duration) defined by a single probability distribution. In such a way, the degree of uncertainty associated with the particular solution, which results from the individual degrees of uncertainty associated with each of the data values on which such solution is based, can be determined.

The software package comprises the following programs:

PERTRA1 — an interactive program which abstracts specific project data from a datafile generated by PERTMASTER ADVANCE.

PERTRA2 — an interactive program which facilitates the keyboard input of task duration probability distribution data.
PERTRA3 — a 'number-crunching' program which carries out the risk analysis on the basis of data generated by PERTRA1 and PERTRA2. The results of the analysis are written to a user-specified datafile in such a form as to enable subsequent graphical presentation thereof by widely available commercial 'graphics' software packages.

All three programs are written in FORTRAN 77 and can be run on any IBM-compatible personal computer using the MS-DOS version 3.3 or higher.
TABLE OF CONTENTS

CONTENTS

<table>
<thead>
<tr>
<th>CONTENTS</th>
<th>PAGE NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>i</td>
</tr>
<tr>
<td>SUMMARY</td>
<td>ii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xiii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xviii</td>
</tr>
</tbody>
</table>

CHAPTER 1: INTRODUCTION TO RISK ANALYSIS       1

1.1 Introduction                               1
1.2 Sources of Uncertainties                  3
    1.2.1 Accuracy of the Data                  3
    1.2.2 Inherent Uncertainties               5
1.3 Risk Analysis                              6
    1.3.1 Risk Analysis Techniques             7
        1.3.1.1 Sensitivity Analysis            7
        1.3.1.2 Probability Analysis           9

CHAPTER 2: NETWORK ANALYSIS                   12

2.1 Historical Background to Critical Path Analysis 12
2.2 The Network Model                          14
    2.2.1 Activity on Arrow Networks           14
3.3.1.7 Delay in Receiving Payments from clients 47
3.3.1.8 Delay in Buying Labour, Plant Hires, Material Supplies and Subcontractors 49
3.3.1.9 Inflation 49
3.4 Internal Rate of Return (IRR) 52

CHAPTER 4: THE ANALYSIS OF UNCERTAINTIES 54
4.1 The Two Approaches 54
  4.1.1 The Statistical Method 55
    4.1.1.1 Interfering Critical Paths 58
  4.1.2 Simulation Techniques 61
4.2 Monte Carlo Simulation 62
4.3 The Difficulties of Simulation 63
  4.3.1 The Necessary Number of Simulations 64
  4.3.2 Data Storage 64
  4.3.3 Distribution Calculation 65
4.4 Triangular Probability Distribution 67
  4.4.1 Calculation of the Parameters 67
4.5 Random Selection of Task Duration 71
  4.5.1 Equiprobability Bands 72
  4.5.2 Direct Calculation Method 73
  4.5.3 Comparison 74
  4.5.4 Selection of Task Duration Using the Direct Calculation Method 75
CHAPTER 5: THE COMPUTER PROGRAMS

5.1 Introduction

5.2 PERTRA1: Data reading and initialisation

5.2.1 Subroutine HEADER

5.2.2 Subroutine READEXP

5.2.2.1 Subroutine ABTASK

5.2.2.2 Subroutine ABTRD

5.2.2.3 Subroutine ABLINK

5.2.2.4 Subroutine ABRES

5.2.2.5 Subroutine ABSUBRES

5.2.2.6 Subroutine ABCAL

5.2.3 Subroutine DIMCHK

5.2.4 Subroutine CALWKS

5.2.5 Subroutine TCOSTS

5.2.6 Subroutine RESCOST

5.2.7 Subroutine LSORT

5.2.8 Subroutine QUIT

5.3 PERTRA2: Interactive input of task duration Data

5.3.1 Subroutine HEADER

5.3.2 Subroutines FDISPLAY, FILTER, TDISPLAY

5.3.2.1 Subroutine FDISPLAY

5.3.2.2 Subroutine FILTER

5.3.2.3 Subroutine TDISPLAY

5.3.3 Subroutine QDISPLAY

5.3.4 Input of Task Duration Data
5.3.5 PERTRA2 Output 122
5.4 PERTRA3: Risk Analysis and Cash Flow Calculations 123
   5.4.1 Calendar Subroutines and Functions 127
      5.4.2.1 Subroutine CALEND 127
      5.4.2.2 Function IDOTW 129
   5.4.2 Subroutine ANALYS 129
      5.4.2.1 Function LNKDRN 135
      5.4.2.2 Function ISADD 135
      5.4.2.3 Function ISDEDC 135
   5.4.3 Subroutine TRIDIS 136
   5.4.4 Subroutine DSELECT 137
      5.4.4.1 Function RAND 137
   5.4.5 Subroutine PCOSTS 138
   5.4.6 Subroutine PSORT 140
   5.4.7 Subroutine PRSTAT 140
   5.4.8 Subroutine CASTAT 140
   5.4.9 Subroutine RRSTAT 140
   5.4.10 Subroutine RRSORT 141
   5.4.11 Subroutine FREQUA 141
   5.4.12 Subroutine FRQIRR 142
   5.4.13 Subroutine FRQCAP 142

CHAPTER 6: PROGRAM OPERATION 143
6.1 PERTRA1 143
   6.1.1 Screen Displays 143
6.1.1.1 Data Input Line 144
6.1.1.2 Screen Display Table 145
6.1.1.3 Message Line 145

6.1.2 Operational Details 146
6.1.2.1 Initialisation 146
6.1.2.2 Reading of Data File 148
6.1.2.3 Input of Working Week Data 150
6.1.2.4 Input of Cost Component Data 152
6.1.2.5 Generation of Data Files and Program Termination 156

6.2 PERTRA2 157
6.2.1 Screen Displays 157
6.2.2 Operational Details 157
6.2.2.1 Program Introduction 157
6.2.2.2 Reading Task Duration Data 160
6.2.2.3 Data Input 160
6.2.2.4 Input of Tasks with VARIABLE Durations 162
6.2.2.4.1 Filtration Parameters 163
6.2.2.4.2 Filtration Parameters Logic 168
6.2.2.4.3 Input of Optimistic and Pessimistic Durations Factors 172
6.2.2.5 Input of Tasks with INVARIABLE Durations 175
6.2.2.6 Updating of Task Data File and Program Termination 177
6.3 PERTRA3

6.3.1 Program Operation

6.3.2 Data Input

6.3.2.1 Random Number Series Type

6.3.2.2 Number of Simulations

6.3.3 Data Output

6.3.3.1 One Simulation Run Data Output

6.3.3.2 Multiple Simulation Runs Output

CHAPTER 7: CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusion

6.2 Recommendations

REFERENCES

APPENDICES

Appendix A Contents of PMA—Generated Export File

Appendix B Samples of Data Files Generated by PERTRA1

Appendix B.1 Sample of the General Data Written to the File FLNAME.GEN

Appendix B.2 Sample of the Task Data Written to the File FLNAME.TSK
<table>
<thead>
<tr>
<th>Appendix B.3</th>
<th>Sample of the Link Data Written to the File FLNAME.LNK</th>
<th>202</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix B.4</td>
<td>Sample of the Cost Data Written to the File FLNAME.CST</td>
<td>203</td>
</tr>
<tr>
<td>Appendix C</td>
<td>Program PERTRA1 and Subroutines</td>
<td>204</td>
</tr>
<tr>
<td>Appendix C.1</td>
<td>Program PERTRA1</td>
<td>204</td>
</tr>
<tr>
<td>Appendix C.2</td>
<td>Subroutine HEADER</td>
<td>208</td>
</tr>
<tr>
<td>Appendix C.3</td>
<td>Subroutine DIMCHK</td>
<td>212</td>
</tr>
<tr>
<td>Appendix C.4</td>
<td>Subroutine READEXP</td>
<td>215</td>
</tr>
<tr>
<td>Appendix C.5</td>
<td>Subroutine RECCOM</td>
<td>221</td>
</tr>
<tr>
<td>Appendix C.6</td>
<td>Subroutine ABTASK</td>
<td>222</td>
</tr>
<tr>
<td>Appendix C.7</td>
<td>Subroutine ABTRD</td>
<td>224</td>
</tr>
<tr>
<td>Appendix C.8</td>
<td>Subroutine ABLINK</td>
<td>226</td>
</tr>
<tr>
<td>Appendix C.9</td>
<td>Subroutine ABRES</td>
<td>228</td>
</tr>
<tr>
<td>Appendix C.10</td>
<td>Subroutine ABSUBRES</td>
<td>229</td>
</tr>
<tr>
<td>Appendix C.11</td>
<td>Subroutine ABCAL</td>
<td>230</td>
</tr>
<tr>
<td>Appendix C.12</td>
<td>Subroutine CALWKS</td>
<td>231</td>
</tr>
<tr>
<td>Appendix C.13</td>
<td>Subroutine TCOSTS</td>
<td>235</td>
</tr>
<tr>
<td>Appendix C.14</td>
<td>Subroutine QUIT</td>
<td>244</td>
</tr>
<tr>
<td>Appendix C.15</td>
<td>Subroutine RESCOST</td>
<td>248</td>
</tr>
<tr>
<td>Appendix C.16</td>
<td>Subroutine SRSWOP</td>
<td>250</td>
</tr>
<tr>
<td>Appendix C.17</td>
<td>Subroutine LSORT</td>
<td>251</td>
</tr>
<tr>
<td>Appendix C.18</td>
<td>Subroutine LSWOP</td>
<td>253</td>
</tr>
</tbody>
</table>
Appendix C.19  Subroutine CHRCNV  254
Appendix C.20  Subroutine CREAD  255

Appendix D  Program PERTRA2 and Subroutines  257

Appendix D.1  Program PERTRA2  257
Appendix D.2  Subroutine HEADER  260
Appendix D.3  Subroutine QDISPLAY  264
Appendix D.4  Subroutine FDISPLAY  271
Appendix D.5  Subroutine FILTER  282
Appendix D.6  Subroutine TDISPLAY  284
Appendix D.7  Subroutine CHRCNV  292
Appendix D.8  Function INTCHR  293
Appendix D.9  Subroutine CREAD  294

Appendix E  Sample of the Updated Task Data Written to
the File FLNAME.TSK  296

Appendix F  Program PERTRA3 and Subroutines  297

Appendix F.1  Program PERTRA3  297
Appendix F.2  Subroutine CALEND  304
Appendix F.3  Function IDOTW  306
Appendix F.4  Subroutine ANALYS  307
Appendix F.5  Function LNKDN  321
<table>
<thead>
<tr>
<th>Appendix</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>F.6</td>
<td>Function ISADED</td>
<td>322</td>
</tr>
<tr>
<td>F.7</td>
<td>Function ISDEDC</td>
<td>323</td>
</tr>
<tr>
<td>F.8</td>
<td>Subroutine TRIDIS</td>
<td>324</td>
</tr>
<tr>
<td>F.9</td>
<td>Subroutine DSELECT</td>
<td>325</td>
</tr>
<tr>
<td>F.10</td>
<td>Function RAND</td>
<td>326</td>
</tr>
<tr>
<td>F.11</td>
<td>Subroutine PCOSTS</td>
<td>327</td>
</tr>
<tr>
<td>F.12</td>
<td>Subroutine PSORT</td>
<td>334</td>
</tr>
<tr>
<td>F.13</td>
<td>Subroutine PRSTAT</td>
<td>335</td>
</tr>
<tr>
<td>F.14</td>
<td>Subroutine CASTAT</td>
<td>337</td>
</tr>
<tr>
<td>F.15</td>
<td>Subroutine RRSTAT</td>
<td>339</td>
</tr>
<tr>
<td>F.16</td>
<td>Subroutine RRSORT</td>
<td>341</td>
</tr>
<tr>
<td>F.17</td>
<td>Subroutine FREQUA</td>
<td>342</td>
</tr>
<tr>
<td>F.18</td>
<td>Subroutine FRQIRR</td>
<td>344</td>
</tr>
<tr>
<td>F.19</td>
<td>Subroutine FRQCAP</td>
<td>346</td>
</tr>
<tr>
<td>F.20</td>
<td>Function IDAY</td>
<td>348</td>
</tr>
</tbody>
</table>

**Appendix G**

One Run Data Output Written to the File

PERTRA3.RES

**Appendix H**

Multiple Runs Output Written to the File

PERTRA3.RES
LIST OF FIGURES

CHAPTER 1:

FIG. (1.1) Sensitivity Analysis Diagram (Spider Diagram) 8
FIG. (1.2) Probability Distribution 11
FIG. (1.3) Cumulative Probability Distribution 11

CHAPTER 2:

FIG. (2.1) Activity on Arrow Network 15
FIG. (2.2) Activity on Node Network 16
FIG. (2.3) Complex Links
   (a) The Fundamental Network 24
   (b) Precedence Network 24
FIG. (2.4) The Seasonal Variations in the Construction Productivity in the U.K. 29

CHAPTER 3:

FIG. (3.1) Project Pre—tender Program 34
FIG. (3.2) Project Cost—Revenue Pattern 37
FIG. (3.3) Project Cash Flow Diagram 40
FIG. (3.4) The Effect of Main Capital Lock—up 44
FIG. (3.5) The Effect of Front End Loading 46
FIG. (3.6) Effect of Client Payment Delay 48
FIG. (3.7) Project Cash Flow Chart with 8% Inflation 51
CHAPTER 4:

FIG. (4.1) The Beta Distribution
FIG. (4.2) A Unique Critical Path
FIG. (4.3) Interfering Critical Paths
FIG. (4.4) Zone of Equal Probability
FIG. (4.5) Triangular and Beta Distribution Compared
FIG. (4.6) Distribution Parameters
FIG. (4.7) Distribution Parameters Calculation Flow Chart
FIG. (4.8) Equi—Probability Bands
FIG. (4.9) Direct Calculation
FIG. (4.10) Selection of Task Duration Using Direct Calculation

CHAPTER 5:

FIG. (5.1) Program PERTRA1 Flow Chart
FIG. (5.2) Flow Chart for the Subroutine HEADER
FIG. (5.3) Subroutine READEXP Flow Chart
FIG. (5.4) Subroutine ABTASK Flow Chart
FIG. (5.5) Subroutine ABTRD Flow Chart
FIG. (5.6) Subroutine ABLINK Flow Chart
FIG. (5.7) Subroutine ABRES Flow Chart
FIG. (5.8) Subroutine ABSUBRES Flow Chart
FIG. (5.9) Subroutine CALWKS Flow Chart
FIG. (5.10) Subroutine TCOSTS Flow Chart 94
FIG. (5.11) Resource Hierarchy 95
FIG. (5.12) Program PERTRA2 Flow Chart 112
FIG. (5.13) Subroutine HEADER Flow Chart 113
FIG. (5.14) Subroutine FDISPLAY Flow Chart 117
FIG. (5.15) Subroutine TDISPLAY Flow Chart 118
FIG. (5.16) Subroutine QDISPLAY Flow Chart 120
FIG. (5.17) Program PERTRA3 Flow Chart 125
FIG. (5.18) Subroutine ANALYS Flow Chart 134
FIG. (5.19) Subroutine TRIDIS Flow Chart 136
FIG. (5.20) Subroutine PCOSTS Flow Chart 139
FIG. (5.21) Subroutine PRSTAT Flow Chart 141
FIG. (5.22) Subroutine FRQIRR Flow Chart 142

CHAPTER 6:

FIG. (6.1) Programs Screen Displays 144
FIG. (6.2) File Name Input Screen Display 147
FIG. (6.3) File Name Confirmation Screen Display 147
FIG. (6.4) Data Abstraction 1 Screen Display 148
FIG. (6.5) Data Abstraction 2 Screen Display 149
FIG. (6.6) Working Week Input Screen Display 150
FIG. (6.7) Working Week Confirmation Screen Display 151
FIG. (6.8) Cost Components Data Screen Display 153
FIG. (6.9) Cost Components Data Screen Display (2) 153
FIG. (6.10) Cost Components Data Screen Display (3) 154
FIG. (6.11) Cost Components Data Continuation
Screen Display (1) 154
FIG. (6.12) Cost Components Data Continuation
Screen Display (2) 155
FIG. (6.13) Termination Screen Display 156
FIG. (6.14) Program Introductory Screen 1 158
FIG. (6.15) Program Introductory Screen 2 158
FIG. (6.16) Program Introductory Screen 3 159
FIG. (6.17) Reading Data Screen 160
FIG. (6.18) Task Duration Data Screen 1 161
FIG. (6.19) Task Duration Data Screen 2 161
FIG. (6.20) Task Duration Data Screen 3 162
FIG. (6.21) First Filter Parameters Screen 1 163
FIG. (6.22) First Filter Parameters Screen 2 164
FIG. (6.23) First Filter Parameters Screen 3 165
FIG. (6.24) Second Filter Parameters Screen 1 166
FIG. (6.25) Second Filter Parameters Screen 5 167
FIG. (6.26) Filters Confirmation Screen 168
FIG. (6.27) Filters Logic Screen 169
FIG. (6.28) Filters Logic Confirmation Screen 170
FIG. (6.29) Sample of Tasks with VARIABLE Durations Display 171
FIG. (6.30) Optimistic Durations Factor Input Screen 172
FIG. (6.31) Pessimistic Durations Factor Input Screen 173
FIG. (6.32) Fctors Confirmation Screen 174
FIG. (6.33) Filtered Tasks Display Screen 175
FIG. (6.34) Coding Screen Display 176
FIG. (6.35) Program Termination Screen Display 177
FIG. (6.36) Project Durations Frequency Diagram 181
FIG. (6.37) Project Durations Cumulative Frequency Diagram 181
FIG. (6.38) Internal Rate of Return (IRR) Frequency Diagram 182
FIG. (6.39) IRR Cumulative Frequency Diagram 182
FIG. (6.40) CAPTIM Frequency Diagram 183
FIG. (6.41) CAPTIM Cumulative Frequency Diagram 183
<table>
<thead>
<tr>
<th>TABLE (3.1)</th>
<th>Project Direct Cost Breakdown</th>
<th>34</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE (3.2)</td>
<td>Project Summary</td>
<td>35</td>
</tr>
<tr>
<td>TABLE (3.3)</td>
<td>Cost/Revenue Summary</td>
<td>36</td>
</tr>
<tr>
<td>TABLE (3.4)</td>
<td>Project Cash Flow</td>
<td>38</td>
</tr>
<tr>
<td>TABLE (3.5)</td>
<td>Cash Flow for Frontend Loaded Project</td>
<td>45</td>
</tr>
<tr>
<td>TABLE (3.6)</td>
<td>Inflationary Project Cash Flow</td>
<td>51</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION TO RISK ANALYSIS

1.1 Introduction

Experience of many projects indicates poor performance in terms of achieving time and cost targets. Many cost and time overruns are attributable to either unforeseen events, which may or may not have been anticipated by an experienced project manager, or foreseen events for which uncertainty was not appropriately accommodated. It is suggested that a significant improvement to project management performance may result from greater attention to the whole process of risk management[9].

Risk and uncertainties do not occur only on large projects[9]. While size can be one of the major causes of risk, other factors include the complexity, speed of construction and location of the project.

Various authors have attempted to distinguish between risk and uncertainty and also between risk and speculative risk. However, in the practice of construction risk management, such distinctions are usually unnecessary and may even be unhelpful[9].
Risk and uncertainty are inherent features of all construction projects and it is suggested that the following are useful basic concepts:[9]

1. Risks and uncertainties are associated with specific events or activities which can be individually identified.

2. A risk event implies that there is a range of outcomes of the event, with each outcome having a probability of occurrence.

3. Some risks offer only the prospect of adverse consequences (loss), for example: structural collapse, bankruptcy, war, sea or flood damage; these may be of a low or high probability but are generally of high impact.

4. Many common construction risks offer the prospect of either loss or gain, for example: productivity of labour and plant, variation, inflation. These are typically of high probability but may be of low or high impact.

5. Because of the individual nature of construction projects, there is usually insufficient objective data to calculate the probability of occurrence of specific outcomes of risk events; some degree of subjective judgement is usually required.

Risks and their effects should be considered in all key decision points throughout the project and by all the parties involved in the decision making process.
1.2 Sources of Uncertainty

Construction is a process which takes place in conditions of extreme uncertainty. Since each construction project, to all intents and purposes, is unique, the planner of such a project is faced with problems which have never before occurred in exactly the same form in which they are presented to him, and he is thus obliged to make forecasts of actions and performance which rely on limited data. This data will become available to him as the work progresses, and the continuous flow of data is a prime reason for regarding the planning and control functions of management as two phases of the same operation. The continual flow of data brings the possibility of continual addition of detail and results in project plans[^6].

The planner can not, in the face of the uncertainty of the data available to him, simply abandon decision making. The detailed data, on which he is dependent, is generated only when the action is begun, and the action will not start if all decision are postponed until all is clear and certain. The planner must at least formulate a strategy for the project. In most cases he must do significantly more than this and provide, for example, cost estimates of an accuracy sufficient for the purpose of tendering. The planner is faced with the worrying task of basing important decisions on what is, at best, incomplete data[^6].

1.2.1 Accuracy of the Data

In most circumstances, the planner of a construction project can increase the confidence with which the data may be regarded by initiating further study
of the problem. He could, for example, commission extra site investigation studies or obtain more detailed information about the supplies of materials or labour in the area of the site. However, this gathering of extra data may be difficult to justify since it is expensive and time consuming. Investment in data gathering is subject to severely diminishing returns because some uncertainties are inherent in the construction process itself and, as accuracy in other data areas is increased, these inherent uncertainties begin to swamp the estimate. The cost of pre-project data gathering is an important constraint for the planner. Under the competitive tendering system, there is a one in six chance that a given tender will be successful. For five tenders out of six, therefore, the planning work is likely to be abortive and the extra costs incurred thereby can only be recovered by increasing the overheads included in the next tender. If pre-project planning costs are too high, overheads will increase with respect to those of competitors and the probability of success will fall correspondingly, thus increasing the problem. The planner must, therefore, aim to optimise the accuracy/cost equation and be prepared to work with data less than certain\(^8\). 

In addition to the cost constraints on pre-project planning, there is a severe constraint upon the time available for such since the competitive tendering system allows very little time for the gathering of additional data. For few projects would the planner be allowed more than twelve weeks to familiarise himself with the project and the area on which the site is located. In these circumstances the planner must work with little more information than that provided by the Client (or the Engineer on his behalf), and contained in the contract documents, together with specialised knowledge of his craft that he has gained through experience of similar projects\(^6\).
Obviously the situation concerning the gathering of data is less difficult if the project is being planned in an environment different from that of the construction industry competitive tendering regime. Time and cost constraints are less severe in these circumstances but they are still there, appearing as the problem, familiar in other design disciplines, of knowing when to 'stop thinking and start doing'. Even in this favourable environment, detailed deterministic data is not available concerning every parameter for, as has been remarked, some uncertainty is an inherent part of the construction process[6].

1.2.2 Inherent Uncertainties

Although the quality of most data can be increased by the gathering of more detailed information, the value of such an exercise is limited by the inherent and irreducible uncertainty of some of the project parameters.

The presence of one uncertain component in a sum prevents the elimination of uncertainty from the total and reduces the data enrichment value of other components. In these circumstances there is a level of accuracy which it is futile to exceed. Therefore, it is possible to increase the accuracy of some of the data available to the planner but not the accuracy of all the data and these inherent uncertainties affect the whole policy of pre-project planning and cost estimation[6].

The most obvious source of inherent uncertainty in Great Britain[6] is the weather. This is difficult to predict in the short term, impossible in the long term and, although the planner can reflect broad expectations in his plans, the unpredictability of detail must somehow be reflected in his plans. Similarly the
effects of erratic materials supply, labour unrest, and plant breakdown must be included in the plans, even though the existence or extent of these events cannot be known \textit{a priori}.

In view of the presence of uncertainty, and the difficulty of its removal, the planner must be aware of its effects. He may know the effects and chose to ignore them, but this should be a decision consciously made and not one made by default\cite{6}.

\textbf{1.3 Risk Analysis}

Many risks and uncertainties are quantifiable in terms of their effect on cost, time or revenue. Such risks can be analysed by measuring their effects on the parameters used to assess project or contract viability (profitability indicators)\cite{9}. 

The common principle behind any technique for the analysis of risk is to allow a range of values to the input data. For example, durations and costs for specific activities, or estimated revenues accruing from the project, are not given single values, but are considered over a range of values within which the decision maker believes they are likely to lie. An extension of this principle is to apply some statistical probability to any values used, so as to weight the possible outcomes to a more likely result\cite{9}.

However, for many construction risks there is insufficient objective data upon which calculations can be made of the probability of occurrence of outcomes. This leads some practitioners to be sceptical of the value of any
form of risk analysis since, by implication, the results must be dependent on some degree of subjectivity regarding the input data. 

1.3.1 Risk Analysis Techniques

Techniques for risk analysis have been theoretically established for a number of years, but their practical application to construction projects has been limited. Two of these are Sensitivity Analysis and Probability Analysis.

1.3.1.1 Sensitivity Analysis

Sensitivity analysis seeks to place a value on the effect of a variation of a single variable parameter within a project by analysing that effect on the project plan. It is the simplest form of risk analysis.

Uncertainty and risk are reflected by defining a likely range of variation for each component of the original base case estimate. In practice, such an analysis is only done for those variables which have a high impact on cost, time or economic return and to which the project will be most sensitive. The effect of change of each of these variable on the final cost or time criteria is then assessed in turn, across the assumed ranges.

If the effects of changes to several parameters are examined, a so called 'spider' diagram (see Figure 1.1) constitutes the most useful method of presenting the results such that the most 'critical' variables may be readily identified.
One weakness of sensitivity analysis is that the variables are treated individually. This leads to severe limits on the extent to which combinations of variables can be assessed directly from the data. A further weakness of such an analysis is that the sensitivity diagram gives no indication of the anticipated probability of occurrence of any event.

In spite of these weaknesses it has been found that considerable benefits...
may derive from the use of this simple tool. Such benefits includes:

(a) the powerful impact on management of the realization that there is a range of possible outcomes for a project;

(b) enhanced reality of the decision-making process, despite the increased complexity of the information on which such decisions are based;

(c) the ability to compare the robustness of projects to specific uncertainties;

(d) the relative importance of each variable becoming immediately apparent, thus highlighting those areas which would most benefit from any attempts to reduce or control uncertainty, or those areas which require further development work;

1.3.1.2 Probability Analysis

Probability analysis is a more sophisticated form of risk analysis: it overcomes the limitations of sensitivity analysis by specifying a probability distribution of each variable and then considering situations where any (or all) of these variables can change their initial values at any given time.

Defining the probability of occurrence of any specific value of a variable may be a difficult problem: not only does every project have many unique features, but political, commercial and fiscal environments change quickly. Nevertheless, it has proved possible to make tentative estimates of probability distributions and ranges.
Essentially, a distribution profile is allocated to the range which has been
defined for the variable. A number of profiles are possible, but simple ones
are advocated in the absence of statistical data.

The problem of assessing how risks can occur in combination is usually
overcome by using a sampling approach, running the analysis a number of
times taking random values of each variable (for example the Monte Carlo
technique)[9].

The outcome of this full analysis is a range over which the final solutions
could lie, and the probability of achieving such solutions is often shown
diagrammatically. Examples of such are shown in Figures 1.2[9] and 1.3[9].

Depending on the network model being used, and the results required, the
analysis can either be time—only, cost—only or, in view of the inherent
relationship between both parameters, integrated (time and cost).

Probability analysis has had some notable successes in terms of its
predictive ability and consequent assistance to managers in decision making[9].
Figure 1.2 Probability Distribution

Figure 1.3 Cumulative Probability Diagram
CHAPTER 2

NETWORK ANALYSIS

2.1 Historical Background to Critical Path Analysis (CPA)

Before the advent of CPA, probably the best-known way of trying to plan was by means of a bar or Gantt chart and, although this is extremely useful in many cases, it suffers from inability to show the inter-relationship between the various activities. Thus, it is not possible to deduce from a Gantt chart that, say activity X must be completed before activity I can be started, or that a delay between activity Y and activity Z is permissible but not essential. In small projects this is not serious, as the planner can remember the various links between activities, but in large projects such feats of memory are impossible and the Gantt charting technique is then of very limited value[10].

The middle of the 1950s saw an explosion in interest in this problem. In Great Britain[10], the Operational Research Section of the Central Electricity Generation Board investigated the problems concerned with the overhaul of generating plant a – task of considerable complexity which was increasing in importance as new high-performance plant was being brought into service. By 1957, the O.R. Section had devised a technique which consisted essentially of identifying the 'longest irreducible sequence of events' and, using this
technique, the time taken to carry out an experimental overhaul at a power station was reduced to 42\% of the previous average time for the same work. Continuing to work upon these lines, the overhaul time was subsequently further reduced by 1960 to 32\% of the previous average time. The rather clumsy name: 'longest irreducible sequence of events' was soon replaced by the name, 'major sequence' and it was pointed out, for example, that delays in the 'major sequence' would delay completion times, but that difficulties elsewhere need not necessarily involve extensions of total time. This work of the O.R. group was not made public, although comprehensive reports were circulated internally which foreshadowed much later work carried out elsewhere.

At much the same time, work was being undertaken in the U.S.A. and, in early 1958, the U.S. Navy Special Projects Office\cite{9} set up a team to devise a means of dealing with the planning and subsequent control of complex work. This investigation was known as Program Evaluation Research Task, which gave rise (or possibly derived from) the code name PERT. By February 1958, Dr. C.E. Clark, a mathematician in the PERT team, presented the early notions of arrow-diagramming, doubtless deriving from his study of graphics. This early work was rapidly polished and, by July 1958, the first report, *PERT: Summary Report Phase* \cite{10}, was published. By this time, the full title of the work had become *Program Evaluation and Review Technique* and the value of the technique seemed well established. By October 1958, it was decided to apply PERT to the Fleet Ballistic Missiles Programme\cite{10}, where it was credited with saving two years in the development of the Polaris missile.

Similar development work was being undertaken elsewhere, for example: in the U.S. Air Force\cite{10} under the code name PEP. Also in 1958, the E.I. du Pont de Nemours Company\cite{10} used a technique called the Critical Path
Method (CPM) to schedule and control a very large project and, during the first complete year's use of CPM, it was credited with saving the company $1.0 million. Subsequent use underlined the basic simplicity and extraordinary usefulness of this method, and by 1959, Dr. Mauchly, who had worked with Du Pont project, set up an organization to solve industrial problems using Critical Path Method.

Since 1958, considerable work has been carried out, mainly in the United States of America, in consolidating and improving these techniques.

2.2 The Network Model

Two systems of representation of networks are used in project management: the first is known as 'Activity on Arrow', examples being the well-known CPM and PERT programs; the second is known as 'Activity on Node', and PDM is an example of this.

2.2.1 Activity on Arrow Networks

Activity on Arrow networks, see Figure 2.1, were first used in the late 1950s when the two research teams coincidentally invented the very similar techniques CPM and PERT.

PERT and CPM are each based on the simplification of the fundamental network which occurs if the events linked by 'zero time' arrows are combined. If this is done, the arrows joining the nodes can be said to represent activities...
while the nodes continue to represent events, although now the events are, in fact, combinations of two or more simple events.

![Activity on Arrow Network](image)

**Figure 2.1 Activity on Arrow Network.**

The simplification has, however, been bought at the price of some ambiguity for the nodes representing more than one event. If, as is sometimes the case, it is necessary to limit the events at the node, then the 'zero time' arrows of the fundamental network must be re-introduced. Thus to represent the exact situation the 'zero time' arrows (called dummy arrows), must be shown in the network as illustrated by the 'broken' arrow in Figure 2.1. As the complexity of networks increases so does the proportion of dummy arrows in the network, robbing the diagram of some of the simplicity which was won by combining events.
2.2.2 Activity on Node Networks

A further simplification of the fundamental network entails combining the two events which mark the beginning and end of an activity, and the arrow which represent the activity itself. This combination leads to a technique known as the Precedence Diagram Method (PDM) which was introduced in the 1960s\[8\]. PDM is used in the construction industry to an increasing extent, although the ten—year lead by the Activity on Arrow diagrams has built a considerable 'brand loyalty' for the older techniques. A section from an Activity on Node Network is shown in Fig.2.2.

![Activity on Node Network](image)

**Figure 2.2 Activity on Node Network.**

The simplification of the network, brought by the change to precedence diagrams, is not bought at the cost of ambiguity in the diagram, for conventionally the nodes of a precedence network are drawn in the elongated form which makes it possible to visualise the left—hand end of the node as
the start of the activity and the right-hand end as the finish. Thus arrows will enter from the left and leave from the right. There is no need for dummy arrows or equivalent in precedence networks, for only one arrow, that presenting the carrying out of the task, can ever occur between the beginning and the end of the task.

2.2.3 A Comparison Between the Two Networking Systems

Although CPM and PERT have established a firm hold within the construction industry planning environment, there are strong arguments in favour of a change to the use of PDM. These arguments derive from the lack of ambiguity in PDM and from the relative ease of the use of PDM within computer programs.

It has been pointed out above that the CPM simplification of the original network introduces ambiguities. These ambiguities can be overcome by the use of dummy arrows but this destroys some of the simplifications and introduces a potentially dangerous dual status of arrows, some representing both logic and activities while others, the dummies, representing logic alone. Whilst the presence of dummy arrows may not be troublesome in simple networks, in complex networks they will dominate the diagram and will become troublesome when the analysis of the diagram is carried out. The precedence diagram, on the other hand, utilises only one type of arrow, that representing the logical link between activities, the activity arrow having been combined into the node.

The specification of the network for computer usage is much simpler if PDM, rather than CPM, is used. In CPM, the nodes present rather ephemeral
events and the activities are defined as linking these. Thus, to code the activities the operator of a computer program must first draw out the network, numbering the nodes, and then use this network during the input stage of the program. With PDM this is not so. In order completely to define the network the user must merely have a list of activities and be in a position to specify which activities are dependent on or precedent to other activities. No prior drawing of network is necessary.

Thus although there are two networking systems available for use in the construction industry, all the advantages, other than the familiarity of the industry with CPM, favour change to the widespread adoption of PDM. All the subsequent discussion and the subroutines developed for this research work will be concentrated, therefore, on the use of the precedence diagram as the most useful model of project.

2.3 The Analysis of Simple Networks

It has been argued that a project is a series of activities and the model which has been developed is such that the sequence of activities can be predicted mathematically by forming the activities into a network. The next concern is to develop a mathematical procedure whereby the network can be analysed.

All the activities in the network could take place at any time but, given a particular start date of the project, there is a limit to how early each activity commence. The first part of the analysis of networks is concerned with the identification of the earliest possible time for each activity in the network, this
operation being known as the forward pass.

If a forward pass is carried out for the whole network the, earliest possible time for the last activity will be obtained, and, if the aim is to complete the whole project in the shortest possible time, then this early time of the final activity will also be the latest permissible time. The calculation of these latest permissible times forms the second part of the analysis and is called the 'backward pass'.

2.3.1 The Forward Pass

In order to calculate the earliest possible time of an activity all the activities which must, in accordance with the logic of the project, precede it must be considered and allowance must be made for the durations which are assigned to any of the network arrows. In the case of the precedence network, where all the arrows have zero duration, the start time of an activity will be set to the latest of the finish times of the preceding activities.

As this process continually looks backward to the precedents, it is most efficient to carry out the calculations systematically, working from the start activity, which has no precedents, to the end activity which has every event in the network as its precedent. In this way the calculation is progressive and cumulative: an activity times being determined only when all its precedents have been themselves assigned an early time.

By convention, the upper part of the precedence network node is used to show the early dates: the upper left showing the earliest start of an activity
the upper right showing the earliest finish. Starting from the first activity in
the network, setting each start time to the latest precedent finish time and
each finish time to the start time plus the duration and the values are the
earliest possible times for the various activities i.e. for, the Activity on Arrow
networks, the earliest possible times (EET) by which each event can be
reached is as follows:

\[
\text{EET(succeeding event)} = \text{EET(preceding event)} + \text{Activity Duration}
\]

Where more than one activity leads into an event, the EET is taken to be
the largest value of event time computed from any of the routes leading into
that event.

For the Activity on Node model, the earliest start time (ES) of an activity
is the highest earliest finish times (EF) of preceding activities and

\[
\text{EF} = \text{ES} + \text{Activity Duration}
\]

2.3.2 The Backward Pass

If the end activity of the project is to take place at the earliest possible
time, then the times of all other activities in the network which affect it have
a time which they must not exceed. To calculate this latest permissible time a
process similar to that developed for the forward pass is required. In this case,
however, the analysis is concerned not with the activities between the start of
the project and the activity being considered, but with those between the
activity and the end of the project.
Once again the calculation proceeds in an orderly fashion, but in this case the progression is backwards from the finish activity which have no dependent activities to start, and hence the name 'backward pass'. For, Activity on Arrow networks, the calculation of latest event times (LET) is carried out as follows:

\[ \text{LET}(\text{preceding event}) = \text{LET}(\text{succeeding event}) - \text{Activity Duration} \]

where more than one activity leads from an event, the (LET) is taken to be the smallest value of event time computed using any of the routes leading back into that event.

For the Activity on Node model, the latest finish time of an activity (LF) is the lowest latest start time (LS) of their succeeding activities and

\[ \text{LS} = \text{LF} - \text{Activity Duration} \]

2.4 Criticality and Float

The forward and backward pass through the network produce for each activity a range of times at which the activity can take place without jeopardising the completion date of the project. The limits of this range are the earliest possible activity times, produced by the forward pass procedure, and the latest permissible time, the product of the backward pass.

In some cases these two times will coincide, and the earliest possible time will also be the latest permissible time, and then any slip will delay the completion of the project. These activities are known as the critical activities,
and the path through the network containing them is called the 'critical path'.

Activities away from the critical path have a range of possible times, and this range is known as float. Thus float, the difference between the early and the late times of an activity, is a measure of the tolerance within the network, and comparisons of the float of activities can help the manager of a project to assess priority. In complex projects where, for example, expensive resources are to be shared, the ability to delay the start of one activity, or through the sharing of a common resource to extend its duration, can be of major importance to the manager. Criticality and float make such decision possible.

2.5 Improvements to Realism

The validity of the analysis can be increased by making the model of the project on which the planner is working as realistic as possible. The aim of the developer of all interactive design programs should be to reduce to a minimum the mental gymnastics which the designer has to perform as he takes the presented results and assimilates them prior to deciding his next action[6].

Five techniques will be described, most of them being possible in the non-computer environment when used individually. Their use in combination would face the manual analyst with a complexity such that, at the least, the analysis would lose the elegance which makes hand calculation of network pleasant and that, in the worst case, would force the planner to make mistakes.
2.5.1 Complex Links

One of the advantages of the precedence diagram method is that the specification of links, which are more complex than the direct link, is possible. These complex links, which, together with their equivalents in the fundamental network model, are shown in Figure 2.3[^6], enable the planner to describe relationships which can only otherwise be described by the splitting of activities, an exercise which would result a significant increase in the complexity of the network. The complex links can be 'leads', which specify that the start of an activity is in some way dependent on the start of its preceding activity, or 'lags', which specify a relation between the finish of an activity and the finish of its preceding activity.

Complex links allow the planner to use larger activities than would be possible otherwise, and so reduce the complexity of networks both for the analyst and for the user of the results. Their use should be limited, however, for it is tempting for the planner to form 'omnibus' activities which contain a large number of sub—activities and which, because they are amorphous, make detail control impossible[^6].

Experience shows that, in a typical construction network, 10 to 15% of the links in a network may be complex. In some cases, however, particularly in the building industry, wide use is made of very large activities and up to 80% of the links may be complex (with corresponding control difficulties)[^6]. The incorporation of complex links into the analysis is not difficult. Referring to Figure 2.3 it can be seen that, in the forward pass, the start of activity 'BACKFILL' is tied not to the completion of the previous activity but to the partial completion thereof.
(a) The Fundamental Network

(b) The Precedence Network.

Figure 2.3 Complex Links
Thus 'BACKFILL' can start after one day's work has been done on 'DRAINAGE', and thus its start is tied to the start of 'DRAINAGE'. Similarly, the 'lag' link tying the start of the last day's work on 'BACKFILL' to the completion of drainage. In each case the usual forward and backward progression through the network is used as the basis of calculation.

Two points should be noted, the first of which is that, If the lag link in Figure 2.3 were not present there would be no tie between the finish dates of the activities and the rather strange situation would occur in which the earliest possible completion of the 'DRAINAGE' was day 8, but the latest permissible date was day 9, while the start of the activity remained critical. Consideration of the fundamental network explains this, for the float occurs within the second or third day's work on the 'DRAINAGE' activity leaving the first day's work as critical. Secondly, the delay between the start of the 'DRAINAGE' and the start of the 'BACKFILL' is not absolute. If the first day's drainage work is interrupted, then the delay will be extended.

Complex links are one of the strengths of the precedence network method in that they enable the planner to represent the network in the way that he pictures it (that, for example, the 'BACKFILL' starts one day after the start of the pipelaying) rather than forcing him to make rather artificial divisions in a continuous operation. Whilst they should be available as an option in all PDM programs, their use should be carefully monitored if difficulties in the control phase of the project are to be avoided.
2.5.2 Project Closedown

It is very unusual for projects to continue without a break other than the usual stoppages for weekends; even the busiest project has to accommodate national holidays. When these stoppages occur they obviously lengthen the duration of the current activities.

One way of dealing with this is to construct plans in terms of project days, the project day number being equal to the number of working days since the beginning of the project. This approach is widely used in the non-computer environment but has disadvantages. The major disadvantage is that the construction of a separate project calendar divorces the project activities from the outside world and makes interrelation between them more difficult for the planner and the manager[8]. A second problem caused by this approach is that such an analysis method can not handle those activities which are unaffected by close own; examples of these include the curing of concrete and the consolidation of an embankment under surcharge, i.e. those which continue whether there are men on site or not. An alternative to the project calendar approach which avoids these difficulties, but at a cost of extending the analysis time, is to consider the effect of closedown within the analysis itself. As the duration of each activity is being incorporated into the network (i.e. the calculation of finish times in the forward pass and start times in the backward pass) the location of the activity in time is checked to see if it coincides with a close—down period. If it does then the duration is extended by a time equal to the length of the close—down.
2.5.3 Window Times

A very frequent constraint on the planner of the projects is the requirement that an activity should fall within the specified period. Examples of this are the 'possession times' provided by the railways for the carrying out of work near their tracks and the 'weather windows' which have dictated the launch and positioning of off-shore structures for the oil industry. This type of constraint is easily built into the basic analysis method for the forward and backward pass merely by setting the event times prior to analysis for the forward and backward passes respectively to the start and finish of the time window for the activity concerned and then only altering the event times if they are to be increased (in the forward pass) or reduced (in the backward pass).

Experience shows that very rarely are project plans free from constraints of this type[6]. The absence of the facility automatically to include them in the program seriously inhibits the usefulness of planning programs, but their inclusion is both easy and cheap.

2.5.4 Calendar Dates

The tying of project events to calendar dates has already been shown to be important. The major advantage of the provision of calendar dates is that it removes the need for the planner to carry two calendars in his head, making the use of output much more easy. It is particularly valuable where the project is tied to external events as when, for example, window times are being used. The provision of calendar dates is one of the essential features of
a useful planning program.

Although the use of calendar dates is of great help to the planner, this help is necessary only at the output stage. There is, therefore, merit in delaying consideration of calendar dates until output data are being presented and in using a project calendar within the analysis.

2.5.5 Seasonal Variations

The construction industry in the U.K. is particularly sensitive to adverse weather conditions, and this sensitivity must be reflected by the planner as he positions activities in time and estimates activity durations using forecasts of production. Where this adjustment is made, it is made after the analysis of the network by extending those activities which have been shown to occur during the winter months and reducing the times of those which fall in the summer. Many Contractors have curves of productivity which act as guides for the planner and the site manager; one such is shown in Figure 2.4[e]. These curves correct not only for the effects of bad weather but also for the shorter working hours of the northern winter and the effect on the morale of the labour force due to working in unpleasant conditions.

A computer technique which allows for the seasonal variations to production is obviously of benefit, for all activities can then be estimated as if the work were to be carried out at those times of the year (April and September) when an average production can be assumed. Like the provision of window times, this facility is very easily built into the analysis method.
Figure 2.4 The Seasonal Variation in Construction Productivity in the UK.

The ability to adjust times automatically is advantageous not only to the planner of projects but also to those responsible for the control phase and, perhaps most lucratively, those concerned with the costing of disruption and variations. This technique cannot of course be used in the absence of a routine to calculate calendar dates. It consist simply of the identification of the month in which the activity is programmed to take place and the application of a suitable factor to the stored duration. It is a simple but remarkably effective addition to the traditional analysis, but one which cannot be made without the help of the computer.
CHAPTER 3

CASH FLOW FORECASTING

3.1 The Need for Cash Flow Forecasting

Each year the construction industry usually experiences a proportionally greater number of bankruptcies than do other industries[4]. One of the final causes of bankruptcy is a shortage of cash resources combined with a failure to convince financial institutions and other creditors that this shortage is only temporary. This sort of situation comes about for a variety of reasons, the most usual being a tendency to concentrate more on the relative magnitudes of cash inflows and outflows, i.e. profit, than on the relative timings thereof.

A cash flow is the transfer of money into or out of the company. The timing of cash flow is important. There is always, whatever the industry, a time lag between incurring the production or manufacturing costs of an article and receiving the cash payment for the finished product.

In the context of the construction industry, even though the contractor is paid at regular intervals throughout the project and not in a lump sum at the end, the contractor is required to carry the cost for somewhere between one and two months owing to the fact that work completed during any given month is not actually paid...
for until the end of the following month.

It is worth mentioning that, even if the contractor has credit facilities with suppliers of plant and materials, there are no credit facilities for paying labour, so that, the contractor will always have to carry a proportion of the cost until such time as he's paid.

It is this time lag between costs and receipts which is responsible for the most common economic feature of construction project. Namely that, particularly with a tight profit margin, most projects don't break even until they are virtually completed.

Together with the risks associated with construction, this is probably the reason why the construction industry experiences more bankruptcies in any given year than the majority of other industries.

It can be conclude that, irrespective of the size of the company, the need to forecast cash requirements is vital so that provisions can be made for these difficult times before they occur.

Contractors who undertake cash flow forecasting do so at two levels: Project level—which is carried out at the tendering stage, for single projects—and company level, carried out at regular intervals, which involves the aggregation of the cash flows for all active projects[4].

These two types of forecasts require different treatments. In the foremer case, the estimator has all the project details at the estimating stage and, because the forecast applies only to one project, the estimator can produce a carefully calculated
forecast based on these details by allocating bill items to 'activities' on the pre-tender 'bar chart' or 'network'. This creates a direct link between the estimators build-up for each item and the pre-tender construction program and allows the production of value versus time and cost versus time curves from which 'cash in' and 'cash out' can be calculated. This calculation is too detailed to be repeated for every project, every quarter or every month, so contractors often devise short cuts when undertaking the company or divisional cash flow forecasts, such shortcuts allowing cash flow forecasts to be made more regularly than just the one detailed forecast at the estimating stage.

3.2 The Requirement for a Forecasting System

Cash flow forecasting is strongly advisable and, for it to be meaningful, must be done regularly. It therefore follows that, for forecasting to be done regularly, the method must be simple and yet accurate enough for the purpose. It is essential, therefore, to reduce the data required for cash flow forecasting to the minimum possible compatible with reliable forecasts and to streamline the necessary calculations[4].

Cash flow forecasting is carried out for the following reasons:

(a) To ensure that sufficient cash is available to meet demands.

(b) To provide a reliable indicator to financing institutions, that, advances made can be repaid according to an agreed programme.

(c) To determine the cost of financing.
(d) To ensure that cash resources are fully utilised to the benefit of the company.

3.2.1 The Data Needed for Cash Flow Calculations

In construction companies, the most appropriate approach is to calculate cash flows on a project basis (which is the immediate concern of this research project) and to aggregate the cash flows from all projects and head office to form the overall company cash flow. This could be structured into divisions or areas for larger companies. The data required for a project are:

1. The magnitude and timing of incurred costs (cash outflow).

   i.e. the costs incurred by the contractor in carrying out the project, being based on the estimator's cost build up and the pre—tender programme.

2. The magnitude and timing of receipts (cash inflow).

   i.e. the payments by the client for the work completed, being based on incurred costs, plus the appropriate profit margin, and the pre—tender programme.

The compilation of a project cash flow is illustrated by considering a small construction project, the pre—tender programme and estimated financial breakdown of which are shown in Figure 3.1[17] and Tables 3.1[17] and 3.2[17].
### Table 3.1 Project Direct Cost Breakdown.

<table>
<thead>
<tr>
<th>ACTIVITY</th>
<th>LABOUR</th>
<th>MATERIALS</th>
<th>PLANT</th>
<th>SUB-CONT.</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>17,000</td>
<td>9,450</td>
<td>11,250</td>
<td>-</td>
<td>37,700</td>
</tr>
<tr>
<td>B</td>
<td>12,160</td>
<td>12,780</td>
<td>47,550</td>
<td>-</td>
<td>72,480</td>
</tr>
<tr>
<td>C</td>
<td>3,240</td>
<td>1,800</td>
<td>2,250</td>
<td>-</td>
<td>7,290</td>
</tr>
<tr>
<td>D</td>
<td>13,580</td>
<td>7,580</td>
<td>9,100</td>
<td>-</td>
<td>30,240</td>
</tr>
<tr>
<td>E</td>
<td>11,340</td>
<td>9,450</td>
<td>17,010</td>
<td>-</td>
<td>37,800</td>
</tr>
<tr>
<td>F</td>
<td>14,880</td>
<td>8,200</td>
<td>9,100</td>
<td>-</td>
<td>31,980</td>
</tr>
<tr>
<td>G</td>
<td>18,720</td>
<td>11,850</td>
<td>16,400</td>
<td>-</td>
<td>46,980</td>
</tr>
<tr>
<td>H</td>
<td>20,450</td>
<td>11,350</td>
<td>13,600</td>
<td>-</td>
<td>45,400</td>
</tr>
<tr>
<td>J</td>
<td>5,500</td>
<td>-</td>
<td>9,460</td>
<td>43,200</td>
<td>58,160</td>
</tr>
<tr>
<td>K</td>
<td>10,980</td>
<td>6,120</td>
<td>7,290</td>
<td>-</td>
<td>24,400</td>
</tr>
<tr>
<td>L</td>
<td>28,420</td>
<td>18,970</td>
<td>47,460</td>
<td>-</td>
<td>94,850</td>
</tr>
<tr>
<td>M</td>
<td>42,100</td>
<td>23,400</td>
<td>28,000</td>
<td>-</td>
<td>93,500</td>
</tr>
<tr>
<td>N</td>
<td>12,800</td>
<td>6,360</td>
<td>18,960</td>
<td>-</td>
<td>37,920</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
<td>10,500</td>
<td>-</td>
<td>48,100</td>
<td>58,600</td>
</tr>
<tr>
<td>Q</td>
<td>19,530</td>
<td>10,890</td>
<td>13,050</td>
<td>-</td>
<td>43,470</td>
</tr>
<tr>
<td>R</td>
<td>5,870</td>
<td>3,150</td>
<td>3,780</td>
<td>-</td>
<td>12,800</td>
</tr>
<tr>
<td>S</td>
<td>4,760</td>
<td>2,880</td>
<td>3,920</td>
<td>-</td>
<td>11,560</td>
</tr>
<tr>
<td>T</td>
<td>4,960</td>
<td>2,610</td>
<td>3,150</td>
<td>-</td>
<td>10,710</td>
</tr>
<tr>
<td>U</td>
<td>5,400</td>
<td>12,900</td>
<td>11,400</td>
<td>-</td>
<td>29,700</td>
</tr>
<tr>
<td>V</td>
<td>3,870</td>
<td>2,070</td>
<td>2,810</td>
<td>-</td>
<td>8,550</td>
</tr>
<tr>
<td>TOTALS</td>
<td>254,940</td>
<td>172,120</td>
<td>274,740</td>
<td>91,300</td>
<td>793,100</td>
</tr>
</tbody>
</table>

Figure 3.1 Project Pre-tender Programme.
Table 3.2 Project summary.

Based on the pre-tender programme and the financial breakdown, the cumulative cost and revenue pattern is shown in Table 3.3[17] and Figure 3.2[17].
<table>
<thead>
<tr>
<th>ACTIVITY</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>37.700</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>-</td>
<td>46.320</td>
<td>24.150</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>-</td>
<td>-</td>
<td>7.290</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>D</td>
<td>-</td>
<td>-</td>
<td>16.120</td>
<td>16.120</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>E</td>
<td>-</td>
<td>-</td>
<td>12.800</td>
<td>26.200</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>15.890</td>
<td>15.890</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>G</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>11.700</td>
<td>23.400</td>
<td>11.700</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>H</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>9.080</td>
<td>18.160</td>
<td>18.160</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>J</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>29.080</td>
<td>29.080</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>K</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>16.260</td>
<td>6.120</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>L</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>13.660</td>
<td>27.100</td>
<td>27.100</td>
<td>27.100</td>
<td>-</td>
</tr>
<tr>
<td>M</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>48.780</td>
<td>6.870</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>N</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>6.320</td>
<td>12.840</td>
<td>12.840</td>
<td>6.320</td>
<td>-</td>
</tr>
<tr>
<td>O</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>23.440</td>
<td>23.440</td>
<td>11.720</td>
<td>-</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>14.490</td>
<td>14.490</td>
<td>14.490</td>
</tr>
<tr>
<td>Q</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>12.600</td>
<td>-</td>
</tr>
<tr>
<td>R</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>5.480</td>
</tr>
<tr>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>U</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>9.800</td>
</tr>
<tr>
<td>V</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

| CUMULATIVE COST | 44.490 | 99.920 | 188.440 | 249.280 | 342.860 | 438.980 | 536.210 | 632.450 | 710.870 | 770.960 | 826.430 | 874.100 |

Table 3.3 Cost/Revenue Summary.

The cost/revenue data has been compiled subject to the following assumptions:

1. The costs associated with each activity are evenly spread over the duration of the activity.

2. Overheads are evenly spread over the project duration.

3. Each activity carries the same profit margin.
4. Revenue is earned immediately each item of work is completed

The cost and revenue curves in Figure 3.2 represent the commitment of the contractor, to pay the necessary costs, and the client, to pay for the completed items of work, they do not represent the actual monetary transactions required to produce the project cash flow.

The project cash flow as given in Table 3.4 is compiled in accordance with the following assumptions in regard of receipts and incurred costs:

Figure 3.2 Project Cost—Revenue Pattern.
<table>
<thead>
<tr>
<th>ACTIVITY</th>
<th>MONTH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>A</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>B</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>C</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>D</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>E</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>F</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>G</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>I</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>J</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>N</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>P</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>Q</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>R</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>S</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>T</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>U</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>V</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td>CUMULATIVE COST</td>
<td>23,750</td>
</tr>
<tr>
<td>GROSS RECEIPTS</td>
<td>-</td>
</tr>
<tr>
<td>RETENTION</td>
<td>-</td>
</tr>
<tr>
<td>CUMULATIVE RECEIPTS</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3.4 Project Cash Flow.

38
(a) Receipts from the Client

1. Work completed in any given month is certified for payment at the end of that month and paid for at the end of the following month.
2. 5% retention is deducted from each monthly valuation, up to a cumulative maximum of 3% of the tender total.
3. 50% of retention is to be refunded on completion.
4. 50% of retention is to be refunded on expiry of the maintenance period of 5 months.

(b) Incurred Costs

1. Labour paid as required.
2. Payment for plant and materials subject to one month's credit.
3. Subcontractors are paid in monthly 'single payment' increments immediately following receipts of the appropriate certified payment from the client.

3.3 Capital Lock-up

The negative cash flow in the early stages of projects represents 'locked-up' capital which must either be supplied from the contractor's cash reserves or borrowed from financial institutions. If the required capital is borrowed the interest
payable should be charged to the project; if the required capital is supplied from the contractors reserves the company is being deprived of its interest-earning capability and should therefore charge the project accordingly. A measure of the interest payable is represented by the area between the cash outflow and cash inflow curves, given the units CAPTIM (capital x time). Figure 3.3[17]. shows the cash flow diagram for the project together with the calculated negative captim.

This compound measure, say £688,803 £ x month, represents the volume of borrowing from the extremes of £688,803 for one month to £1 for 688,803 months.
A simple interest calculation is all that is required to convert captim into an interest charge, if it is assumed that the required finance is borrowed at an annual interest rate of 13¼%, then:

\[
\text{Chargeable interest} = 688,803 \times [(1.135)^{1/12} - 1] = £7,307
\]

where \([(1.135)^{1/12} - 1]\) is the monthly equivalent of 13¼% p.a.

If the required margin is to be maintained, the interest charges may be passed on to the client in one of the following forms:

(a) An additional lump in the BoQ Adjustment item.

(b) A general increase of item rates.

(c) A combination of both.

While if the tender sum is remain unchanged by absorbing the interest charges as additional overhead costs, the margin is reduced thus:

\[
\text{Total cost} = £874,100 + £7307 = £881,407
\]

\[
\frac{\text{Tender Sum} - \text{Total Cost}}{\text{Total Cost}} \times 100% = \frac{917,807 - 881,407}{881,407} \times 100% = 4.13%
\]
The maximum required cash resource can be determined directly from the project cash flow, which is equal to £136,265 at the end of month eight.

Interest calculation are usually based on the negative captim only, assuming that the cash released by the project does not earn interest, or the interest earned by the positive captim is subtracted from the interest paid on the negative captim. It is likely that the interest paying rate will be different and probably greater than the interest earning rate. The use of this captim measure enables the effect of interest charges to be evaluated.

### 3.3.1 Factors Affecting Capital Lock—up

The factors that affect the capital lock—up for an individual project or contract are as follows.

#### 3.3.1.1 Margin

The margin, whether profit margin or contribution (profit plus head office overheads) margin, is amongst the most important because it determines the excess over costs and is this excess that control the capital lock—up. Quite simply, the larger the margin, the less capital that is locked up; conversely, the smaller the margin, the more capital that is locked up in the contract.

The margin that should be used in calculating contract cash flows should be the 'effective' margin that is being achieved at the time of executing the contract. This effective margin is neither the margin included at the tender stage nor the margin
achieved at the end of the contract when all claims are settled. For example if the
tender panel included a margin of 9% and due to variations and other client
interference the cost rose by 4%, this would reduce the 'effective' margin to only
5%. Even if the contractor recovered another 6% in claims, making his overall
achieved margin at the end of the contract 10%, the 'effective' margin at the time
of executing the contract and thus determining the cash flow, would still be only
5%.

Most contractors use the tender margin when calculating contract cash flows and
this leads to an optimistic forecast. Although the margin is an important factor in
determining a contract's cash flow the tender margin is usually chosen for market
reasons rather than cash flow reasons. The effects on the project under consideration
are shown in Figure 3.4[17].

3.3.1.2 Retention

In the U.K. the system of retention simply reduces the effective margin during
the execution of the contract and the effect of retentions can be included in the
calculations as shown. In times of very low margins the retention can reduce the
effective margin to zero or less. As retentions are fairly standard in public sector
contracts there is little scope for negotiation to reduce retention and improve cash
flows.

3.3.1.3 Claims

As explained above under the section of margins, claims can return a contract to
its original intended level of profit. However, as the settlement of claims is normally subject to some delay the actual settlement does not improve a contract's cash flow and the circumstances given rise to a genuine claims are likely to worsen the contract's cash flow. The settlement of claims is, of course, important to the company's cash flow and therefore it is important that claims are settled as quickly as possible.

Figure 3.4 The Effect of Margin on Capital Lock-up.
3.3.1.4 Front-end Rate Loading

Front end rate loading is the device whereby the earlier items in the bill carry a higher margin than the later items. This has the effect of improving the effective margin in the early stages of the contract while keeping the overall margin at a competitive level. It is in these early stages that capital lock up is at its worst. The degree to which front end rate loading can be done depends on the client’s awareness.

The effects on project under consideration is shown in Table 3.5 and Figure 3.5 in which the overall tender margin of £43,705 (Table 3.2) is carried 'pro-rata' by activities A to J (Figure 3.1) which are completed in the first six months of the project.

<table>
<thead>
<tr>
<th>MONTH</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUMULATIVE COSTS</td>
<td>23,750</td>
<td>58,300</td>
<td>134,130</td>
<td>202,860</td>
<td>285,860</td>
<td>378,400</td>
<td>470,840</td>
<td>573,330</td>
<td>682,070</td>
<td>794,320</td>
<td>858,890</td>
<td>851,300</td>
<td>874,100</td>
<td>-</td>
</tr>
<tr>
<td>VALUE</td>
<td>-</td>
<td>48,828</td>
<td>60,611</td>
<td>72,950</td>
<td>93,000</td>
<td>103,573</td>
<td>103,503</td>
<td>96,080</td>
<td>93,240</td>
<td>84,190</td>
<td>88,080</td>
<td>85,480</td>
<td>38,870</td>
<td>-</td>
</tr>
<tr>
<td>RETENTION</td>
<td>-2,446</td>
<td>-3,241</td>
<td>-3,648</td>
<td>-4,860</td>
<td>-5,184</td>
<td>-5,129</td>
<td>-3,428</td>
<td>-</td>
<td>-</td>
<td>-13,767</td>
<td>-13,767</td>
<td>-13,767</td>
<td>-13,767</td>
<td>-13,767</td>
</tr>
</tbody>
</table>

Table 3.5 Cash Flow for Front end Loaded Project.
3.3.1.5 Overmeasurement

Overmeasurement is the device whereby the amount of work certified in the early months of a contract is greater than the amount of work done. This is compensated for in later measurements. Thus, overmeasurement has the same effect as front end rate loading; it improves the 'cash in' in the early stages and reduces the capital lock-up.
3.3.1.6 Back-end Rate Loading and Undermeasurement

Back end rate loading is the opposite to front end rate loading. Back end rate loading is the device whereby the later items in the bill carry a higher margin than the earlier items. Undermeasurement is the situation whereby the amount of work certified in the early months of a contract is less than the amount actually done. Both these devices have the effect of increasing the capital lock-up and, in most circumstances, these situations are not sought by contractors. However in index-linked price adjustment contracts these devices enhance the monies recovered from price fluctuations in times of inflation. If inflation is high in comparison to the cost of money the contractor may get a better return by funding a larger capital lock-up and gaining on price fluctuations rather than minimising the capital lock-up in the more traditional way. This circumstances existed in the United Kingdom in 1975 and 1976[4].

Thus, only when inflation is very high, which is generally difficult to predict, and borrowing limitations are not critical, can the approach of back end rate loading be countenanced at the tender stage. However, monthly over or undermeasurement gives a contractor a flexibility of policy over the duration of a contract such that changes can be made quickly in response to the predicted movements in the inflation indices and money markets. The effectiveness of this policy also depends on the client's awareness.

3.3.1.7 Delay in Receiving Payment from Client

The time between interim measurement, issuing the certificate and receiving payment is an important variable in the calculation of cash flows. Although monies
out goes to many destinations, e.g. labour, plant hirers, materials suppliers and subcontractors, the monies in comes only from one source (the client). Thus any increase in the delay in receiving this money delays all the income for the contract with a resulting increase in the capital lock—up. The time allowed for this payment is specified in the contract and normally interest is charged if payment is late. This may act as incentive to the client but if he is slow in paying it is the contractor who has to find the cash. The effect of delays on receiving payments on the project under consideration is shown in Figure 3.6.
3.3.1.8 Delay in Paying Labour, Plant Hirers, Materials Suppliers and Subcontractors

The time interval between receiving goods or services and paying for these is the credit the contractor receives from his suppliers. A one week delay is normal in paying labour and any thing between three to six weeks is normal in paying plant hirers and material suppliers. Any increase in these would reduce the capital required to fund a contract. However these items evolve as part of the normal commercial trading arrangements and any increase in these times may undermine commercial confidence in the company. Hence, these factors are not usually seen as suitable for controlling capital lock-up in a contract.

3.3.1.9 Inflation

The effects of inflation on the project cash flow under consideration is shown in Table 3.6[17] and Figure 3.7[17], which have been compiled using the following data:

1. Tender Margin: 5% at current prices.

2. Inflation Rate: 8% p.a.

3. Interest Rate: 13½% p.a.

4. Increased costs reimbursed using the CPF clause in the I.C.E. conditions of contract.
The parameters in Table 3.6 have been calculated as follows:

1. Average Cost Increase Factor for costs incurred the \( n^{th} \) month,

\[
(\text{ACIF})_n = \left\{ \frac{(1.08)^{n/12} - (1.08)^{(n-1)/12}}{2} \right\}
\]

2. Actual costs incurred during the \( n^{th} \) month

\[= (\text{Cost})_n \times (\text{ACIF})_n\]

3. Price Fluctuation Factor for work carried out during the \( n^{th} \) month,

\[
(\text{PFF})_n = 0.9 \left\{ (1.08)^{n/12} - 1 \right\}
\]

4. Payment of increased costs incurred for work completed during the \( n^{th} \) month,

\[= (\text{Value})_n \times (\text{PFF})_n\]

The calculated financial effects are as follows:

Tender Sum: Remains unchanged.

Interest on borrowed capital (@ 13\% p.a.): £7743

Total Cost: £912,047 + £7743 = £919,790

\[
\text{Effective Margin} = \frac{935,015 - 919,790}{919,790} = 3.61\%
\]

Maximum Cash Requirement: £143,607 at the end of month eight.
### Table 3.6 Inflationary Project Cash Flow.

<table>
<thead>
<tr>
<th>MONTH</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
</tr>
</thead>
<tbody>
<tr>
<td>COST INCREMENT</td>
<td>23.760</td>
<td>35.650</td>
<td>64.830</td>
<td>78.720</td>
<td>81.010</td>
<td>92.740</td>
<td>103.440</td>
<td>93.490</td>
<td>86.410</td>
<td>80.810</td>
<td>72.680</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>M. COST INCREASE FACTOR</td>
<td>1.00322</td>
<td>1.01067</td>
<td>1.01817</td>
<td>1.02571</td>
<td>1.03338</td>
<td>1.04117</td>
<td>1.04915</td>
<td>1.05734</td>
<td>1.06575</td>
<td>1.07438</td>
<td>1.08323</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ACTUAL CUMULATIVE COSTS</td>
<td>23.826</td>
<td>59.720</td>
<td>125.550</td>
<td>208.902</td>
<td>288.280</td>
<td>383.385</td>
<td>493.188</td>
<td>614.085</td>
<td>747.223</td>
<td>839.360</td>
<td>922.547</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>48.813</td>
<td>97.824</td>
<td>196.816</td>
<td>398.032</td>
<td>606.040</td>
<td>902.274</td>
<td>1200.803</td>
<td>1600.052</td>
<td>2000.302</td>
<td>2400.552</td>
<td>2800.802</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PPP</td>
<td>0.00576</td>
<td>0.01162</td>
<td>0.01748</td>
<td>0.02334</td>
<td>0.02933</td>
<td>0.03531</td>
<td>0.04132</td>
<td>0.04733</td>
<td>0.05334</td>
<td>0.05936</td>
<td>0.06540</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>INCURRED COSTS</td>
<td>27.5</td>
<td>67.2</td>
<td>131.0</td>
<td>254.6</td>
<td>362.7</td>
<td>540.4</td>
<td>714.7</td>
<td>933.0</td>
<td>1141.7</td>
<td>1341.7</td>
<td>1541.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 3.7 Project Cash Flow Chart with 8% Inflation.**
3.4 Internal Rate of Return (IRR)

The internal rate or return of a project is the discount rate that make a serious of cash flows both negative and positive, costs and receipts, discount net, Net Present Value, equal to zero, Mathematically defined:

\[
\sum_{t=0}^{n} \frac{A_t}{(1+r)^t} = 0
\]

where \( A_t \) = cash flows in and out
\( r \) = internal rate of return
\( t \) = time period
\( n \) = project duration

In order to find the IRR, it is necessary to establish the value of the interest that will just equate the present worth of all of the future cash flows both positive and negative considered over the full period of the project.

The IRR can only be determined by using trail and error. This achieved by estimating a value of \( r \), the required IRR, the present worth of the future net cash flow, is then calculated using this value, it is obviously saves time and calculation if the initial guess is reasonably close to the true rate of return.

If the resulting calculation gives a positive answer, then a second higher rate is chosen. This will smaller discount factor and hence a smaller discounted cash flow. If the net answer is now negative, the actual rate of return can be by interpolation between the two assumed rates. If the second answer is still positive, a further assumption of a higher discount rate needs to be made and the calculation repeated until two interest rates are reached that give one negative net summation and one
positive net summation.

Interpolation between interest rates is carried out as though the relationship between the interest factors and the present worth is linear in form. This is not in fact the case and therefore a small but not overly significant error occurs in the final interest rate calculated by linear interpolation. The closer are the two interest rates between which interpolation will take place, the smaller will tend to be the error. Using the computer in calculation the IRR under this research will overcome this problem and that's by using the method illustrated in the flow chart in Figure 3.8.

\[
\begin{align*}
\text{Calculate NPV} \\
\text{If NPV is positive then} \\
\text{r = r + \Delta r} \\
\text{If NPV is not positive then} \\
\text{r = r - \Delta r} \\
\text{\Delta r = r/10} \\
\text{If \Delta r < 0.00001 then} \\
\text{STOP}
\end{align*}
\]
Chapter 1 has shown that in some circumstances the analysis of uncertainty as part of the general analysis process is desirable and that, at the least, the confidence of the planner in his results is increased if he has available the means to apprise the risk. This chapter will review the two techniques which have been developed to handle the analysis of uncertainties, and will concentrate on finding solutions to the difficulties facing the chosen method for the development of the computer programs for this research project.

4.1 The Two Approaches

Statistics could be said to be the science of risk, and thus it is natural that a solution to the problem of handling risk should be sought there. The early network analysis packages did not, in general, provide facilities for the handling of uncertainty, but those which did adopted the PERT technique and used elementary statistics and a simple arithmetical process for the analysis.

As been mentioned previously, PERT was devised in the 1950s with one particular project in mind: the research and development required for the
production of the Polaris missile\[6\]. Those responsible for the Polaris project were realistic enough to realise that a deterministic approach to the planning of the project would not be satisfactory. Since the uncertainty involved in such a research-centered project is so great that it quickly swamps the analysis and makes deterministic forecasts literally incredible. To overcome this difficulty the developers of the PERT system built into their analysis not only a value for the expected duration of each activity but also a parameter representing the expected variation of the duration. These two variables were then carried through the analysis and all results were presented in terms of these parameters.

An alternative approach which, because it relies on the power of modern computer, was adopted rather later than PERT, is the so-called 'Monte-Carlo' simulation\[6\]. This technique, previously described, uses the power and patience of the computer to analyse the project model many times, each time allowing the project variables to vary randomly within their allotted distribution. The results are stored as a list of recorded values on which statistical operations can be carried out. This technique has the advantage over the PERT technique of flexibility, but it buys this advantage at the price of computer aid.

4.1.1 The Statistical Method

The techniques of statistics can be used to combine uncertain numbers, and these techniques were used by the developers of the PERT system to enable them to produce useful answers from data which, because of the nature of the research and development task, were very uncertain\[6\]. Their method relied on the central limit theorem, which shows that the mean and the variance of the
sum of a number of uncertain numbers can be easily predicted and that the
distribution of the sum tends to be Gaussian in form.

Using the central limit theorem it is possible, if the value of the mean
and the variance for each component is known, quickly to calculate the mean
and the variance of the sum. Knowing the values of the mean and the
variance and knowing the form of the resulting distribution, the assessment of
risk is trivial.

The central limit theorem holds for any shape of component distribution
and thus the approach used in PERT is applicable to the asymmetrical
duration distributions for activities which are typically found in project
planning.

PERT was built around the 'Beta' distribution as the standard for the
component distribution. There is, however, no theoretical justification for this
choice, merely the pragmatic reason that it provides a wide variety of
distribution forms and the calculation associated with it are very simple.

The Beta distribution is of the form:

\[ Y = x^a (1 - x)^b \]

which, depending on the values adopted by a and b, produces a range of
distributions both skew and symmetrical (Figure 4.1).[6]

The beta distribution has the advantages over other distributions of
simplicity and flexibility.
The distributions are defined in PERT not by the specification of statistical parameters, which would be foreign (and frightening) to most planners, but by the specification of three values of duration, the most optimistic estimate, the estimate of the most likely duration, and the most pessimistic estimate. The optimistic and pessimistic estimates have a 10% chance of being optimistic (in the case of the pessimistic values) or pessimistic (in the case of the optimistic values). Denoting these values as \( a \), \( b \) and \( c \), the parameters of the activity distribution can be calculated as

\[
\tilde{\chi} = \frac{a + 4b + c}{6}
\]

\[
\sigma = \frac{c - a}{4}
\]
Thus it is possible very quickly to move from estimates of duration, which are familiar ground to the planner, to the statistical parameters required for use in the statistical analysis. These parameters can then be used within the analysis of the network which, using the central limit theorem, can merely add the values of the activities on the critical path in order to produce the parameters for the distribution of completion date.

This approach is powerful and allows the planner even without the aid of computer very quickly to produce distributions of event times and thus estimates of risk overrun. The calculations required of the planner are simple and the planner is, for most of the calculation, working with numbers which are significant to him (an important advantage if errors are to be avoided). The approach does, however, have an important difficulty, the handling of parallel critical paths.

4.1.1.1 Interfering Critical Paths

For a simple network with a unique critical path and considerable float elsewhere in the network the simple addition of the values of mean duration and of variance for each activity on the critical path is accurate. Such a situation is shown in Figure 4.2. In this simple network there are no circumstances in which path A can become critical; the length of the project is always governed by path B.

Difficulties arise if there are parallel paths which may each be critical in different circumstances. This situation is shown in Figure 4.3. In this case it is possible for the network path which is normally critical (path B) to become
subcritical because another path (path A) exceeds it. Either of the paths can now be critical and the distribution of completion date must include both possibilities, because duration of less than 36 days can be produced with either A or B critical and each possibility must contribute to the probability of completion within a given time.

It is, of course, possible to handle this problem of interfering critical paths using the methods of classical statistics. For this case the distribution can be shown to be given by\(^6\)

\[
f(X) = f_A(X) \int (f_B((0 \rightarrow X))dX) \quad A \text{ is critical}
\]

\[
+ f_B(X) \int (f_A((0 \rightarrow X))dX) \quad B \text{ is critical}
\]

but this calculation becomes onerous for large and complex networks where the number of possible interferences is much larger than the two described here.

The difficulty associated with the use of PERT for complex networks dictates the need for an alternative approach in such situations. This does not imply that PERT is not a useful tool, merely that in complex networks it loses the simplicity which is its major strength\(^6\). One alternative to PERT, the 'Monte Carlo simulation', has been available for many years but is becoming increasingly attractive as the cost of computer power, on which it is dependent, is reduced.
Figure 4.2 A Unique Critical Path.

Figure 4.3 Interfering Critical Paths.
4.1.2 Simulation Techniques

The analysis of uncertainty described before relies on carrying the parameters of uncertainty through the analysis and, using the method of statistics, combining them to provide parameters of uncertainty of the result. PERT carries out this calculation using the same method as would have been used in a purely deterministic analysis, but the analysis is done once only and the range of possible answers is represented in the result.

An alternative approach is to create the uncertainty of the result not by carrying uncertainty through a single analysis but by carrying a large number of analyses using data values chosen at random from the original ranges. Thus the distribution comes not from the carrying of uncertainty through a single analysis but from the combination of the results of a large number of deterministic analyses. This alternative technique is known as Monte Carlo simulation.

It is obvious that the distribution produced by a simulation analysis is dependent upon the data values of the component parts. The analysis seeks to model many repetitions of the project and thus must take, as input, values of the variable (in this case duration) which are chosen from a menu of equally probable values going into analysis must consist not only of the choice at random of a value from within the specified range, but must also weight the choice towards the most probable values. This weighted choice is made by dividing the distribution of the variable into equal areas (that is area of equal probability) and then by choosing at random between these areas. Figure 4.4[6] shows how, assuming a menu of ten values and a beta distribution, this would be done manually.
When the distribution of each component data value has been specified and divided in this way, a random choice is made between the equiprobable zones for each component and one cycle of the simulation analysis is carried out by allotting a value for the duration of each activity to that which represents the chosen zone. Many cycles are carried out and the results of the analyses are combined to provide distributions for specified events.

4.2 Monte Carlo Simulation

This technique is based on experimentation and simulation and is used in situations where a solution in the form of an equation would be difficult or impossible. The basic steps are:

(a) The range of values of the risk being considered are assessed, together with the probability distribution most suited to each risk.
(b) A value for each risk within its specified range is selected; this value should be randomly chosen and must take account of probability distribution.

(c) The outcome of the project is calculated using the combination of values selected for each one of the risk.

(d) The calculation is repeated a number of times to obtain the probability distribution of the project outcome; the iterations required depends on the number of variables and the degree of confidence required, but typically lies between 100 and $1000^\circ$.

4.3 The Difficulties of Simulation

A simulation analysis is a useful part of an interactive design system only if it is fast. This need for speed must be recognised in the writing of the computer programs and, at times, it may be necessary to sacrifice accuracy on this particular altar. The saving in calculation time can be achieved, without serious loss of accuracy, by reducing the number of simulation cycles and by storing cumulative rather than individual values of data. The third area of saving is the generation of the individual distribution prior to the selection of deterministic duration values. The implication of these recommendations to the development of a computer program must now be considered.
4.3.1 The Necessary Number of Simulations

If the analyst wishes accurately to predict the distribution of any event in the network then he will require a very large number of simulation cycles to produce it. Reducing the number will produce progressively less satisfactory result. If, however, the analyst requires an estimate of mean event time and is satisfied with an assumption of a normal distribution and an estimated value of standard deviation, then the number of simulation cycles can be enormously reduced.\(^6\)

4.3.2 Data Storage

Even in large computers the problem of data storage can be troublesome if both the number of events in a network and the number of simulation cycles is high. This problem can be overcome by combining the data as they are produced and carrying forward a cumulative total. This technique, bringing the advantage of reducing the amount of memory required or of increasing the speed of operation of the program, has the disadvantage that it destroys the shape of the distribution and imposes a normal distribution on the result. Previous studies\(^6\) have shown that this is not a serious error at least for events away from the early events near the start of the network.
4.3.3 Distribution Calculation

It has become usual to present the distribution of activity duration by a Beta distribution given by the formula

\[ Y = X^a (1 - X)^b \]

There is no theoretical basis for this choice; it is chosen merely because the curve can be moulded, by the selection of the parameters \( a \) and \( b \), to fit the curve to a shape that experienced planners would draw to describe the distribution of durations. The curve can be symmetrical or asymmetrical, flat or sharp. Where manual calculations are being carried out, particularly within the PERT program, the distribution provides easy formula for both mean and standard deviation and is thus convenient.

When the method of analysis is changed to a Monte Carlo simulation, as is being described here, and when the means of doing the work is moved from the manual environment to the computer, then the Beta distribution becomes rather less convenient. No direct use is now made of the mean and standard deviation of the component distributions; instead the centroids of the zones of equal probability must be identified.

Clearly this calculation does not present any difficulty: given the three duration values as is usual in PERT (and assuming a value \( b \)) the value of \( a \) can be calculated. Progressive numerical integration of the curve can then be used to produce the positions of the various centroids and these can be used as the menu items from which the random choice is made prior to each simulation cycle. While this calculation is simple, it is also cumbersome and
time consuming, especially when it must be repeated for each activity for each simulation cycle. There is merit, therefore, in identifying a distribution which allows the direct calculation of the centroids of equiprobable zones.

It has been proved from previous studies\[^6\] that the results of simulation analyses are insensitive to the shape of the individual activity duration distribution if the number to be combined is greater than two or three, and that this condition applies to all the events in the network apart from those very near the beginning of the forward pass. In view of this insensitivity it seems to be unnecessarily pedantic to insist on the use of a traditional but not theoretically derived beta distribution, for any distribution which approximates to the general shape of the expected duration could be used. When this is considered along with the need for a geometrically simple shape, it seems that a triangular distribution could form a compromise between simplicity, giving fast calculation times, and shape, giving a realistic representation. Figure 4.5\[^6\] contrasts the triangular distribution recommended with the beta distribution it replaces.

![Figure 4.5 Triangular and Beta Distribution Compared.](image)
4.4 Triangular Probability Distribution

4.4.1 Calculation of the Parameters

From Figure 4.6, \( f_0, f_m \) and \( f_p \) are the probability values at the durations \( d_0, d_m \) and \( d_p \) respectively.

Known parameters:

Optimistic duration \( d_0 \)
Most likely duration \( d_m \)
Pessimistic duration \( d_p \)

\( d_0 \) and \( d_p \) are chosen such that there is respectively a 99% and 1% probability of exceedence.

Overall area = 1.0

Figure 4.6 Distribution Parameters.
\[ f_m(\delta_1 + \delta_2 + \delta_3 + \delta_4) = 1.0 \quad (1) \]

Shaded area = 0.01

\[ \frac{1}{2} \delta_1 f_o = 0.01 \quad (2) \]

\[ \frac{1}{2} \delta_4 f_p = 0.01 \quad (3) \]

**Linearity**

\[ \frac{f_o}{\delta_1} = \frac{f_m}{\delta_1 + \delta_2} \quad f_o = \frac{f_m \delta_1}{\delta_1 + \delta_2} \quad (4) \]

\[ \frac{f_p}{\delta_4} = \frac{f_m}{\delta_3 + \delta_4} \quad f_p = \frac{f_m \delta_4}{\delta_3 + \delta_4} \quad (5) \]

from equation (2)

\[ f_o = \frac{0.02}{\delta_1} \]
Substitute in equation (4)

\[
\frac{0.02}{\delta_1} = \frac{f_m \delta_1}{\delta_1 + \delta_2}
\]

\[
f_m \delta_1^2 - 0.02 \delta_1 - 0.02 \delta_2 = 0 \tag{6}
\]

Equation (6) is a quadratic which may be solved for \(\delta_1\) from:

\[
\delta_1 = \frac{0.02 \pm \sqrt{(0.02)^2 + 0.08 f_m \delta_2}}{2f_m} \tag{7}
\]

But

\[
\sqrt{0.02^2 + 0.08 f_m \delta_2} > 0.02 \text{ for all } f_m, \delta_2^* \]

Since \(\delta_1\) must be > 0 for all cases:
\[ \delta_1 = \frac{0.02 + \sqrt{0.04 + 0.08f_m \delta_2}}{2f_m} \]  

(8)

From equation (3) \[ f_p = \frac{0.02}{\delta_4} \]

Substitute in equation (5)

\[ f_m \delta_4 - 0.02^2 \delta_4 - 0.02 \delta_3 = 0 \]  

(9)

Using similar reasoning as above, this yields

\[ \delta_4 = \frac{0.02 + \sqrt{0.04 + 0.08f_m \delta_2}}{2f_m} \]  

(10)

Calculation of the distribution parameters is carried out in accordance with the flow chart in Figure 4.7
4.5 Random Selection of Task Duration

The essence of so-called 'Monte Carlo' simulation technique is that, for each task, a duration is randomly selected from the range of durations defined by the appropriate probability distribution, with the selection being weighted towards the most probable value.

There are two approaches by which this can be achieved, both of which
make use of a pseudo random number generator which outputs a series of uniformly distributed random numbers within a specified range.

The two approaches are detailed hereunder for the particular case of a triangular probability distribution. It may be assumed that the principles remain the same for any distribution provided that the parameters of that distribution are known.

4.5.1 Equi—probability Bands

The overall zone of probability (i.e. the distribution area) is subdivided into N zones of equal probability as shown in Figure 4.8

For convenience, each equi—probability band is associated with a single duration (d_i, i = 1, N) which is taken to be the duration value corresponding to the band centroid.
Owing to the non-linear native of the equations used to determine the band limits and centroidal values, the band parameters are evaluated progressively, using the \( N^{th} \) band parameters as a basis for evaluating the \((N+1)^{th}\) band parameters.

In any given instance, the task duration \( d_i \) is selected on the basis of a uniformly distributed random in the range \( 1 \leq i \leq N \).

### 4.5.2 Direct Calculation Method

Consider the triangular probability distribution shown in Figure 4.9

\[ P( d < d_j ) = A_j \]

where

\[ A_j = 0 \quad \text{for } d_j = D_0 \]
\[ A_j = 1 \quad \text{for } d_j = D_1 \]

Figure 4.9 Direct Calculation
Thus a given value of $A_j$ in the range $0 < A_j < 1$ may be associated with a value of $d_j$ in the range $D_0 < d_j < D_1$.

A few calculations show that if values of $A_j$ are randomly selected from a uniformly distributed series of values $0 < A_j < 1$, the resultant distribution of $d_j$ is of the form shown.

In any instance, therefore, the task duration may be calculated directly from a uniformly distributed random number in the range $0.0 < A < 1.0$

4.5.3 Comparison

By comparing the two approaches, it has been found that:

In terms of computational requirement the 'Equiprobability' band approach has two major disadvantages

(a) A significant amount of calculation is required to define the bands for each task.

(b) Once defined, the centroidal duration for each task require storage for subsequent use. This require significant memory space.

In terms of flexibility the Equiprobability band approach precludes the use of integer tasks duration. Thus given rise to a number of disadvantages:

(1) Operations with floating point numbers are substantially slower than operations with integers.
(2) Fractional tasks durations introduces unnecessary complications into the 'Cash Flow' computations which would not arise with integer task durations.

(3) In reality, although a task may very well be completed in, say 10.30 days it is unlikely that the succeeding task will commence until the start of day 12. Thus, the duration of the task is effectively expanded to 11 days i.e an integer duration.

The above leads to select the 'Direct calculation' method for the development of the computer subroutine required.

4.5.4 Selection of Task Duration Using the Direct Calculation Method

From Figure 4.10

Overall area = 1

\[
S_0 = \frac{f_m}{D_m - D_0}; \quad S_1 = \frac{f_m}{D_m - D_1}
\]
Case 1 \((0 \leq A \leq A_0)\)

\[ d_j = D_0 + \frac{2A}{\sqrt{S_0}} \]

Case 2 \((A_0 < A \leq 1)\)

\[ d_j = D_1 - \frac{2(1-A)}{\sqrt{S_1}} \]
CHAPTER 5

THE COMPUTER PROGRAMS

5.1 Introduction

As suggested in chapter 4, the analysis of uncertainty in a large and complex project is best carried out by performing a large number of project simulations, based on data values (such as activity durations) chosen at random from within specified probability distribution, resulting in a range of possible solutions (such as project durations) defined by a single probability distribution.

In order to apply this technique, known as 'Monte Carlo' simulation, the suite of PERTRA programs was developed to operate on the basis of project data (task durations, logical dependencies, resource usage etc.) output, in the form of a text file, by PERTRMASTER ADVANCE – one of the most widely used commercial software packages for construction planning.

The PERTRA suite comprises three separate programs as follows:

1. PERTRA1: Data Reading and Initialisation;
2. PERTRA2: Interactive Input of Task Duration Data;
3. PERTRA3: Risk Analysis and Cash Flow Calculations;
All three programs are written in FORTRAN 77 and, in order to utilise the graphic support provided thereby, PERTRA1 and PERTRA2 compiled using some of the 'C' language graphics subroutines.

The programs will now be discussed in turn.

5.2 PERTRA1— Data Reading and Initialisation

The essence of the program is as follows:

1. Reading of selected data from a user-specified file containing project data output by PERTRMMASTER ADVANCE.

2. Initialisation of this data to convert it to a form suitable for use by PERTRA2 and PERTRA3.

3. Creation of intermediate data files for use by PERTRA2 and PERTRA3.

The program flow chart is shown in Figure 5.1 and the program listing is given in Appendix C.1.
SUBROUTINE DIMCHK
Check adequacy of array dimensions; terminate program if dimensions inadequate

SUBROUTINE READEXP
Read the following data from user-specified file:
General data, Task data, Link data, Resource data
Subresource data and Calendar data

SUBROUTINE CALWKS
Keyboard entry of working week appropriate to each calendar used by the project

Convert resource names into indices

SUBROUTINE TCOSTS
Calculate user-specified cost components for each task

Define duration of non-zero S-S links as a percentage of source task duration

SUBROUTINE LSORT
Sort logical links into a suitable order to facilitate analysis of the project network

SUBROUTINE QUIT
Write data to specified data files; Display concluding message.

Figure 5.1 PERTRA1 Flow Chart
The main program routines, subroutines and functions are now discussed:

5.2.1 Subroutine HEADER

The purpose of this subroutine is to generate an introductory screen display which provides general information, pertaining to the use of PERTRA1, and to enable keyboard input of the PERTMASTER export data file name.

The above is carried out as shown in the flow chart in Figure 5.2. The program listing is given in Appendix C.2.

Figure 5.2 Flow Chart for the Subroutine HEADER.
5.2.2 Subroutine READEXP

This subroutine reads data from the user-specified export file (FLNAME.EXP) generated by PERTMASTER.

The contents of the data export file are formatted as shown in Appendix... The abstraction of data from the file is carried out in accordance with the flow chart shown in Figure 5.3. The subroutine listing is given in Appendix C.4.

As can be seen in Appendix A, the contents of each record in the export file contain a number of parameters which may or may not be required by the PERTRA programs. Each of these parameters is 'comma delimited'. Selective data abstraction, therefore, is carried out on the basis of the 'comma-related' position of the relevant parameters in each record. This is achieved by calling subroutine RECCOM, immediately after reading any given record, which determines the number of commas (NCOMMA) in the record and the positions thereof (ICOMMA(I), I=1,NCOMMA).

Selective data abstraction is carried out by the following subroutines called by subroutine READEXP, with the choice of subroutine being determined by type of record (denoted by the initial character of the record) required to be read.
Figure 5.3 Subroutine READEXP Flow Chart.

SUBROUTINE READEXP
5.2.2.1 Subroutine ABTASK

This subroutine abstracts the required Task data, from a given Task record in the export file and codes tasks according to their position in the network as follows:

- **Start task**: SFCODE = 0
- **Normal task**: SFCODE = 1
- **Finish task**: SFCODE = 2

The steps of the above are shown in the flow chart in Figure 5.4 and the listing of the subroutine is given in Appendix C.6.

Task data abstracted by subroutine ABTASK is as follows:

1. Task type (TTYPE).
2. Task name (TNAME).
3. Task work breakdown structure (TWBS).
4. Task calendar (TCAL).
5. Task duration (TDURN).
6. Task code (SFCODE).
Figure 5.4 Subroutine ABTASK Flow Chart.
5.2.2.2 Subroutine ABTRD

This subroutine abstracts the following Task Resource Demand data from the Task Resource demand records:

1. Task resource name (CTRES).
2. Task resource quantity (CTRESQ).

The subroutine flow chart is shown in Figure 5.5 and the listing is given in Appendix C.7.

![Figure 5.5 Subroutine ABTRD Flow Chart.](image-url)
5.2.2.3 Subroutine ABLINK

This subroutine abstracts the following link data from a link record:

1. Source Task Name (CLFROM);

2. Target Task Name (CLTO);

3. Link Type (CLTYPE);

4. Link Duration (LDURN).

As can be seen in Appendix A, the first three of the above parameters are alphanumeric variables and require to be abstracted as such. In FORTRAN programs, however, operations involving alphanumeric variables are considerably slower and less efficient than those involving numeric data. It is, therefore, preferably to use the latter wherever possible. This being the case, subroutine ABLINK converts source/target task names to indices and link types to numbers. As an example the Export File link data is shown below:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEAR1</td>
<td>SETUP</td>
<td>F→ S</td>
<td>0</td>
</tr>
<tr>
<td>CLEAR2</td>
<td>FINISH</td>
<td>F→ S</td>
<td>0</td>
</tr>
<tr>
<td>CMPRESS</td>
<td>SSEPCO</td>
<td>F→ F</td>
<td>0</td>
</tr>
<tr>
<td>CMPRESS</td>
<td>SSWPCO</td>
<td>F→ F</td>
<td>0</td>
</tr>
<tr>
<td>CPESC1</td>
<td>CPC1FW</td>
<td>F→ S</td>
<td>0</td>
</tr>
<tr>
<td>CPESC1</td>
<td>CPC1RF</td>
<td>S→ S</td>
<td>2</td>
</tr>
</tbody>
</table>
Converted link data will be as follows:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>209</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>248</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>258</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

The subroutine flow chart is shown in Figure 5.6, with the listing being given in Appendix C.8.
SUBROUTINE ABLINK
Abstracts link data from export file record (REC)
Changes source task LFROM and target task LTO
from names to indices
Outputs link type LTYPE in integer form

Determine commas positions in record
SUBROUTINE RECCOM

Abstract data between second and third commas as source task name CLFROM

Abstract data between third and fourth commas as target task name CLTO

Abstract data between fourth and fifth commas as link type CLTYPE

Abstract data after fifth comma as link duration LDURN

Change Source and Target task names to indices:
Source Task Index • LFROM
Target Task Index • LTO

Change link type from alphanumeric variable to integer variable:
Link Type Index + LTYPE
F-S = 1 ; S-S = 2 ; F-F = 3 ; S-F = 4

RETURN

Figure 5.6 Subroutine ABLINK Flow Chart.
5.2.2.4 Subroutine ABRES

This subroutine abstracts the following resource data from a resource record.

1. Resource Name (RNAME).

2. Resource Type (RTYPE).

The abstraction of the resource data is carried out as illustrated in the subroutine flow chart in Figure 5.7. The subroutine listing is given in Appendix C.9.

![Figure 5.7 Subroutine ABRES Flow Chart.](89)
5.2.2.5 Subroutine ABSUBRES

Abstracts the following SUB—RESOURCE data from a sub—resource record:

1. Resource Name (CRTOP).
2. Sub—resource Name (CRSUB).
3. Sub—resource Quantity (RSUBQ).

The flow chart of the subroutine ABSUBRES is shown in Figure 5.8. Listing of the subroutine is given in Appendix C.10.
5.2.2.6 Subroutine ABCAL

This subroutine abstracts the Calendar Name (CAL) from a calendar record. The listing of the subroutine is given in Appendix C.10.

5.2.3 Subroutine DIMCHK

One of the major disadvantages associated with the use of FORTRAN, in the context of reading data files of varying lengths, is the requirement that array dimensions should be specified at the time of compiling the program. In the event that the amount of data, being read by a FORTRAN program from such a file, exceeds the specified capacity of the arrays, however, the program continues to read the 'excess' data and overwrites (to the appropriate extent) the previously—filled arrays. Since no error message is displayed, the user is blissfully unaware of such happening.

To prevent such a possibility occurring in PERTRA1, the subroutine DIMCHK has been included in the program immediately before subroutine READEXP.

Subroutine DIMCHK operates in a similar manner to subroutine READEXP, but reads only the first character of each record, determines its type and accumulates the numbers of each type of record. Should these numbers exceed the specified array dimensions, an error message is displayed and the program is terminated. The subroutine listing is given in Appendix C.3.
5.2.4 Subroutine CALWKS

This subroutine determines working week length (ICWK) for each task by facilitating keyboard input of the working week length for each calendar. This is carried out in accordance with the subroutine flow chart in Figure 5.9. Listing of the subroutine CALWKS is given in Appendix C.12.

Figure 5.9 Subroutine CALWKS Flow Chart.
5.2.5 Subroutine TCOSTS

This subroutine enables keyboard input of cost-component data in order to determine the following cost components for each task in the network:

- Labour cost
- Material cost
- Plant cost daily
- Plant cost front end
- Plant cost back end
- Subcontract cost
- Overheads daily
- Overheads front end
- Task price

For each component, the following data is input from the keyboard:

1. Component Code (e.g. XLC for labour cost).
2. Paymeny Accumulation (Monthly/Weekly)
3. Payment Delay (Days/Weeks/Months).

The subroutine flow chart is shown in Figure 5.10 and the subroutine listing is given in Appendix C.13.
Figure 5.10 Subroutine TCOSTS Flow Chart.
5.2.6 Subroutine RESCOST

One of the noteworthy features of PERTMASTER is the use of a resource 'hierarchy' whereby any given resource may call upon a number of sub-resources which may, themselves, call upon sub-sub-resources and so on. The system is illustrated in Figure 5.11 for the resource BFG (Backfill Gang) which consists of:

1. Bomag 35 (B35) @ £75 per day;
2. JCB's (JCB) @ £150 per day;
3. Labourers (LAB) @ £46.5 per day;
4. 7 Ton Tipper Trucks (TIP) @ £80 per day;

and each tipper truck make use of a driver (DRV) @ £50 per day.

![Figure 5.11 Resources Hierarchy.](image)

The hierarchy of the resource BFG will appear in the export file in alphabetic order as follows:
On the basis of the resource hierarchy, subroutine RESCOST determines the total quantities of a specified cost component used by each 'top—level' resource (where a top—level resource is one which is directly utilised by a task). It is then a straightforward matter, by considering each top—level resource used by a given task, to calculate the total quantity of the specified cost component associated with that task.

The operation of subroutine RESCOST is best explained by considering the operations involved in determining, say, the total labour cost component (XLC) associated with Backfill Gang (BFG).
The subresources array generated by subroutine ABSUBRES, corresponding to the export file data previously shown, will be as shown below.

<table>
<thead>
<tr>
<th>I</th>
<th>RTOP(I)</th>
<th>RSUB(I)</th>
<th>RSUBQ(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>7</td>
<td>B35</td>
<td>XPC</td>
<td>75</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>10</td>
<td>BFG</td>
<td>B35</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>BFG</td>
<td>JCB</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>BFG</td>
<td>LAB</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>BFG</td>
<td>TIP</td>
<td>2</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>17</td>
<td>DRV</td>
<td>XLC</td>
<td>50</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>25</td>
<td>JCB</td>
<td>XPC</td>
<td>150</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>28</td>
<td>LAB</td>
<td>XLC</td>
<td>46.5</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>33</td>
<td>TIP</td>
<td>DRV</td>
<td>1</td>
</tr>
<tr>
<td>34</td>
<td>TIP</td>
<td>XPC</td>
<td>80</td>
</tr>
</tbody>
</table>

For ease of understanding, the variables RTOP(I) and RSUB(I) have been shown as a containing resource codes as opposed to resource indices, although the latter will be the case in reality.
1. RESCOST identifies each of the above records where RSUB(I) = XLC. The arrays are then re-ordered such that these records are brought to the 'top' of the array, thus:

<table>
<thead>
<tr>
<th>I</th>
<th>RTOP(I)</th>
<th>RSUB(I)</th>
<th>RSUBQ(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DRV</td>
<td>XLC</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>LAB</td>
<td>XLC</td>
<td>46.5</td>
</tr>
<tr>
<td>7</td>
<td>B35</td>
<td>XPC</td>
<td>75</td>
</tr>
<tr>
<td>10</td>
<td>BFG</td>
<td>B35</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>BFG</td>
<td>JCB</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>BFG</td>
<td>LAB</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>BFG</td>
<td>TIP</td>
<td>2</td>
</tr>
<tr>
<td>25</td>
<td>JCB</td>
<td>XPC</td>
<td>150</td>
</tr>
<tr>
<td>33</td>
<td>TIP</td>
<td>DRV</td>
<td>1</td>
</tr>
<tr>
<td>34</td>
<td>TIP</td>
<td>XPC</td>
<td>80</td>
</tr>
</tbody>
</table>
RESCOST searches the re-ordered array RSUB (starting from I = 3) for those resources (in this case LAB) in RTOP which have been identified as making direct use of XLC. The arrays are then re-ordered as shown.

<table>
<thead>
<tr>
<th>I</th>
<th>RTOP(I)</th>
<th>RSUB(I)</th>
<th>RSUBQ(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DRV</td>
<td>XLC</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>LAB</td>
<td>XLC</td>
<td>46.5</td>
</tr>
<tr>
<td>3</td>
<td>TIP</td>
<td>DRV</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>BFG</td>
<td>LAB</td>
<td>4</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>7</td>
<td>B35</td>
<td>XPC</td>
<td>75</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>10</td>
<td>BFG</td>
<td>B35</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>BFG</td>
<td>JCB</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>BFG</td>
<td>TIP</td>
<td>2</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>25</td>
<td>JCB</td>
<td>XPC</td>
<td>150</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>34</td>
<td>TIP</td>
<td>XPC</td>
<td>80</td>
</tr>
</tbody>
</table>
3. Array RSUB is now searched for subresource which make use of DRV and LAB, namely TIP and BFG respectively. The arrays are sorted accordingly.

<table>
<thead>
<tr>
<th>I</th>
<th>RTOP(I)</th>
<th>RSUB(I)</th>
<th>RSUBQ(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DRV</td>
<td>XLC</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>LAB</td>
<td>XLC</td>
<td>46.5</td>
</tr>
<tr>
<td>3</td>
<td>TIP</td>
<td>DRV</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>BFG</td>
<td>LAB</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>BFG</td>
<td>TIP</td>
<td>2</td>
</tr>
</tbody>
</table>

As can be seen, the process terminates when the searching of RSUB reveals no more relevant sub-resources.

4. The array variable RCOST(J) is initialised thus:

\[
\begin{align*}
\text{RCOST(RTOP)} &= 1 \quad \text{for } \text{RTOP} = \text{XLC} \\
\text{RCOST(RTOP)} &= 0 \quad \text{for } \text{RTOP} \neq \text{XLC}
\end{align*}
\]
5. For I = 1,5, where I is the number of the re-ordered arrays, the following algorithm is used.

\[ \text{RCOST(RTOP(I))} = \text{RCOST(RTOP(I))} + \text{RCOST(RSUB(I))} \times \text{RSUBQ(I)} \]

Thus, the sequence of calculations in this case is as follows:

\[
\begin{align*}
I=1 & \quad \text{RCOST(DRV)} = \text{RCOST(DRV)} + \text{RCOST(XLC)} \times \text{RSUBQ(1)} \\
& = 0 + 1 \times 50 = \£50 \\
I=2 & \quad \text{RCOST(LAB)} = \text{RCOST(LAB)} + \text{RCOST(XLC)} \times \text{RSUBQ(2)} \\
& = 0 + 1 \times 46.5 = \£46.5 \\
I=3 & \quad \text{RCOST(TIP)} = \text{RCOST(TIP)} + \text{RCOST(DRV)} \times \text{RSUBQ(3)} \\
& = 0 + 50 \times 1 = \£50 \\
I=4 & \quad \text{RCOST(BFG)} = \text{RCOST(BFG)} + \text{RCOST(LAB)} \times \text{RSUBQ(4)} \\
& = 0 + 46.5 \times 4 = \£186 \\
I=5 & \quad \text{RCOST(BFG)} = \text{RCOST(BFG)} + \text{RCOST(TIP)} \times \text{RSUBQ(5)} \\
& = 186 + 50 \times 2 = \£286
\end{align*}
\]

Thus, the total labour cost component associated with the Backfill Gang = \£286 per day.

Such calculation will be performed for the other cost components associated with BFG.
Continuing the example, the calculations of the total Daily Plant cost (XPC) associated with the same resource BFG will be performed as follows:

1. Rescost identifies each of the records where RSUB(I) = XPC and brought it to the top, thus the arrays will be as follows

<table>
<thead>
<tr>
<th>I</th>
<th>RTOP(I)</th>
<th>RSUB(I)</th>
<th>RSUBQ(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B35</td>
<td>XPC</td>
<td>75</td>
</tr>
<tr>
<td>2</td>
<td>JCB</td>
<td>XPC</td>
<td>150</td>
</tr>
<tr>
<td>3</td>
<td>TIP</td>
<td>XPC</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>BFG</td>
<td>B35</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>BFG</td>
<td>JCB</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>BFG</td>
<td>LAB</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>BFG</td>
<td>TIP</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>DRV</td>
<td>XLC</td>
<td>50</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>LAB</td>
<td>XLC</td>
<td>46.5</td>
</tr>
<tr>
<td>29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>TIP</td>
<td>DRV</td>
<td>1</td>
</tr>
</tbody>
</table>

102
2. RESCOST searches the re-ordered array RSUB (starting from I= 4) for the resources in RTOP which has been identified as making direct use of XPC (in this case B35, JCB and TIP). The arrays are re-ordered as shown:

<table>
<thead>
<tr>
<th>I</th>
<th>RTOP(I)</th>
<th>RSUB(I)</th>
<th>RSUBQ(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B35</td>
<td>XPC</td>
<td>75</td>
</tr>
<tr>
<td>2</td>
<td>JCB</td>
<td>XPC</td>
<td>150</td>
</tr>
<tr>
<td>3</td>
<td>TIP</td>
<td>XPC</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>BFG</td>
<td>B35</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>BFG</td>
<td>JCB</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>BFG</td>
<td>TIP</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>BFG</td>
<td>LAB</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>DRV</td>
<td>XLC</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>LAB</td>
<td>XLC</td>
<td>46.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>TIP</td>
<td>DRV</td>
<td>1</td>
</tr>
</tbody>
</table>

3. Array RSUB is searched (starting from I= 7) for subresources which make direct use of B35, JCB and TIP namely BFG, which in this example will not be found and the process terminates when the search of RSUB reveals no more relevant sub-resources.
4. The array variable RESCOST(J) is initialised thus:

\[
\begin{align*}
\text{RCOST(RTOP)} &= 1 \quad \text{for} \quad \text{RTOP} = \text{XPC} \\
\text{RCOST(RTOP)} &= 0 \quad \text{for} \quad \text{RTOP} \neq \text{XPC}
\end{align*}
\]

5. For I = 1, 6 where I is the number of Swopped arrays, the same algorithm in the previous example is used. Thus the calculations in this case as follows:

\[
\begin{align*}
\text{I=1} & \quad \text{RCOST(B35)} = \text{RCOST(B35)} + \text{RCOST(XPC)} \times \text{RSUBQ(1)} \\
& \quad = 0 + 1 \times 75 = £75 \\
\text{I=2} & \quad \text{RCOST(JCB)} = \text{RCOST(JCB)} + \text{RCOST(XPC)} \times \text{RSUBQ(2)} \\
& \quad = 0 + 1 \times 150 = £150 \\
\text{I=3} & \quad \text{RCOST(TIP)} = \text{RCOST(TIP)} + \text{RCOST(XPC)} \times \text{RSUBQ(3)} \\
& \quad = 0 + 1 \times 80 = £50 \\
\text{I=4} & \quad \text{RCOST(BFG)} = \text{RCOST(BFG)} + \text{RCOST(B35)} \times \text{RSUBQ(4)} \\
& \quad = 0 + 75 \times 1 = £75 \\
\text{I=5} & \quad \text{RCOST(BFG)} = \text{RCOST(BFG)} + \text{RCOST(JCB)} \times \text{RSUBQ(5)} \\
& \quad = 75 + 150 \times 2 = £375 \\
\text{I=6} & \quad \text{RCOST(BFG)} = \text{RCOST(BFG)} + \text{RCOST(TIP)} \times \text{RSUBQ(6)} \\
& \quad = 375 + 80 \times 2 = £535
\end{align*}
\]

Thus the total Daily Plant cost component associated with the Backfill Gang is £535 per day.

Listing of the subroutine RESCOST is given in Appendix C.15.
5.2.7 Subroutine LSORT

Link data abstracted by the subroutine ABLINK are listed in the designated array in alphabetic order and, in order to perform the analysis calculations, this data requires to be arranged into an order suitable for carrying out the forward and backward passes through the network. Arranging the data into the above orders is carried out by the subroutine LSORT and the swopping subroutine LSWOP.

As with the subroutine RESCOST, the working of subroutine LSORT are best explained by the following example. The following network will be used for the explanation:
1. The variables of the link data abstracted by subroutine ABLINK from the export file will be stored in an alphabetic order as shown below:

<table>
<thead>
<tr>
<th>LFROM(I,1)</th>
<th>LTO(I,1)</th>
<th>LTYPE(I,1)</th>
<th>LDURN(I,1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>D</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>E</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>FINISH</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>A</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>D</td>
<td>F–S</td>
<td>0</td>
</tr>
<tr>
<td>START</td>
<td>C</td>
<td>F–S</td>
<td>0</td>
</tr>
</tbody>
</table>

For ease of understanding, the variables LFROM, LTO and LTYPE have been shown as containing alphanumeric names rather than indices, although the latter will be the case in reality (see section 5.2.2.3).

2. To re-order the link data for the Forward Pass, subroutine LSORT identifies the records where the variable LFROM(I,1) is the first task in the network (having SFCODE=0), using the subroutine LSWOP the arrays are re-ordered such that these records are brought to the 'top', thus:
3. Lsort identifies the relevant target task (in this case: C) and searches the array LFROM(I,1) for all instances where this task appears. As each instance is identified, the relevant link record is brought to the 'top' of the array thus:

<table>
<thead>
<tr>
<th>LFROM(I,1)</th>
<th>LTO(I,1)</th>
<th>LTYPE(I,1)</th>
<th>LDURN(I,1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>C</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>D</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>E</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>FINISH</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>A</td>
<td>F→S</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>D</td>
<td>F→S</td>
<td>0</td>
</tr>
</tbody>
</table>
4. The process is repeated for each of the target tasks appropriate to $c$ — and so on. The final result is as shown below:

<table>
<thead>
<tr>
<th>LFROM(I,1)</th>
<th>LTO(I,1)</th>
<th>LTYPE(I,1)</th>
<th>LDURN(I,1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>C</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>E</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>A</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>D</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>D</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>FINISH</td>
<td>F−S</td>
<td>0</td>
</tr>
</tbody>
</table>

5. The procedure of sorting the $(1,2)$ arrays into 'Backward Pass' order is the exact reverse of that for the 'Forward Pass' order but starting with the last task in the network (SFCODE=2). The final result is as shown below:

<table>
<thead>
<tr>
<th>LFROM(I,2)</th>
<th>LTO(I,2)</th>
<th>LTYPE(I,2)</th>
<th>LDURN(I,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>FINISH</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>D</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>D</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>F</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>A</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>E</td>
<td>F−S</td>
<td>0</td>
</tr>
<tr>
<td>START</td>
<td>C</td>
<td>F−S</td>
<td>0</td>
</tr>
</tbody>
</table>
The subroutine LSORT listing is given in Appendix C.17. and listing of the subroutine LSWOP is given in Appendix C.18.

5.2.8 Subroutine QUIT

This subroutine writes data to the following files for later subsequent use by PERTRA2 and PERTRA3:

1. FLNAME.GEN, general data file. See Appendix B.1.
2. FLNAME.TSK, tasks data file. See Appendix B.2.
3. FLNAME.LNK, link data file. See Appendix B.3.
4. FLNAME.CST, cost data file. See Appendix B.4.

Following a screen display to that effect the subroutine terminate the operation of PERTRA1. Listing of the subroutine is given in Appendix C.14.
5.3 PERTRA2: Interactive Input of Task Duration Data

For the purpose of carrying out the Risk Analysis, the various tasks involved in the project may be categorised as follows:

(a) Tasks with VARIABLE durations:

These are those tasks whose durations are variable in accordance with a probability distribution defined by:

1. 'Optimistic duration': 99% exceedance probability.
2. 'Most likely duration': used in PMA calculations.
3. 'Pessimistic duration': 1% exceedance probability.

(b) Tasks with INVARIABLE durations:

These are those tasks whose durations are invariable and not subject to the probability distribution mentioned above. Milestone and Hammock tasks, by their nature, are both included automatically in this category.

The purpose of PERTRA2, therefore, is to update the information contained in FLNAME.TSK; to include for:

(a) a code which identifies each task as having either a 'VARIABLE' or 'INVARIABLE' duration;

(b) the 'optimistic' and 'pessimistic' durations of each 'VARIABLE' duration task.
Since the purpose of PERTRA2 is essentially the same as that of
PERTRA1, namely: the initialisation of data for use by PERTRA3, the
operations performed thereby should logically have been included as part and
parcel of PERTRA1.

It was felt, however, that such an arrangement would introduce an
unnecessary element of rigidity in that it would require the relevant task
duration data to be entered, in its entirety, in a single session. Additionally, it
would preclude the updating of the task duration data, following an analysis
using one particular set of values for 'optimistic' and 'pessimistic' durations, to
facilitate a subsequent analysis on the basis of a different set of values thereof.

Consequently, PERTRA2 was designed as a separate module which could be
used at any stage in the Risk Analysis process.

A flow chart of the PERTRA2 is shown in Figure 5.12 and a listing of
the program is given in Appendix D.1.
PROGRAM PERTRA2  Interactive input of TASK DURATION for use by Risk Analysis program PERTRA3

Read FLNAME from the file PERTRA.FLS generated by the program PERTRA1

Display program introduction
SUBROUTINE HEADER

Read task data from FLNAME.TSK

Display screen inquiry:
(a) Input data?
(b) Quit?

SUBROUTINE QDISPLAY

QUIT  INPUT  

Input or Quit?

Input filtration parameters
SUBROUTINE FDISPLAY

Filter tasks in accordance with filtration parameters
SUBROUTINE FILTER

Display filtered tasks; input optimistic and pessimistic duration factors if required.
SUBROUTINE TDISPLAY

YES  NO

Filtration O.K.?

Variable

Update variability code for each displayed task
PCODE(i) = 1

Update OPTIMISTIC and PESSIMISTIC durations

Invariable

Update variability code for each displayed task
PCODE(i) = 0

Update FLNAME.TSK

RETURN

Figure 5.12 Program PERTRA2 Flow Chart
The main program routines, subroutines and functions are now discussed:

5.3.1 Subroutine HEADER

The purpose of this subroutine is to generate an introductory screen display which provides general information pertaining to the use of PERTRA2.

A flow chart of subroutine HEADER is shown in Figure 5.13 and a listing of the subroutine is given in Appendix D.2.

![Figure 5.13 Subroutine HEADER Flow Chart.](image_url)
5.3.2 Subroutines FDISPLAY, FILTER, TDISPLAY

The input of task duration date (i.e. 'optimistic' and 'pessimistic' durations) can be a somewhat tedious and time consuming exercise if carried out on a 'task by task' basis.

The time spent on the exercise, however, can be much reduced by taking advantages of a particular feature of many construction projects, namely that such projects contain groups of tasks which are identical in both nature and duration. Thus being the case, it may reasonably be inferred that the variability of the duration of the tasks within each such group is identical. If a given group may be identified, therefore, only one set of task duration parameters need to be input from the keyboard and will apply to each task within that group.

The above subroutines provide a facility whereby the overall task list may be 'filtered' to display selected tasks, with the subsequent input of task duration data applying to each of the displayed tasks and only to those tasks.

5.3.2.1 Subroutine FDISPLAY

This subroutine enables the keyboard input of up to six levels of the following four parameters:

CODE (Name, Type or Work Breakdown Structure of the tasks).

POSITION (The position in the CODE which defines the 'start' of the
SELECTION parameter

OPERAND ('Equal to' or 'Not Equal' to)

SELECTION (A maximum of 9 characters on which selection will be based).

For example, the filtration parameters:

<table>
<thead>
<tr>
<th>CODE</th>
<th>POSITION</th>
<th>OPERAND</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>3</td>
<td>EQ</td>
<td>CONC</td>
</tr>
</tbody>
</table>

will convey an instruction to PERTRA2 to display only those tasks having names which contain the alphanumeric string 'CONC' starting at position 3 therein.

Where more than one level of parameters is specified, FDISPLAY allows the input of a further parameter (AND or OR) which will determine the filtration logic.

If OR is selected, only one of the filtration criteria need be satisfied for a task to be displayed. If AND is selected, all of the filtration criteria require to be satisfied for a task to be displayed.
Amplifying the previous example, the filtration parameters:

<table>
<thead>
<tr>
<th>CODE</th>
<th>POSITION</th>
<th>OPERAND</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>3</td>
<td>EQ</td>
<td>CONC</td>
</tr>
<tr>
<td>TYPE</td>
<td>1</td>
<td>NE</td>
<td>H</td>
</tr>
</tbody>
</table>

LOGIC: AND

will result in the display of only the 'non-hammock' tasks whose names contain the alphanumerical string 'CONC' at position 3.

The subroutine flow chart is shown in Figure 5.14. Listing of the subroutine is given in Appendix D.4.

5.3.2.2 Subroutine FILTER

This subroutine carries out the actual filtration of tasks according to the parameters input via subroutine FDISPLAY. A listing of the subroutine is given in Appendix D.5.
Figure 5.14 Subroutine FDISPLAY Flow Chart.

117
5.3.2.3 Subroutine TDISPLAY

This subroutine scrolls the selected tasks to enable the user to check whether the required filtration has been successfully carried out. A flow chart of the subroutine is shown in Figure 5.15 and a listing of the subroutine is given in Appendix D.6.

![Figure 5.15 Subroutine TDISPLAY Flow Chart](image-url)
5.3.3 Subroutine QDISPLAY

This subroutine performs a double functions, namely:

(a) To generate the 'input of task duration data' screen display appropriate to the 'variability' (i.e. 'Variable' or 'Invariable') of the tasks displayed by TDISPLAY.

(b) To generate a 'quitting' screen display if the user desires to quit PERTRA2 following the input of task duration data.

The subroutine flow chart is shown in Figure 5.14 and a listing of the subroutine is given in Appendix D.3.
SUBROUTINE QDISPLAY
Interactive inquiry display returns the following
JCODE=0 Input data for INVARIABLE duration tasks
JCODE=1 Input data for VARIABLE duration tasks
JCODE=2 No more data to input QUIT program

Clear screen and display table

Display data input line
Read option from keyboard

NO

Option is QUIT?

YES

Update code JCODE=2

NO

Option is INPUT?

YES

Update code JCODE=1

Wrong option
Sound warning

Display TASK DURATIONS options
'Task Duration VARIABLE/INVARIABLE'

NO

Option is VARIABLE?

YES

Update code JCODE=0

Wrong option
Sound warning

Display continuing message
Read option from keyboard

NO

Option is CONTINUE?

YES

RETURN

Figure 5.14 Subroutine QDISPLAY Flow Chart.
5.3.4 Input of Task Duration Data.

Task duration data, for VARIABLE duration tasks, is input in the form of factors by which the 'Most Likely Duration' of task or group of tasks, may be multiplied to obtain the 'Optimistic' and 'Pessimistic' durations thereof.

(a) 'Optimistic duration' factor: ranges from 0.00 to 1.00;

(b) 'pessimistic duration' factor: ranges from 1.00 to 1000.00.

The input of task durations using factors have been chosen because of the following reasons:

(a) In reality 'Optimistic' and 'Pessimistic' durations are usually estimated and expressed as a percentage of the 'Most likely' duration (i.e. half, twice etc.).

(b) Allows the input of task durations in groups which is more efficient and less time consuming.
5.3.5 PERTRA2 Output.

By running the program PERTRA2 the file FLNAME.TSK will be updated to include the following data for each task:

VARIABILITY CODE ( = 0 for INVARIABLE; = 1 for VARIABLE).
OPTIMISTIC DURATION (In Days).
PESSIMISTIC DURATION (In Days)

An Example of the Updated file is given in Appendix E.
5.4 PERTRA3: Risk Analysis and Cash Flow Calculations.

Risk analysis (Time and Cost) using the Monte Carlo simulation technique is the reason behind the development of PERTRA suite of programs of which PERTRA3 is the program which carries out the analysis. The essence of PERTRA3 is as follows:

1. Reading of the data from the files generated by PERTRA1 and PERTRA2.

2. Performance of the following:

(a) Time analysis of the project network using the Most Likely Durations to calculate the project duration.

(b) Cost analysis to determine the project IRR and CAPTIM.

3. Determination of the Duration distribution for each task with a VARIABLE duration.

4. Performance of the following for each simulation:

(a) Random selection of the duration for each task based on the Triangular Probability distribution (see Section 4.5.2).

(b) Performance of the Time Analysis of the project network (based on the selected durations) to determine the project duration.
(c) Performance of the Cost Analysis to determine IRR and CAPTIM.

5. Sorting of project durations, IRR and CAPTIM into rank order and performance of statistical analysis thereon.

6. Writing of data to file PERTRA3.RES.

A flow chart and listing of PERTRA3 is given in Figure 5.16 and Appendix F.1. respectively.
PROGRAM PERTRA3
Risk Analysis using Monte Carlo simulation

Read number of simulations (NRUNS) and type of random generation RGCODE from keyboard.
Read data files name (FLNAME) from the file PERTRA.FLS

Read data from files generated by PERTRA1 and PERTRA2

Change project start date to day number
SUBROUTINE CALEND

Perform Time analysis calculations
SUBROUTINE ANALYS

Change project finish day numbers to calendar dates
SUBROUTINE CALEND

Perform project cost analysis
SUBROUTINE PCOSTS

IF NRUNS > 1 THEN
YES
ELSE
NO
END IF

Write results into the file PERTRA3.RES

STOP

Figure 5.16 Program PERTRA3 Flow Chart.
Calculate Triangular Distribution parameters for tasks with VARIABLE durations

SUBROUTINE TRIDIS

For each simulation i=1,NRUNS initialize the SEED

SUBROUTINE DSELECT

Perform time analysis for the selected tasks

SUBROUTINE ANALYS

Perform cost analysis

SUBROUTINE PCOSTS

SUBROUTINE FREQUA

STOP

For each simulation i=2,NRUNS:

BEGIN LOOP

Select an integer task duration

SUBROUTINE DSELECT

Perform time analysis for the selected tasks

SUBROUTINE ANALYS

Perform cost analysis

SUBROUTINE PCOSTS

End of loop

SUBROUTINE TRIDIS

SUBROUTINE FREQUA

SUBROUTINE PRSTAT

SUBROUTINE RRSTAT

SUBROUTINE FRQIRR

SUBROUTINE FRQCAP

Write data into the file PERTRA3.RES

STOP

Figure 5.16 (Continue)
5.4.1 Calendar Subroutines and Functions

In order to perform the time analysis of the project network, and to facilitate the performance of cash flow calculations, the following subroutines and functions have been included:

5.4.1.1 Subroutine CALEND

This subroutine converts an alphanumeric date (e.g. 05/DEC/92 — as generated by PERTMASTER) into an integer day number (relating to the 'Julian' day number commonly used in astronomy) and vice-versa, as follows:

(a) Conversion of date to day number:

(i) Conversion of alphanumeric date to numeric date ID, IM, IY:

for example: 05/DEC/92 → ID = 5; IM = 12; IY = 1992

(ii) Conversion of numeric date to day number using the following algorithm:

\[
\text{Day Number} = \text{INT}(365.25 \times Y) + \text{INT}(30.6001 \times M) + D
\]

where:
\[ Y = \begin{cases} Y - 1 & \text{if } IM = 1 \text{ or } 2 \\ Y & \text{if } IM > 2 \end{cases} \]

\[ M = \begin{cases} IM + 13 & \text{if } IM = 1 \text{ or } 2 \\ IM + 1 & \text{if } IM > 2 \end{cases} \]

(b) Conversion of day number to alphanumeric date:

(i) Conversion of day number to numeric date, ID, IM, IY, using the following algorithm:

\[
Y = \text{INT} \left[ \frac{\text{Day Number} - 122.1}{365.25} \right]
\]

\[
M = \text{INT} \left[ \frac{\text{Day Number} - \text{INT}(365.25 \times Y)}{30.6001} \right]
\]

\[ D = \text{Day Number} - \text{INT}(365.25 \times Y) - \text{INT}(30.6001 \times M) \]

where:

\[ IM = \begin{cases} M - 13 & \text{If } M = 14, 15 \\ M - 1 & \text{If } M < 14 \end{cases} \]

\[ IY = \begin{cases} Y & \text{If } IM > 2 \\ Y + 1 & \text{If } IM = 1, 2 \end{cases} \]

ID = D
(ii) Conversion of numeric date to alphanumerical date. For example: 02/09/1992 to 02/SEP/1992

Listing of the subroutine is given in Appendix F.2.

5.4.1.2 Function IDOTW

This function calculates the day of the week from the day number using the following algorithm:

\[
\text{Day of the Week} = 7 \times \text{MOD} \left( \frac{\text{Day Number} + 5}{7} \right)
\]

The day of the week is represented by integers 0 to 6 where 0 represents Sunday. For ease of calculation the function changes Sunday from 0 to 7. The listing of the function is given in Appendix F.3.

5.4.2 Subroutine ANALYS

This subroutine performs the time analysis of the network and calculates the following for each task:

(a) Earliest start day number;
(b) Earliest finish day number;
(c) Latest start day number;
(d) Latest finish day number;
It also calculates Hammock tasks durations and identifies critical tasks in the network.

Calculation of the above task schedules is performed in accordance with the following algorithms:

(a) Calculation of Earliest Start day number (ES):

1. If the link type is F → S;

   \[ ES(\text{succeeding task}) = EF(\text{preceding task}) + \text{Link Duration} \]

2. If the link type is S → S;

   \[ ES(\text{succeeding task}) = ES(\text{preceding task}) + \text{Link Duration} \]

(b) Calculation of Earliest Finish day number (EF):

1. If the link type is F → F;

   \[ EF(\text{succeeding task}) = EF(\text{preceding task}) + \text{Link Duration} \]

2. If the link type is S → S or F → S;

   \[ EF(\text{succeeding task}) = ES(\text{succeeding task}) + \text{Duration of succeeding task} \]
(c) Calculation of Latest Start day numbers (LS):

1. If the link type is S—S;

\[ \text{LS(preceding task)} = \text{LS(succeeding task)} - \text{Link Duration} \]

2. If the link type is F—F or F—S;

\[ \text{LS(preceding task)} = \text{LS(succeeding task)} - \text{Duration of preceding task} \]

(d) Calculation of Latest Finish day numbers (LF)

1. If the link type is F—F;

\[ \text{LF(preceding task)} = \text{LF(succeeding task)} - \text{Link Duration} \]

2. If the link type is F—S;

\[ \text{LF(preceding task)} = \text{LS(succeeding task)} - \text{Link Duration} \]

It should be noted that, since the task and link durations are specified in 'working' days, use of the above algorithms will result in the schedule dates being in the form of 'working' day numbers — which are of strictly limited use schedule-wise where a task's 'working' week differs from the standard 'calendar' week of 7 days.

To ensure that the above dates are in the form of 'actual' day numbers (as opposed to 'working' day numbers), the following procedures and
algorithms have been included in the subroutine:

(A) Forward Pass:

Tasks and/or link durations are adjusted to include for 'down time' by adding the number of non—working days (TLADD) given by the following algorithm:

\[
\text{TLADD} = \text{INT} \left[ \left( \frac{\text{DURN} + \text{IESDOW} - 2}{\text{TWK}} \right) \cdot (7 - \text{TWK}) \right]
\]

where;

DURN = Task or Link duration;
TWK = Task working days/week;
IESDOW = Link or Task Earliest start day of the week

Subsequently—calculated Earliest Start/Finish is adjusted to ensure that it will fall on a working day by adding the integer ISADD calculated as follows:

\[
\text{IF} \quad \text{IESDOW} > \text{TWK};
\]

\[
\text{ISADD} = 8 - \text{IESDOW}
\]

(B) Backward Pass:

The task and/or link duration (as the case may be) is adjusted to include
for 'down time' by adding the integer TLADD calculated by the following algorithm:

\[
TLADD = \text{INT} \left[ \frac{(\text{DURN} - \text{LFDO} + \text{TWK} - 1)}{\text{TWK}} \right] (7 - \text{TWK})
\]

where;

- DURN = Task or Link duration;
- LFDO = Task or Link Latest Finish day of the week;
- TWK = Task working days/week

Calculated Latest Finish/Start should be adjusted to fall in a working day by subtracting the integer ISDDCT calculated as follows:

\[
\text{If } \text{LFDO} > \text{TWK};
\]

\[
\text{ISDDCT} = \text{LFDO} - \text{TWK}
\]

**NOTE:**

Using of the above algorithms is varying in accordance with the type of link and the calculated schedule date, this variation can be seen in the listing of the subroutine ANALYS given in Appendix F.4.

The subroutine flow chart is shown in Figure 5.17.
SUBROUTINE ANALYS
Performs network time analysis calculations;
Calculates Hammock tasks durations;
Identifies Critical tasks in the network

Initialise project finish day number IPFDNO;
Identify the number of the first task NSTART;
Initialise Tasks Earliest start day number IESDNO;
Initialise tasks earliest finish day number IEFDNO

For each link record
in the Forward Pass

Calculate the absolute integer link duration
FUNCTION LNKDRN

According to the link type, calculate
target task earliest start day number
and earliest start day of the week number

Calculate, target task, earliest finish day number
and earliest finish day of the week number

NO
End of loop
YES

Identify the number of the last task NFINISH

Initialise tasks latest start and
latest finish day numbers

For each link record
in the Forward Pass

Calculate the absolute integer link duration
FUNCTION LNKDRN

According to the link type, calculate
target task earliest start day number
and earliest start day of the week number

Calculate, target task, earliest finish day number
and earliest finish day of the week number

YES
End of loop
NO

Calculate project finish day of the week number;
Adjust project finish day number IPFDNO

Calculate project duration IPRDRN

Calculate Hammock tasks durations

Update tasks criticality index ICRIT

RETURN

Figure 5.17 Subroutine ANALYS Flow Chart
5.4.2.1 Function LINDRN

Since the link duration in the case of S—S link type is given always by a percentage of the preceding task, this function is called by the subroutine ANALYS to change the (S—S) link type duration from a percentage to an absolute Integer duration. It also changes all other type of link duration to Integer durations. Listing of the function is given in Appendix F.5.

5.4.2.2 Function ISADD

This function is called by the subroutine ANALYS to calculate the number of non—working days, week—ends, to be added to the task Earliest start day number in order to prevent it from falling in a non—working day. Listing of the function is given in Appendix F.6.

5.4.2.3 Function ISDEDC

It is called by the subroutine ANALYS to calculate the number of non—working days to be deducted in order to prevent the calculated Latest Finish day number of falling in a non—working day. Listing of the function is given in Appendix F.7.
5.4.3 Subroutine \textit{TRIDIS}

Calculates the triangular distribution parameters for any given task with \textsc{VARIABLE} duration. It performs the calculation in accordance with the algorithms (1), (2), (3) in Section 4.4.1. The subroutine flow chart is shown in Figure 5.18. Listing of the subroutine is given in Appendix F.8.

![Subroutine TRIDIS Flow Chart](image)

Figure 5.18 Subroutine TRIDIS Flow Chart.
5.4.4 Subroutine DSELECT

Randomly selects an Integer task duration on the basis of the duration distribution parameters calculated by the subroutine TRIDIS. Listing of the subroutine is given in Appendix F.9.

5.4.4.1 Function RAND

This function is called by the subroutine DSELECT to return a random real number, calculated on the basis of a 'seed', in the range 0.0 to 1.0. The 'seed' requires initialisation prior to the function being called.

If a Repeatable series of random numbers is required, the 'seed' initialised in the main program should be a fixed number. If a non-repeatable series of random numbers is required, the seed is initialised by using the variable IHUN, hundred of a second, in the MICROSOFT FORTRAN compiler built-in function GETTIM(IHR, IMIN, ISEC, IHUND) and the 'seed' initialised should range from 0.0 < SEED < 259200. Listing of the function is given in Appendix F.10.
5.4.5 Subroutine PCOSTS

This subroutine performs the second part of the overall analysis: Cost Analysis. This is carried out as follows:

1. By distributing the tasks cost components on the project working duration.

2. Calculates the daily costs and adjust them according to the method of payment intervals and payment delays read from the cost file (FLNAME.CST) generated by the program PERTRA1.

3. Based on the above modified data, it calculates the project CAPTIM and Internal Rate of Return.

Steps of carrying out the above are shown on the subroutine flow chart in Figure 5.19. Listing of the subroutine is given in Appendix F.11.
SUBROUTINE PCOSTS
Distributes costs on the project days according to the payment method and delay;
Calculates the CAPTIM and IRR

Calculate project cost days (DAYS);
Distribute costs on the project days

For each Cost Component
Check payment method

NO

WEEKLY?

YES

Payment method is MONTHLY
Accumulate costs on the last day of the month
Accumulate costs on the last day of the week

Check Payment delay

NO

WEEKLY?

YES

Change the weeks into days and accumulate the payment

MONTHLY?

NO

YES

DAILY?

NO

No delay is required

Adjust payment not to be on weekends

GO to next cost component

YES

End of loop

Calculate total daily costs;
Accumulate total daily costs;
Accumulate total task prices

Calculate the CAPTIM
Calculate the IRR

RETURN

Figure 5.19 Subroutine PCOSTS Flow Chart.
5.4.6 Subroutine PSORT

Sorts the project durations, resulting from each 'simulation run', into rank order in order to perform statistical analysis on the results. Results are shown in Appendix. Listing of the subroutine is given in Appendix F.12.

5.4.7 Subroutine PRSTAT

Performs statistical analysis on the project durations after sorting them in rank order as shown in the flow chart in Figure 5.20. Listing of the subroutine is given in Appendix F.13.

5.4.8 Subroutine CASTAT

Performs the same steps of subroutine PRSTAT for the sorted CAPTIM results. Listing of the subroutine is given in Appendix F.14.

5.4.9 Subroutine RRSTAT

Performs the same steps of subroutines PRSTAT and CASTAT for the sorted IRR results. Listing of the subroutine is given in Appendix F.15.
SUBROUTINE PRSTAT
Performs statistical analysis to a number of runs NRUNS project durations

Calculate Minimum, Maximum, Sum and Sum square of project durations
MINPDU, MAXPDU, ISUM, ISUMSQ

Calculate Mean project duration PRMEAN and the Standard deviation PROSIG

Calculate Median project duration MEDIAN

Calculate the results distribution skewness PSKEW

RETURN

Figure 5.20 Subroutine PRSTAT Flow Chart.

5.4.10 Subroutine RRSORT

Sorts the values of Internal Rate of Return (IRR) and CAPTIM, as calculated by each 'simulation run', into rank order. Listing of the subroutine is given in Appendix F.16.

5.4.11 Subroutine FREQUA

Divides the number of runs of project durations into bands, calculate the frequency and cumulative frequency for each band and determine the bands midpoints to allow presenting the results in a graphical mode. Listing is given in Appendix F.17.
5.4.12 Subroutine FRQIRR

Performs the same steps as subroutine FREQUA, but, for the number of runs of IRR. The flow chart of the subroutine is shown in Figure 5.21. Listing of the subroutine is given in Appendix F.18.

5.4.13 Subroutine FRQCAP

Performs the same steps of subroutines FREQUA and FRQIRR for the CAPTIM results. Listing of the subroutine is given in Appendix F.19.

Figure 5.21 Subroutine FRQIRR Flow Chart.
6.1 **PERTRA1**

6.1.1 *Screen Displays*

With a view to ensuring that the program is user-friendly and straightforward to operate — two of the prime requirements of any computer program — PERTRA1 generates a number of screen displays to facilitate the interactive input of data from the keyboard and to keep the user abreast of exactly what the program is doing at any given time. Each screen display is of the form shown in Figure 6.1
6.1.1.1 Data Input Line

In cases where alphanumeric data is required to be input, the data input line will display details of the data required and will contain a pair of brackets [— ], containing a flashing cursor, indicating the maximum permissible length of the input string. Mistakes may be rectified in the normal way by using the 'backspace' key.

If the input string exceeds the maximum permissible length, a warning bleep is sounded.

In cases where the required input consists of selection of a number of options, the data input line will display a menu of available options highlighting the first character of each option. Selection of the required option is carried out by pressing the required key.
6.1.1.2 Screen Display Table

The screen display table has two functions, namely:

(a) To display introduction or termination information when entering or 'quitting' the program.

(b) To display, in tabular form (for clarity), data which has been input by the user.

6.1.1.3 Message Line

The message line displays either of two types of message:

(a) A message briefing the user (when appropriate) as to what the program is doing at any given time.

(b) A message containing information (when appropriate) pertaining to the input of data by the user.
6.1.2 Operational Details

6.1.2.1 Initialisation

Program operation is commenced, in the usual manner, by typing the program name PERTRA1.

The resultant screen display — shown in Figure 6.2 — indicates for the user to input the name (with no extension) of the PMA-generated export file which the program is required to read.

The input name is displayed as shown in Figure 6.3 (which assumes the input file name to be BRIDGE) and the user is given the option of confirming the file name or repeating the exercise in the event of an error in the input. The appropriate options are displayed in the data input line, selection of which is made by typing Y or N.
Program PERTRA1

Interactive initialisation program for the Monte-Carlo simulation carried out by Program PERTRA3

PMA Export File: BRIDGE.EXP

Figure 6.2 File Name Input Screen Display

OK? YES NO

Program PERTRA1

Interactive initialisation program for the Monte-Carlo simulation carried out by Program PERTRA3

PMA Export File: BRIDGE.EXP

Figure 6.3 File Name Confirmation Screen Display.
6.1.2.2 Reading of Data File

Typing the letter Y will result in the display of the data abstraction screen as shown in Figure 6.4. The status of the data abstraction will be changing on the screen as the program finishes the relevant data reading.

```
<table>
<thead>
<tr>
<th>RECORD TYPE</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEADER</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>PROJECT</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>TASK</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>LINK</td>
<td>IN PROGRESS</td>
</tr>
<tr>
<td>RESOURCE</td>
<td></td>
</tr>
<tr>
<td>CALENDAR</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 6.4 Data Abstraction 1 Screen Display
When all data is abstracted, a message (ALL DATA ABSTRACTED) will appear on the message line and the screen display will be as in Figure 6.5. At this stage the user is requested to hit any key for continuation.

![Data Abstraction Table]

<table>
<thead>
<tr>
<th>RECORD TYPE</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEADER</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>PROJECT</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>TASK</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>LINK</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>RESOURCE</td>
<td>COMPLETED</td>
</tr>
<tr>
<td>CALENDAR</td>
<td>COMPLETED</td>
</tr>
</tbody>
</table>

ALL DATA ABSTRACTED
Hit any key to continue

Figure 6.5 Data Abstraction 2 Screen Display
6.1.2.3 Input of Working Week Data

Hitting any key will result in the display of the working week data screen (see Figure 6.6). The number of working days for each displayed calendar should be input from the keyboard.

Figure 6.6 Working Week Input Screen Display
Finishing the input of all such data, the user is given the option of confirming the displayed data or repeating the exercise in the event of an error in such. These options are given in the confirmation screen shown in Figure 6.7.

![Figure 6.7 Working Week Confirmation Screen Display](image-url)
6.1.2.4 Input of Cost Component Data

Confirmation of the correctness of working-week data will result in the cost component data screen display as shown in Figure 6.8.

Code: [ ] (Hit <CR> if component not applicable)

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>LOADING</th>
<th>CODE</th>
<th>PAYMENT</th>
<th>PAYMENT DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABOUR</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MATERIALS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Back</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBCONTRACT</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TASK PRICE</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.8 Cost Components Data Screen Display
At this stage, the user is requested to input the cost component code as defined by PMA or hitting the <CR> key if the cost component is not applicable. If the input code is not defined, a message will appear in the message line informing the user that the cost component code is not found and should try again to input the correct code.

Screens display for the input of the CODE, PAYMENT (weekly or monthly) and PAYMENT DELAY (days or weeks or months) are shown in Figure 6.9, Figure 6.10 and Figure 6.11 respectively.

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>LOADING</th>
<th>CODE</th>
<th>PAYMENT</th>
<th>PAYMENT DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABOUR</td>
<td>Normal</td>
<td>XLC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MATERIALS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Back</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBCONTRACT</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TASK PRICE</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.9 Cost component Data Screen Display (2)
**Payment delay:** [ ] (Hit <CR> if no payment delay)

**Cost Component Data**

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>LOADING</th>
<th>CODE</th>
<th>PAYMENT CODE</th>
<th>PAYMENT DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABOUR</td>
<td>Normal</td>
<td>XLC</td>
<td>WEEKLY</td>
<td></td>
</tr>
<tr>
<td>MATERIALS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Back</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBCONTRACT</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TASK PRICE</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[e.g. 28d = 28 days; 6w = 6 weeks; 1m = 1 month;]

**Figure 6.10 Cost Component Data Screen Display (3)**

**Continue?** YES NO

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>LOADING</th>
<th>CODE</th>
<th>PAYMENT CODE</th>
<th>PAYMENT DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABOUR</td>
<td>Normal</td>
<td>XLC</td>
<td>WEEKLY</td>
<td></td>
</tr>
<tr>
<td>MATERIALS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANT</td>
<td>Back</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBCONTRACT</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Normal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Front</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TASK PRICE</td>
<td>Spread</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6.11 Cost Component Data Continuation Screen Display (1).**
By finishing the data input of a cost component as shown in Figure 6.11, the user is given the option of confirming the accuracy of the data input for the relevant cost component or repeating the exercise in the event of any error in the input. Selection of the option is made by typing Y or N.

The same process is continued for the other components bearing in mind that, for the cost components PLANT (FRONT and BACK) and OVERHEADS (FRONT), the user will not have the option for the input of their PAYMENT and PAYMENT DELAY as they will be assumed and displayed as the input of the NORMAL cost component for each of them.

Finishing the above, a continuation screen as in Figure 6.12 will be displayed given the user the option to continue or repeat the exercise once again. Continuation options is made by typing Y or N.

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>LOADING</th>
<th>CODE</th>
<th>PAYMENT</th>
<th>PAYMENT DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABOUR</td>
<td>Normal</td>
<td>XLC</td>
<td>WEEKLY</td>
<td>-</td>
</tr>
<tr>
<td>MATERIALS</td>
<td>Front</td>
<td>XMC</td>
<td>MONTHLY</td>
<td>28 days</td>
</tr>
<tr>
<td>PLANT</td>
<td>Normal</td>
<td>XPC</td>
<td>MONTHLY</td>
<td>1 month</td>
</tr>
<tr>
<td>PLANT</td>
<td>Front</td>
<td>XPF</td>
<td>MONTHLY</td>
<td>1 month</td>
</tr>
<tr>
<td>PLANT</td>
<td>Back</td>
<td>XPB</td>
<td>MONTHLY</td>
<td>1 month</td>
</tr>
<tr>
<td>SUBCONTRACT</td>
<td>Spread</td>
<td>XSC</td>
<td>MONTHLY</td>
<td>4 weeks</td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Normal</td>
<td>XOD</td>
<td>MONTHLY</td>
<td>-</td>
</tr>
<tr>
<td>OVERHEADS</td>
<td>Front</td>
<td>XOF</td>
<td>MONTHLY</td>
<td>-</td>
</tr>
<tr>
<td>TASK PRICE</td>
<td>Spread</td>
<td>XTP</td>
<td>MONTHLY</td>
<td>1 month</td>
</tr>
</tbody>
</table>

Figure 6.12 Cost Component Continuation Display (2).
6.1.2.5 Generation of Data Files and Program Termination

Typing the option Y will result in the 'termination' screen display shown in Figure 6.13 which will show the data files written and to be used by the programs PERTRA2 and PERTRA3.

When all the data is written to the relevant files a termination message will appear in the message line.

![Termination Display Screen](image)

Figure 6.13 Termination Display Screen
6.2 PERTRA2

6.2.1 Screen Displays

PERTRA2 generates the same screen types as generated by the program PERTRA1 (see Section 6.1.1).

6.2.2 Operational Details

6.2.2.1 Program Introduction

Program operation is commenced, in the usual manner, by typing the program name PERTRA2.

The resultant is the 'introductory' screen shown in Figures 6.14 to 6.16 which displays information about PERTRA2.
Program PERTRA2

Interactive input of TASK DURATION data for the Monte-Carlo simulation carried out by Program PERTRA3.

Basic task data is read from the appropriate .TSK file generated by Program PERTRA1. Following input of the task duration data, this file will be updated, thus providing the facility of performing intermediate checks on the data file.

For convenience, PERTRA2 facilitates SELECTIVE display of tasks on the basis of:

(a) Task NAME;
(b) Task WORK BREAKDOWN STRUCTURE;
(c) Task TYPE;

Hit any key to continue

Figure 6.14 Program Introductory Screen 1

Program PERTRA2

For the purpose of data input, tasks may be categorised as follows:

TYPE 1: Those whose durations are VARIABLE in accordance with a probability distribution defined by:

(a) OPTIMISTIC duration (99% exceedance probability);
(b) MOST LIKELY duration (used in PMA calculations);
(c) PESSIMISTIC duration (1% exceedance probability);

TYPE 2: Those whose durations are INVARIABLE and are not subject to a probability distribution as above;

Hit any key to continue

Figure 6.15 Program Introductory Screen 2
Program PERTRA2

Note 1: MILESTONE tasks will automatically be included within category 2.

Note 2: By virtue of their nature, HAMMOCK tasks will also be automatically included within category 2.

Hit any key to continue

Figure 6.16 Program Introductory Screen 3
6.2.2.2 Reading Task Duration Data

Hitting any key after the program introduction result in the display of reading data screen as shown in Figure 6.17 which is briefing the user of what the program is doing at that stage.

<table>
<thead>
<tr>
<th>Program PERTRA2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note 1: MILESTONE tasks will automatically be included within category 2;</td>
</tr>
<tr>
<td>Note 2: By virtue of their nature, HAMMOCK tasks will also be automatically included within category 2;</td>
</tr>
</tbody>
</table>

Reading Task Duration data from file: BRIDGE.TSK

Figure 6.17 Reading Data Screen

6.2.2.3 Data Input

Automatically the program will display the task duration data screen as shown in Figure 6.18 giving the user the option to input the task duration data or quit the program. The appropriate option is selected by typing the
letter I or Q. Selection of the INPUT option results in the screen display shown in Figure 6.19 giving the user the option to select the type of task duration to input (VARIABLE/INVARIABLE). The option is selected in the usual manner by typing the one of the highlighted letters V or I.

Option: INPUT QUIT

Figure 6.18 Task Duration Data Screen 1

Task Duration: VARIABLE/INVARIABLE

Figure 6.19 Task Duration Data Screen 2
6.2.2.4 Input of Tasks with VARIABLE Durations

Selecting the option V will result in the display of the screen shown in Figure 6.20 giving the user the option of confirming the input of task duration data for tasks with VARIABLE durations (see Section 5.3.(a)) by typing the letter Y or going back to the previous screen by typing the letter N and repeating the exercise again.

Figure 6.20 Tasks Duration Screen 3
6.2.2.4.1 Filtration Parameters

Continuing will result in the display of the first filtration parameters screen as shown in Figure 6.21. The user is given the option of selecting one of the following filtration parameters (see Section 5.3.2):

1. Task Name (NAME);
2. Task Type (TYPE);
3. Task Work Breakdown Structure (WBS);

or the option of quitting the input of data (QUIT).

Figure 6.21 First Filter Parameters Screen 1
In case of selection of the code TYPE, the user will not be required to input the parameter POS as, the program will automatically display, as shown in Figure 6.22, the number 1 highlighting the third filtration parameter OPERAND giving the user the options equal (EQ) or not equal (NE).

Figure 6.22 First Filter Parameters Screen 2
Selection of the option EQ by typing the highlighted letter E will result in the display of the SELECTION as shown in Figure 6.23 and the user is required to input the type of tasks to be selected. Typing mistakes can be corrected by using the backspace key as indicated in the message line.

Tasks with VARIABLE durations

Figure 6.23 First Filter Parameters screen 3
Typing the task type, for example, T and hitting the <CR> key will result in the display of the second filter parameters screen shown in Figure 6.24.

<table>
<thead>
<tr>
<th>Filter</th>
<th>Code</th>
<th>Position</th>
<th>Operator</th>
<th>Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TYPE</td>
<td>1</td>
<td>EQ</td>
<td>T</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Logic:**

Figure 6.24 Second Filter Parameters Screen 1
Repeating the same exercise, for example, selection of all tasks with work breakdown structure, starting from the fourth character, not equal to DELAY will result in the display of the screen shown in Figure 6.25.

Tasks with VARIABLE durations

<table>
<thead>
<tr>
<th>FILTER</th>
<th>CODE</th>
<th>POS</th>
<th>OPRN</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TYPE</td>
<td>1</td>
<td>EQ</td>
<td>T</td>
</tr>
<tr>
<td>2</td>
<td>WBS</td>
<td>4</td>
<td>NE</td>
<td>DELAY</td>
</tr>
</tbody>
</table>

LOGIC:

Figure 6.25 Second Filter Parameters Screen 5
Following the input of the requisite number of filtration parameters, up to a maximum of six, the QUIT option should be chosen to allow the user to input the logic parameters.

6.2.2.4.2 Filtration Parameters Logic

At this stage (see Figure 6.26), the user is giving the option of confirming the displayed parameters (Y) or repeating the exercise (N) in the event of an error being detected.

<table>
<thead>
<tr>
<th>Filter</th>
<th>Code</th>
<th>Position</th>
<th>Operator</th>
<th>Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TYPE</td>
<td>1</td>
<td>EQ</td>
<td>T</td>
</tr>
<tr>
<td>2</td>
<td>WBS</td>
<td>4</td>
<td>NE</td>
<td>DELAY</td>
</tr>
</tbody>
</table>

CONTINUE? YES NO

Figure 6.26 Filters Confirmation Screen
If Y is selected, the logic input screen (see Figure 6.27) is displayed, giving the user the option of input AND or OR. If AND is chosen, each of the displayed filtration criteria must be met for a task to be selected. If OR is chosen, only one of the displayed criteria must be met for a task to be selected.

Tasks with VARIABLE durations

<table>
<thead>
<tr>
<th>FILTER</th>
<th>CODE</th>
<th>POS</th>
<th>OPRN</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TYPE</td>
<td>1</td>
<td>EQ</td>
<td>T</td>
</tr>
<tr>
<td>2</td>
<td>WBS</td>
<td>4</td>
<td>NE</td>
<td>DELAY</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.27 Filters Logic Screen
For this example, both of the filtration criteria should be met and selection of the option AND will result in the screen display shown in Figure 6.28.

Tasks with VARIABLE durations

Figure 6.28 Filters Logic Confirmation Screen
Selection of the option CANCEL will delete the previously selected filtration parameters and logic and the exercise is to be repeated again. Selection of the option GO will result in the display of the selected tasks as shown in the sample given in Figure 6.29.

**Figure 6.29 Sample of Tasks With VARIABLE Duration Display**
6.2.2.43 Input of Optimistic and Pessimistic Durations Factors

Confirming the accuracy of the displayed tasks will result in the display of the screen shown in Figure 6.30 and the input of the Optimistic duration factor (see Section 5.3.4) will be highlighted. The data input line is showing the range of the factor to input.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>WBS</th>
<th>DURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S4UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
<tr>
<td>S5BCONC</td>
<td>T</td>
<td>RW-CONC</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BEXC</td>
<td>T</td>
<td>RW-EWK</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BFILL</td>
<td>T</td>
<td>RW-EWK</td>
<td>7.00</td>
</tr>
<tr>
<td>S5BFW</td>
<td>T</td>
<td>RW-FW</td>
<td>2.00</td>
</tr>
<tr>
<td>S5BFWS</td>
<td>T</td>
<td>RW-FW</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BRF</td>
<td>T</td>
<td>RW-RF</td>
<td>4.00</td>
</tr>
<tr>
<td>S5UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
<tr>
<td>S6BCONC</td>
<td>T</td>
<td>RW-CONC</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BEXC</td>
<td>T</td>
<td>RW-EWK</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BFILL</td>
<td>T</td>
<td>RW-EWK</td>
<td>7.00</td>
</tr>
<tr>
<td>S6BFW</td>
<td>T</td>
<td>RW-FW</td>
<td>2.00</td>
</tr>
<tr>
<td>S6BFWS</td>
<td>T</td>
<td>RW-FW</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BRF</td>
<td>T</td>
<td>RW-RF</td>
<td>4.00</td>
</tr>
<tr>
<td>S6UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
</tbody>
</table>

OPTIMISTIC duration factor: H
PESSIMISTIC duration factor: [ ]

Figure 6.30 Optimistic Duration Factor Input Screen
Typing the Optimistic Duration factor and hitting the <CR> key will result in the display of the screen shown in Figure 6.31, indicating the input of the Pessimistic Duration factor and showing the range of the factor in the data input line.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>WBS</th>
<th>DURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S4UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
<tr>
<td>S5BCONC</td>
<td>T</td>
<td>RW-CONC</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BEXC</td>
<td>T</td>
<td>RW-EWK</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BFILL</td>
<td>T</td>
<td>RW-EWK</td>
<td>7.00</td>
</tr>
<tr>
<td>S5BFW</td>
<td>T</td>
<td>RW-FW</td>
<td>2.00</td>
</tr>
<tr>
<td>S5BFWS</td>
<td>T</td>
<td>RW-FW</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BRF</td>
<td>T</td>
<td>RW-RF</td>
<td>4.00</td>
</tr>
<tr>
<td>S5UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
<tr>
<td>S6BCONC</td>
<td>T</td>
<td>RW-CONC</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BEXC</td>
<td>T</td>
<td>RW-EW</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BFILL</td>
<td>T</td>
<td>RW-EWK</td>
<td>7.00</td>
</tr>
<tr>
<td>S6BFW</td>
<td>T</td>
<td>RW-FW</td>
<td>2.00</td>
</tr>
<tr>
<td>S6BFWS</td>
<td>T</td>
<td>RW-FW</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BRF</td>
<td>T</td>
<td>RW-RF</td>
<td>4.00</td>
</tr>
<tr>
<td>S6UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
</tbody>
</table>

OPTIMISTIC duration factor: .5000
PESSIMISTIC duration factor: [Redacted]

Figure 6.31 Pessimistic Duration Factor Input Screen
The user is, once again, having the option of confirming the accuracy of the data input as can be seen in the screen display shown in Figure 6.32.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>WBS</th>
<th>DURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S4UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
<tr>
<td>S5BCONC</td>
<td>T</td>
<td>RW-CONC</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BEXG</td>
<td>T</td>
<td>RW-EWK</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BFILL</td>
<td>T</td>
<td>RW-EWK</td>
<td>7.00</td>
</tr>
<tr>
<td>S5BFW</td>
<td>T</td>
<td>RW-FW</td>
<td>2.00</td>
</tr>
<tr>
<td>S5BFWS</td>
<td>T</td>
<td>RW-FW</td>
<td>1.00</td>
</tr>
<tr>
<td>S5BRF</td>
<td>T</td>
<td>RW-RF</td>
<td>4.00</td>
</tr>
<tr>
<td>S5UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
<tr>
<td>S6BCONC</td>
<td>T</td>
<td>RW-CONC</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BEXC</td>
<td>T</td>
<td>RW-EW</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BFILL</td>
<td>T</td>
<td>RW-EWK</td>
<td>7.00</td>
</tr>
<tr>
<td>S6BFW</td>
<td>T</td>
<td>RW-FW</td>
<td>2.00</td>
</tr>
<tr>
<td>S6BFWS</td>
<td>T</td>
<td>RW-FW</td>
<td>1.00</td>
</tr>
<tr>
<td>S6BRF</td>
<td>T</td>
<td>RW-RF</td>
<td>4.00</td>
</tr>
<tr>
<td>S6UNITS</td>
<td>T</td>
<td>RW-UNITS</td>
<td>3.00</td>
</tr>
</tbody>
</table>

OPTIMISTIC duration factor: 0.5000
PESSIMISTIC duration factor: 2.0000

Figure 6.32 Factors Confirmation Screen
6.2.2.5 Input of Tasks with INVARİABLE durations

The selection of the tasks with INVARİABLE durations, for example, concrete curing, consolidation of an embankment under surcharge etc. (see Section 5.3.(b)), is carried out in the same manner as for the selection of tasks with variable durations, but the input of optimistic and pessimistic durations factors is not required. The result of the selection exercise is shown in Figure 6.33.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>WBS</th>
<th>DURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1BFWS</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S2BFWS</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S3BFWS</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S4BFWS</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S5BFWS</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S6BFWS</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
</tbody>
</table>

Figure 6.33 Filtered Tasks display screen
Confirming the accuracy of the displayed tasks will result in the display of the screen shown in Figure 6.34. The user is required to hit any key and the program will code all the displayed tasks accordingly.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>WBS</th>
<th>DURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1FWSD</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S2FWSD</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S3FWSD</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S4FWSD</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S5FWSD</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
<tr>
<td>S6FWSD</td>
<td>T</td>
<td>RW-DELAY</td>
<td>5.00</td>
</tr>
</tbody>
</table>

Tasks will be coded accordingly
Hit any key to continue

Figure 6.34 Coding Screen Display
6.2.2.6 Updating of Task Data File and Program Termination

Finishing the coding the program automatically will update the Task data file (see Section 5.3.5) and the screen shown in Figure 6.35 will be displayed with a program termination message in the message line.

![Program Termination Screen](image)

Figure 6.35 Program Termination Screen
6.3 PERTRA3

6.3.1 Program Operation

PERTRA3 is considered to be the "number-crunching" program in the suite of the PERTRA programs and its interaction with the user is very limited. For this reason the program was written without the use of the screen displays used in the programs PERTRA1 and PERTRA2.

Program operation is commenced, in the usual manner, by typing the program name PERTRA3.

6.3.2 Data Input

6.3.2.1 Random Numbers series Type

The user is required to input the type of random number series to be generated by the program.

If a REPEATABLE series of random numbers is required (see Section 5.4.4.1), the letter R should be typed followed by \(<\text{CR}>\). If a NONREPEATABLE series is required, the letter N should be typed, followed by \(<\text{CR}>\).

In the event of typing any letters other than R or N, a warning message will be displayed and the correct code should be typed again.
6.3.2.2 Number of Simulations

The next data input required is the number of simulation runs required (see Section 4.3.1). The program is capable of performing up to 1000 runs.

Following the input of the above data the program commences the analysis calculations and writes the results in the file PERTRA3.RES.

By means of appropriate screen messages, the user is kept abreast of the operations currently being performed by the program.

6.3.3 Data Output

6.3.3.1 One Simulation Run Data Output

If a single simulation run is chosen (see above), the following results will be written to the file PERTRA3.RES.

(a) Project Start Date.

(b) Project Finish Date.

(c) Project Duration (days)

(d) Project Internal Rate of Return (IRR).
63.3.2 Multiple Simulation Runs Output

If more than one simulation run is required, the results will be written into the file PERTRA3.RES as given in the sample in Appendix H.

Presenting the results graphically can be carried out using any interactive graphics software. The output shown in Figures 6.36 to 6.41 was achieved using the software package: HARVARD GRAPHICS.
Figure 6.36 Project Durations Frequency Diagram

Figure 6.37 Project durations Cumulative Frequency Diagram
Figure 6.38 Internal Rate of Return (IRR) Frequency Diagram

Figure 6.39 IRR Cumulative Frequency Diagram
Figure 6.40 CAPTIM Frequency diagram

Figure 6.41 CAPTIM Cumulative Frequency Diagram
CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

The usual techniques of including the most likely duration in the analysis and producing a single most likely result tends to underestimate the risks in a project. It neglect most of the information that bears in future possibilities. For this reason, risk analysis programs have been developed to report a range of outcomes for the project; including possible worse outcomes as well as desired profitable ones.

By their very nature, risk analysis programs require substantial computing power. In the past, this restricted their use to large organisations and projects. However, the last decade has seen rapid improvements in the capabilities of microcomputers to the extent that it became feasible to develop risk analysis programs for use on personal computers.

The PERTRA programs were therefore developed to quantify the effect of risks on cost and duration of projects. However it should be emphasized that the results are based on estimates to tasks durations (optimistic, pessimistic and most likely) and the accuracy and the confidence in it are very well depend
upon the original estimates.

7.2 Recommendations

As the computer results are usually taken to be accurate and the confidence in these results are very high, it should be emphasised here that, especially for the Cost Analysis case, this results should be taken as a guide and the following should be taken into consideration to increase the confidence in these results:

1. The analysis is based only on logical linkage between tasks and does not take into consideration the rescheduling of tasks, for the purpose of maximising the efficiency of resource usage, which usually takes place in reality, therefore a mechanism to consider that should be incorporated in the development of the analysis routines.

2. The costs used by the program are based on the tasks demand for resource while in reality the majority of resource costs are based on the supply thereof. It is therefore recommended that such a facility be included in the program.

3. The inclusion of project close-down and seasonal variations (see Section 2.5.2 and Section 2.5.5).

4. The inclusion of a facility to consider the project maintenance period in the cost analysis subroutines.
5. A facility to take account of the retention money should be included.

6. A facility to consider the effects of the inflation and particularly the rise in the prices during the contract period is to be developed.

7. **FORTRAN 77** is a programming language recommended for the "number-crunching" programs, for example **PERTRA3**, but not for the programs which deals extensively with character operations as the case of **PERTRA1** and **PERTRA2**.

8. **FORTRAN** programs restricting the user of specifying the arrays dimensions before compiling and linking this limits the program only to the project in hand, therefore, a second thought about the programming language is to be considered.

9. In the case of using the **FORTRAN** language the compilation of the program **PERTRA3** using the compiler **SALFORD** is recommended because of the advantages of using the graphics support of this compiler which allows the presentation of the results in graphics, on the screen or in hard copy, directly. The reason for not using the above mentioned compiler is its availability at a later stage of this project.
REFERENCES


188
Appendix A: Contents of PMA—Generated Export File.

1. General Format.

The general order of records contained within the export file is as follows:

Header Record

Project Record

Task Record
Output Record
Task Resource Demand Record 1
Task Resource Demand Record 2
. . . .
Task Resource Demand Record N

Task Record
Output Record
Task Resource Demand Record 1
Task Resource Demand Record 2
. . . .
Task Resource Demand Record N

. . . .

Task Record
Output Record
Task Resource Demand Record 1
Task Resource Demand Record 2
. . . .
Task Resource Demand Record N

Task Record
Output Record
Task Resource Demand Record 1
Task Resource Demand Record 2
. . . .
Task Resource Demand Record N

Link Record (Link 1)
Link Record (Link 2)
. . . .
. . . .
. . . .
. . . .
Link Record (Link N)
2. Record Format

The contents and format of the data contained within each type of record are as follows, with data required by PERTRA programs being annotated *:

**Header Records**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>Record Type</td>
</tr>
<tr>
<td>Plan Name</td>
<td>Up to 9 characters</td>
</tr>
<tr>
<td>Left Header *</td>
<td>Quote delimited</td>
</tr>
<tr>
<td>Right Header *</td>
<td>Quote delimited</td>
</tr>
<tr>
<td>Work Breakdown</td>
<td>Up to 12 characters</td>
</tr>
<tr>
<td>Plan Type</td>
<td>Single character (A = arrow, P = precedence)</td>
</tr>
<tr>
<td>Units Number</td>
<td>(Unused)</td>
</tr>
<tr>
<td>Calendar Name</td>
<td>Up to 8 characters</td>
</tr>
<tr>
<td>Codes Name</td>
<td>Up to 8 characters (Unused)</td>
</tr>
<tr>
<td>Subtime Units</td>
<td>Number between 1 and 80</td>
</tr>
<tr>
<td>Version Number *</td>
<td>Number between 0 and 999</td>
</tr>
<tr>
<td>Plan Start Date</td>
<td>Date string (&quot;DD/MMM/YY[:SS]&quot;)</td>
</tr>
<tr>
<td>Plan Finish Date</td>
<td>Date string (&quot;DD/MMM/YY[:SS]&quot;)</td>
</tr>
<tr>
<td>Time— now</td>
<td>Date string (&quot;DD/MMM/YY[:SS]&quot;)</td>
</tr>
</tbody>
</table>

(Note: In date strings [:ss] refers to subtime units if they are in use within the plan. The brackets[ ] refer to an optional item and are not present in the field.)
Example of Header Record:

H,"BRIDGE","Construction Management IV","Model Solution","N— NN— ",P,0,"DAILY" ,","1,0," 7/JUN/92","12/APR/93"," 7/JUN/92

Project — defines each sub-project within the plan.

P Record type
Project Name * Up to 9 characters
Parent Name Up to 9 characters
Project Header * Quote delimited
Project Start Date * Date string ("DD/MMM/YY[ :SS]")
Project Finish Date Date string ("DD/MMM/YY[ :SS]")

Example of Project Record:

P,"BRIDGE",",""Road over Road Bridge"," 7/JUN/92","12/APR/93"

Task — defines a task as an input by the user.

T/M/K/X/S * Record type — Task, Milestone, Hammock, Sub-project
Project Name Up to 8 characters
Task Name * Up to 9 characters — Quote delimited
Task Description Quote delimited
Work Breakdown
Structure * Up to 12 characters
Original Duration * Unsigned integer
Remaining Duration Unsigned integer
Preferred Start Date String ("DD/MMM/YY[:SS]")
Scheduled Start Date String ("DD/MMM/YY[:SS]")
Scheduled Finish Date String ("DD/MMM/YY[:SS]")
Actual Start Date String ("DD/MMM/YY[:SS]")
Actual Finish Date String ("DD/MMM/YY[:SS]")
Target Start Date String ("DD/MMM/YY[:SS]")
Target Finish Date String ("DD/MMM/YY[:SS]")
Calendar Name * Up to 8 characters
Percent Complete Unsigned integer as a string

Note: Percent complete only appears if percent complete is set 'Enter'

Example of Task Records:


Output Record – defines the schedule of the task as calculated by PERTMASTER Advance

Z Record type
Project Name Up to 9 characters
Task Name Up to 9 characters – Quote delimited
Calculated Preferred Start Date String ("DD/MMM/YY[:SS]")
Preferred Finish
Early Start
Early Finish
Late Start
Late Finish
Total Preferred
Start Float
Total Preferred
End Float
Free Preferred
Start Float
Free Preferred
End Float
Derived Duration
Percent Complete
Start Task *
Finish Task *

Example of Output Record:

Z,"BRIDGE","CLEAR1"," 7/JUN/92"," 8/JUN/92"," 7/JUN/92"," 8/JUN/92"," 8/JUN/92
"," 9/JUN/92"," 1"," 1"," 0"," 0"," 2","0
","0","0"
Task Resource Demand – gives details of the demand for resources made by the task as defined by the user.

D

Record type

Project Name
Up to 8 characters

Task Name
Up to 9 characters – Quote delimited

Resource Code *
Up to 3 characters

Resource Demand *
Double precision number

Example of Task Resource Demand Records:

D, "BRIDGE", "CLEAR1", "DRT", 1
D, "BRIDGE", "CLEAR1", "LAB", 2
D, "BRIDGE", "CLEAR1", "TIP", 1

Link – gives details of precedence logic links as defined by the user

L

Record type

Project Name
Up to 9 characters

Preceding Task *
Name 9 characters

Succeeding Task *
Name up to 9 characters

Link Type *
3 characters – See Note below

Link Duration *
Unsigned integer

Note: The link type appears in the file as "S—S", "F—S", "S—F", "F—F".
Example of Link Records:

<table>
<thead>
<tr>
<th>Resource Code</th>
<th>Resource Name</th>
<th>Resource Type</th>
<th>Storage Type</th>
<th>Resource Calendar</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;BRIDGE&quot;</td>
<td>&quot;CLEAR1&quot;</td>
<td>&quot;SETUP&quot;</td>
<td>&quot;F-S&quot;</td>
<td>&quot;0&quot;</td>
</tr>
<tr>
<td>&quot;BRIDGE&quot;</td>
<td>&quot;CLEAR2&quot;</td>
<td>&quot;FINISH&quot;</td>
<td>&quot;F-S&quot;</td>
<td>&quot;0&quot;</td>
</tr>
<tr>
<td>&quot;BRIDGE&quot;</td>
<td>&quot;CMPRESS&quot;</td>
<td>&quot;SSEPCO&quot;</td>
<td>&quot;F-F&quot;</td>
<td>&quot;0&quot;</td>
</tr>
</tbody>
</table>

Resource Definition — defines the resources in use on the plan as defined by the user

<table>
<thead>
<tr>
<th>R</th>
<th>Record type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Code *</td>
<td>Up to 3 characters</td>
</tr>
<tr>
<td>Resource Name</td>
<td>Quote delimited</td>
</tr>
<tr>
<td>Resource Type *</td>
<td>Single character (D/F/B/S)</td>
</tr>
<tr>
<td>Storage Type</td>
<td>Number describing whether limited, unlimited, actual,cost or budget</td>
</tr>
<tr>
<td>Resource Calendar</td>
<td>Up to 8 characters</td>
</tr>
</tbody>
</table>

Note: Character corresponds respectively to an integer value. D= Daily, F = Front loading, B = Back loading, and S = Spread.

Example of Resource Definition Records:

R,"621","Cat 621 Scraper",D,0,"ALLDAYS"
Resource Supply — defines the availability for each resource as defined by the user.

S
Resource Code
Up to 3 characters
Time Offset
Date string("DD/MMM/YY[:SS]")
Supply Quantity
Type defined by 'storage type' above

Subresource Supply — defines demands made by one resource for another in a resource hierarchy as defined by the user.

U
Record type
Resource Code *
Up to 3 characters
Subresource Code *
Up to 3 characters
Resource Demand *
Double precision number

Example of Sub-resource Supply

U,"BFG","B35",1
U,"BFG","JCB",2
U,"BFG","LAB",4
U,"BFG","TIP",2
Resource Demand — defines the demand for each resource as calculated by PERTMASTER ADVANCE.

Q Record type

Resource Code Up to 3 characters

Start of Demand Date String ("DD/MMM/YY[ :SS]")

End of Demand Date String ("DD/MMM/YY[ :SS]")

Demand Level Double precision number

Example of Resource Demand Records:

Q,"BFG","20/JUN/91","20/JUN/91",1
Q,"BFG","4/JUL/91","4/JUL/91",1
Q,"BFG","3/SEP/91","3/SEP/91",2

Calendar — gives details of the calendars in use on the plan as defined by the user.

C Record type

Calendar Name * Up to 9 characters

Example of Calendar Records:

C,"ALLDAYS"
C,"DAILY"
**Working Week** — defines the typical working week.

**W** Record type  
Calendar Name Up to 9 characters  
Working Week spec. Two integers separated by commas describing start to finish of a working week spec. in subunits.

**Example of Working Week Record:**

\[ W, \text{DAILY}, 1, 5 \]

**Holiday** — defines a holiday period within the last mentioned calendar as defined by the user. There will be a list of these records for each calendar.

**O** Record type  
Calendar Name Up to 8 characters  
Start Date Date String ("DD/MMM/YY[:SS]")  
End Date Date String ("DD/MMM/YY[:SS]")

**Working Day** — Defines a work day period within a calendar as defined by the user. There will normally be a list of these records for each calendar.

**Y** Record type  
Calendar Name Up to 8 characters  
Start Date Date string ("DD/MMM/YY[:SS]")  
End Date Date string ("DD/MMM/YY[:SS]")

199
APPENDIX B: Samples of Data Files Generated by PERTRA1

Appendix B.1: Sample of the general data written to the file FLNAME.GEN

<table>
<thead>
<tr>
<th>BRIDGE 0</th>
<th>(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case Study No. 1</td>
<td>(2)</td>
</tr>
<tr>
<td>Road over Road Bridge</td>
<td>(3)</td>
</tr>
<tr>
<td>Construction Management IV</td>
<td>(4)</td>
</tr>
<tr>
<td>15/APR/91</td>
<td>(5)</td>
</tr>
</tbody>
</table>

Definition of rows:

(1): Project Name; Project Version.
(2): Left Header.
(3): Project description.
(4): Right Header.
(5): Project Start Date.
Appendix B.2: Sample of the Task Data written to the File FLNAME.TSK

274  (Number of tasks in the project)

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>T</td>
<td>CLEAR1</td>
<td>GN-CLEAR</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0.00</td>
<td>2.00</td>
</tr>
<tr>
<td>T</td>
<td>CLEAR2</td>
<td>GN-CLEAR</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0.00</td>
<td>5.00</td>
</tr>
<tr>
<td>H</td>
<td>CMPRESS</td>
<td>GN-CMPRESS</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0.00</td>
<td>136.00</td>
</tr>
<tr>
<td>T</td>
<td>CPC1CO</td>
<td>PR-CONC</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>T</td>
<td>CPC1D1</td>
<td>PR-DELAY</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Definition of Columns are:

(1): Task Type.
(2): Task Name.
(3): Task Work Breakdown Structure.
(4): Number of Working Days/Week
(5): Task Code (Start, Finish or Normal Task)
(6): Dummy Array to be Used Later for Variable/Nonvariable duration Task Codes.
(7): Dummy array to be Used Later for Optimistic Durations.
(8): Task Duration (Most Likely Duration).
(9): Dummy Array to be Used Later for Pessimistic Durations.
Appendix B.3: Sample of Link Data written to the File FLNAME.LNK

<table>
<thead>
<tr>
<th>437</th>
<th>(Number of links)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
<th>(8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>268</td>
<td>1</td>
<td>1</td>
<td>.00</td>
<td>158</td>
<td>40</td>
<td>3</td>
<td>.00</td>
</tr>
<tr>
<td>268</td>
<td>158</td>
<td>1</td>
<td>.00</td>
<td>2</td>
<td>40</td>
<td>1</td>
<td>.00</td>
</tr>
<tr>
<td>1</td>
<td>209</td>
<td>1</td>
<td>.00</td>
<td>268</td>
<td>158</td>
<td>1</td>
<td>.00</td>
</tr>
<tr>
<td>158</td>
<td>40</td>
<td>3</td>
<td>.00</td>
<td>270</td>
<td>2</td>
<td>1</td>
<td>.00</td>
</tr>
<tr>
<td>209</td>
<td>49</td>
<td>1</td>
<td>.00</td>
<td>96</td>
<td>2</td>
<td>1</td>
<td>.00</td>
</tr>
</tbody>
</table>

Definition of Columns:

(1): Source Task Number (Forward Pass).
(2): Target Task Number (Forward Pass).
(3): Link Type (Forward Pass).
(4): Link Duration (Forward Pass).

Columns 5 to 8 is the same as the above, but, for the Backward pass.
Appendix B.4: Sample of the Cost Data written to the file FLNAME.CST

<table>
<thead>
<tr>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>XLC W 0 0</td>
<td>XMC M M 1</td>
<td>XPC M M 1</td>
<td>XPF M M 1</td>
</tr>
<tr>
<td>XPB M M 1</td>
<td>XSC M M 1</td>
<td>XOD M 0 0</td>
<td>XOF M 0 0</td>
</tr>
<tr>
<td>XTP M M 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The above four columns definitions is as follows:

(1): Cost component
(2): Payment Interval (W= Weeks, M= Months)
(3): Delay of payment (D= Days, W= Weeks, M= Months, 0= No delay)
(4): Number of Months or Weeks of delay.

<table>
<thead>
<tr>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
<th>(8)</th>
<th>(9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>143.00</td>
<td>.00</td>
<td>200.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>797.06</td>
</tr>
<tr>
<td>238.00</td>
<td>.00</td>
<td>200.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
</tr>
<tr>
<td>.00</td>
<td>.00</td>
<td>40.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
</tr>
<tr>
<td>145.00</td>
<td>209.79</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>381.19</td>
</tr>
<tr>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
<td>.00</td>
</tr>
</tbody>
</table>

Definition of the above columns is as follows:

(9): Task Price.
APPENDIX C: Program PERTRA1 and Subroutines

Appendix C.1 Program PERTRA1

C******************************************************************************************
C** Program PERTRA1.FOR **
C** **
C** Reads project data from user-specified export file **
C** FLNAME.EXP generated by PERTMASTER ADVANCE **
C** **
C** Writes selected data to files for subsequent use by **
C** Risk Analysis Programs PERTRA2.FOR and PERTRA3.FOR **
C** **
C** GENERAL data: FLNAME.GEN **
C** TASK data: FLNAME.TSK **
C** LINK data: FLNAME.LNK **
C** COST data: FLNAME.CST **
C** **
C** Writes FLNAME to file PERTRA.FLS for subsequent use **
C** **
C******************************************************************************************
C
COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400),TRES(400,10),
&TRESQ(400,10),TCAL(400),NTRES(400),SFCODE(400),TWK(400),
&CTRES(400,10),TCOST(400,9),TWBS(400)
COMMON/LINKS/LFROM(500,2),LTO(500,2),LTYPE(500,2),
&LDURN(500,2)
COMMON/RES/RNAME(100),RTYPE(100),RCOST(100)
COMMON/SUBRES/CRTOP(110),CRSUB(110),RSUBQ(110),RSUB(110),
&RTOP(110)
COMMON/CALS/CAL(5),ICWK(5)
COMMON/PROJ/PHEAD(3),PVER,PNAME, PSTART
COMMON/COSTS/CCODE(9),CPAY(9),CARR(9),CARRQ(9)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC, IDM

C
CHARACTER*1  TTYPE, RTYPE, C1,PVER, CPAY, CARR
CHARACTER*3  RNAME, CTRES, CRTOP, CRSUB, RCODE, CCODE
CHARACTER*8  CAL, TCAL, PNAME
CHARACTER*9  TNAME, PSTART
CHARACTER*12  FLNAME, TWBS
CHARACTER*30  PHEAD
CHARACTER*185  REC

C
REAL  LDURN
INTEGER  SFCODE,TWK, TRES, RTOP, RSUB, CARRQ

C
C...Set TEXT mode

C
CALL SETMODE(0)
C...Display introduction; Input file name from keyboard
C
CALL HEADER(FLNAME,NCHRFN)
C
C...Set maximum value of counters according to dimensions
C...Note: Dimensions must be exactly divisible into 64K
C
ITMAX=400
ILMAX=500
IRMAX=100
IUMAX=110
ICMAX=5
IDMAX=10
C
C...Check adequacy of dimensions
C
WRITE(6,28)
28 FORMAT(/,/," Carrying out preliminary check on dimensions")
CALL DIMCHK(FLNAME)
ICODE=0
WRITE(6,30) IT
30 FORMAT( /, ' No. of TASKS: ',113)
IF(IT.GT.ITMAX)THEN
WRITE(6,40)
40 FORMAT(' Dimensions exceeded !!')
IDIM=(IT-MOD(IT,50))+50
WRITE(6,50)IDIM
50 FORMAT(' Increase dimensions to ',13,' and re-compile')
ICODE=1
ENDIF
WRITE(6,60)IDM
60 FORMAT(/, ' Max. No. of TR DEMANDS: ',113)
IF(IDM.GT.IDMAX)THEN
WRITE(6,40)
IDIM=(IDM-MOD(IDM,5))+5
WRITE(6,50)IDIM
ICODE=1
ENDIF
WRITE(6,70)IL
70 FORMAT(/, ' No. of LINKS: ',113)
IF(IL.GT.ILMAX)THEN
WRITE(6,40)
IDIM=(IL-MOD(IL,50))+50
WRITE(6,50)IDIM
ICODE=1
ENDIF
WRITE(6,80)IR
80 FORMAT(/, ' No. of RESOURCES: ',113)
IF(IR.GT.IRMAX)THEN
WRITE(6,40)
IDIM=(IR-MOD(IR,10))+10
WRITE(6,50)IDIM
ICODE=1
ENDIF

WRITE(6,90) IU
90 FORMAT(/,' No. of SUB-RESOURCES: ',15)
   IF(IU.GT.IUMAX) THEN
      WRITE(6,40)
      IDIM=(IU-MOD(IU,10))+10
      WRITE(6,50) IDIM
      ICODE=1
   ENDIF
   WRITE(6,100) IC
100 FORMAT(/,' No. of CALENDARS: ',19)
   IF(IC.GT.ICMAX) THEN
      WRITE(6,40)
      IDIM=(IC-MOD(IC,5))+5
      WRITE(6,50) IDIM
      ICODE=1
   ENDIF
   IF(ICODE.LT.1) THEN
      WRITE(6,102)
      102 FORMAT(/,',' ALL DIMENSIONS OK !!')
   ELSE
      GO TO 999
   ENDIF

C...Read data from FLNAME
C
   CALL READEXP(FLNAME)
C
C...Determine working week ICWK for each calendar
C
   CALL CALWKS
C
C...Convert TRD resource names into resource indices
C
   DO 410 I=1,IT
      IF(NTRES(I).LT.1) GO TO 410
      DO 400 K=1,NTRES(I)
         DO 390 L=1,IR
            IF(CTRES(I,K).EQ.RNAME(L)) THEN
               TRES(I,K)=L
               GO TO 400
            ENDIF
            390 CONTINUE
         ENDIF
         400 CONTINUE
      ENDIF
      410 CONTINUE

C...Convert SR resource names into resource indices
C
   DO 450 I=1,IU
      DO 420 J=1,IR
         IF(CRTOP(I).EQ.RNAME(J)) THEN
            RTOP(I)=J
            GO TO 430
         ENDIF
      ENDIF
      420 CONTINUE
   450 CONTINUE
IF(CRSUB(I).EQ.RNAME(J)) THEN
  RSUB(I)=J
  GO TO 450
ENDIF
440 CONTINUE
450 CONTINUE
C
C...Calculate cost components for each task
C
CALL TCOSTS
C
C...Define durations of NON-ZERO S-S LINKS as fractions of
C...SOURCE TASK durations;
C
DO 600 I=1,IL
  IF(LTYPE(I,1).EQ.2.AND.LDURN(I,1).NE.0.0)THEN
    J=LFROM(I,1)
    LDURN(I,1)=LDURN(I,1)/TDURN(J)
  ENDIF
600 CONTINUE
C
C...Sort links into FORWARD/BACKWARD PASS order
C
CALL LSORT
C
C...Write data to datafiles; display concluding message;
C
CALL QUIT(FLNAME,NCHRFN)
999 END
SUBROUTINE HEADER(FLNAME,NCHR)

c*****************************************************************************
c Subroutine HEADER
     Introductory display for program PERTRA1
     Keyword input of PMA export file name FLNAME
*****************************************************************************
C
CHARACTER*1 CHR1
CHARACTER*12 FLNAME
CHARACTER*42 TABREC(15),BLANK
C
C...Initialise display table strings
C
DO 20 I=1,15
   TABREC(I)(1:1)=CHAR(186)
   TABREC(I)(42:42)=CHAR(186)
DO 10 J=2,41
   TABREC(I)(J:J)=' '     
10  CONTINUE
20  CONTINUE
BLANK=TABREC(1)
BLANK(1:1)= ' '     
BLANK(42:42)= ' '     
TABREC(1)(1:1)=CHAR(201)
TABREC(1)(42:42)=CHAR(201)
TABREC(5)(1:1)=CHAR(204)
TABREC(5)(42:42)=CHAR(204)
TABREC(11)=TABREC(5)
TABREC(15)(1:1)=CHAR(200)
TABREC(15)(42:42)=CHAR(200)
DO 30 J=2,41
   TABREC(I)(J:J)=CHAR(205)
30  CONTINUE
TABREC(3)(14:28)='Program PERTRA1'
TABREC(7)(3:40)='Interactive initialisation program for'
TABREC(8)(3:40)='the Monte-Carlo simulation carried out'
TABREC(9)(13:30)='by Program PERTRA3'
TABREC(13)(8:23)='PMA Export File:
C
C...Specify option highlight delay
C
NDEL=30
C
C...Define table position
C
    NLTL=6
    NCTL=20
    NLTOP=NLTL-1
    NCTOP=NCTL
C
C...Clear screen; draw table
C
    CALL SCLEAR
    CALL TXTCLR(15,1)
    DO 40 I=1,15
       CALL TMOVETO((NLTL+I-1),NCTL)
       IF(I.LT.2.OR.I.GT.4) THEN
           CALL OUTEXT(TABREC(I))
       ELSE
           CALL OUTEXT(TABREC(I)(1:1))
           CALL TXTCLR(0,6)
           CALL OUTEXT(TABREC(I)(2:41))
           CALL TXTCLR(15,1)
           CALL OUTEXT(TABREC(I)(42:42))
       ENDIF
    40 CONTINUE
C
C...Display FILE NAME input line
C
    50 CALL TXTCLR(15,0)
    CALL TMOVETO(NLTOP,NCTOP)
    CALL OUTEXT(BLANK)
    CALL TMOVETO(NLTOP,NCTOP)
    CALL OUTEXT('File name (no extension): ['\]')
C
C...Read file name from keyboard;
C...convert to upper case if required;
C
    NCHR=8
    FLNAME=''
    CALL CREAD(NLTOP,(NCTOP+27),FLNAME,NCHR)
    CALL CHRCNV(FLNAME)
C
C...Remove unwanted spaces from FLNAME
C
    FLNAME((NCHR+1):(NCHR+4))='.EXP'
C
C...Update table
C
    CALL TXTCLR(7,1)
    CALL TMOVETO((NLTL+12),(NCTL+24))
    CALL OUTEXT(FLNAME)
C Display CONTINUATION line; highlight
C first character of options;

C CALL TXTCLR(15,0)
CALL TMVOETO(NLTOP,NCTOP)
CALL OUTEXT(BLANK)
CALL TMVOETO(NLTOP,NCTOP)
CALL OUTEXT('OK? YES NO')
CALL TXTCLR(4,0)
CALL TMVOETO(NLTOP,(NCTOP+5))
CALL OUTEXT('Y')
CALL TMVOETO(NLTOP,(NCTOP+10))
CALL OUTEXT('N')

C Hide cursor

C CALL TXTCLR(0,0)
CALL TMVOETO(25,70)
CALL OUTEXT(' ')
CALL TMVOETO(25,70)

C Read option from keyboard; convert to upper case if required;

60 CALL KBREAD(0,CHR1)
CALL CHRCNV(CHR1)
IF(CHR1.EQ.'Y')THEN

C Option is CONTINUE; highlight option; exit routine

CALL TXTCLR(0,7)
DO 70 M=1,NDEL
     CALL TMVOETO(NLTOP,(NCTOP+5))
     CALL OUTEXT('YES')
70 CONTINUE
GO TO 999
ELSEIF(CHR1.EQ.'N')THEN

C Option is DISCONTINUE; highlight option;
C out table entry; Input correct file name;

CALL TXTCLR(0,7)
DO 80 M=1,NDEL
     CALL TMVOETO(NLTOP,(NCTOP+10))
     CALL OUTEXT('NO')
80 CONTINUE
CALL TXTCLR(0,1)
CALL TMVOETO((NLTL+12),(NCTL+24))
CALL OUTEXT(BLANK(1:12))
GO TO 50
ELSE

210
C
C...Inappropriate character; try again
C
    CALL SPCHAR(1)
    GO TO 60
    ENDIF
C
C...Exit routine
C
  999 CALL TXTCLR(7,0)
    CALL SCLEAR
    RETURN
    END
Appendix C.3: Subroutine DIMCHK

SUBROUTINE DIMCHK(FLNAME)
  SUBROUTINE DIMCHK
  ** Subroutine DIMCHK
  ** Performs check on adequacy of PERTRA1/PERTRA2 dimensions
  **
  COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IDM
  CHARACTER*1 Cl
  CHARACTER*12 FLNAME

  C...Initialise control statements
  ASSIGNED 20 TO LABIF

  C...Zero array counters
  IT=0
  IL=0
  IR=0
  IU=0
  IC=0
  IDM=0

  C...Open file on unit 3
  OPEN(UNIT=3,FILE=FLNAME)

  C...Skip first two records
  READ(3,15)Cl
  READ(3,15)Cl

  C...Read 1st character of record
  READ(3,15)Cl
  10 FORMAT(A1)
  GO TO LABIF

  C...Check if TASK record
  IF(Cl.EQ.'T'.OR.Cl.EQ.'M'.OR.Cl.EQ.'K'.OR.Cl.EQ.'S')THEN

  C...TASK record; Update counter
  IT=IT+1

  C...Zero TASK RESOURCE DEMAND counter
  ID=0
C...Skip OUTPUT record; Read next record
C
READ(3,15)C1
GO TO 10
C
C...Check if TASK RESOURCE DEMAND record
C
ELSEIF(C1.EQ.'D')THEN
C
C...TASK RESOURCE DEMAND record; Update counter
C
ID=ID+1
IF(ID.GT.IDM)IDM=ID
GO TO 10
ENDIF
C
C...Check if LINK record
C
30 IF(C1.EQ.'L')THEN
C
C...LINK record; All TASK type records abstracted;
C...Change label LABIF to avoid unnecessary testing
C
ASSIGN 30 TO LABIF
C
C...Update counter;
C
IL=IL+1
GO TO 10
ENDIF
C
C...Check if RESOURCE record
C
40 IF(C1.EQ.'R')THEN
C
C...RESOURCE record; All TASK/LINK records abstracted;
C...Change label LABIF to avoid unnecessary testing;
C
ASSIGN 40 TO LABIF
C
C...Update counter;
C
IR=IR+1
GO TO 10
C
C...Check if (unrequired) RESOURCE DEMAND record
C
ELSEIF(C1.EQ.'Q'.OR.C1.EQ.'S')THEN
C
C...Unrequired record; Read next record
C
GO TO 10

213
C...Check if SUB-RESOURCE record
C
ELSEIF(C1.EQ.'U')THEN
C
C...SUB-RESOURCE record; Update counter
C
IU=IU+1
GO TO 10
ENDIF
C
C...Check if CALENDAR record
C
50 IF(C1.EQ.'C')THEN
C
C...CALENDAR record; All TASK/LINK/RESOURCE records abstracted;
C...Change label LABIF to avoid unnecessary testing
C
ASSIGN 50 TO LABIF
C
C...Update counter;
C
IC=IC+1
GO TO 10
ELSE
C
C...All data abstracted; Close file
C
CLOSE(3)
ENDIF
C
C...Exit
C
RETURN
END
Appendix C.4: Subroutine READEXP

SUBROUTINE READEXP(FLNAME)
C***********************************************************************
C** Subroutine READEXP **
C** Reads data from user-specified export file FLNAME **
C** generated by PMA **
C***********************************************************************

COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400), TRES(400,10),
& TRESQ(400,10), TCAL(400), NTRES(400), SFCODE(400), TWK(400),
& CTRES(400,10), TCOST(400,9), TWBS(400)
COMMON/LINKS/LFROM(500,2), LTO(500,2), LTYPE(500,2),
& LDURN(500,2)
COMMON/RES/RNAME(100), RTYPE(100), RCOST(100)
COMMON/SUBRES/CRTOP(110), CRSUB(110), RSUBQ(110), RTOP(110),
& RSUB(110)
COMMON/CALS/CAL(5), ICWK(5)
COMMON/PROJ/PHEAD(3), PVER, PNAME, PSTART
COMMON/RECORD/REC, ICOMMA(16)
COMMON/COUNTS/IT, ID, IL, IR, IU, IC, IDM
C
CHARACTER*1 TTYPE, RTYPE, CI, PVER
CHARACTER*3 RNAME, CTRES, CRTOP, CRSUB
CHARACTER*8 CAL, TCAL, PNAME
CHARACTER*9 TNAME, PSTART
CHARACTER*11 STATUS(2)
CHARACTER*12 FLNAME, TWBS
CHARACTER*28 TABREC(12)
CHARACTER*30 PHEAD
CHARACTER*185 REC
C
REAL LDURN
INTEGER SFCODE, TWK, TRES, RTOP, RSUB
C
C...Initialise display table records
C
DO 510 I=1,12
   TABREC(I)(1:1)=CHAR(186)
   TABREC(I)(28:28)=CHAR(186)
DO 500 J=2,27
   TABREC(I)(J:J)=CHAR(205)
500 CONTINUE
510 CONTINUE
   TABREC(1)(1:1)=CHAR(201)
   TABREC(1)(28:28)=CHAR(187)
   TABREC(12)(1:1)=CHAR(200)
   TABREC(12)(28:28)=CHAR(188)
DO 520 J=2,27
   TABREC(1)(J:J)=CHAR(205)
   TABREC(12)(J:J)=CHAR(205)
   CONTINUE
215
CONTINUE
TABREC(1)(7:22)='Data Abstraction'
TABREC(3)(3:13)='RECORD TYPE'
TABREC(3)(19:24)='STATUS'
TABREC(5)(4:9)='HEADER'
TABREC(6)(4:10)='PROJECT'
TABREC(7)(4:7)='TASK'
TABREC(8)(4:7)='LINK'
TABREC(9)(4:11)='RESOURCE'
TABREC(10)(4:11)='CALENDAR'
STATUS(1)='IN PROGRESS'
STATUS(2)='COMPLETED'

C...Draw display table (BRIGHT WHITE on BLUE)
C
CALL SCLEAR
CALL TXTCLR(15,1)
NLTL=5
NCTL=27
DO 530 I=1,12
   CALL TMOVETO((NLTL+I-1),NCTL)
   CALL OUTEXT(TABREC(I))
530 CONTINUE

C...Set text colour for 'STATUS' entries (WHITE on BLUE)
C
CALL TXTCLR(7,1)

C...Initialise control statements
C
ASSIGN 80 TO LABIF
ASSIGN 90 TO LABTW
ASSIGN 130 TO LABLW
ASSIGN 170 TO LABRW
ASSIGN 210 TO LABCW

C...Zero array counters
C
IT=0
IL=0
IR=0
IU=0
IC=0

C...Open export file FLNAME on unit 3
C
OPEN(UNIT=3,FILE=FLNAME)

C...Read HEADER record
C
READ(3,20)REC
20 FORMAT(A185)
   CALL TMOVETO((NLTL+4),(NCTL+15))
   CALL OUTEXT(STATUS(1))
C Determine comma positions in record
C
CALL RECCOM(10)
C
Zero character variable
C
DO 50 I=1,3
   DO 40 J=1,30
      PHEAD(I)(J:J)=''
      40 CONTINUE
   50 CONTINUE
C
Abstract character variables
C
Left header
C
PHEAD(1)=REC((ICOMMA(2)+2):(ICOMMA(3)-2))
C
Right header
C
PHEAD(3)=REC((ICOMMA(3)+2):(ICOMMA(4)-2))
C
Project version number
C
PVER=REC((ICOMMA(10)+1):(ICOMMA(10)+1))
CALL TMOVETO((NLTL+4),(NCTL+15))
CALL OUTEXT(STATUS(2))
CALL SPCHAR(1)
C
Read PROJECT record
C
READ(3,20)REC
CALL TMOVETO((NLTL+5),(NCTL+15))
CALL OUTEXT(STATUS(1))
C
Determine comma positions in record
C
CALL RECCOM(5)
C
Zero character variable
C
PNAME=''
C
Abstract character variable
C
Project Name
C
PNAME=REC(4:(ICOMMA(2)-2))
C
Project Description
C
PHEAD(2)=REC((ICOMMA(3)+2):(ICOMMA(4)-2))
C
C...Project Start Date
C
  PSTART=REC((ICOMMA(4)+2):(ICOMMA(5)-2))
  CALL TMOVETO((NLTL+5),(NCTL+15))
  CALL OUTEXT(STATUS(2))
  CALL SPCHAR(1)
C
C...Read record and abstract 1st character
C
  70  READ(3,20)REC
    C1=REC(1:1)
    GO TO LABIF
C
C...Check if TASK record
C
  80  IF(C1.EQ.'T'.OR.C1.EQ.'M'.OR.C1.EQ.'K'.OR.C1.EQ.'S')THEN
C
C...TASK record; Write screen message on FIRST task record only
C
    GO TO LABTW
    90  CALL TMOVETO((NLTL+6),(NCTL+15))
    CALL OUTEXT(STATUS(1))
    ASSIGN 110 TO LABTW
C
C...Update counter; Abstract data
C
    110  IT=IT+1
    CALL ABTASK
C
C...Zero TASK RESOURCE DEMAND counter and read next record
C
    ID=0
    GO TO 70
C
C...Check if TASK RESOURCE DEMAND record
C
    ELSEIF(C1.EQ.'D')THEN
C
C...TRD record; Update counter; Abstract data and read next record
C
    ID=ID+1
    NTRES(IT)=ID
    CALL ABTRD
    GO TO 70
    ENDIF
C
C...Check if LINK record
C
    120  IF(C1.EQ.'L')THEN
C
C...LINK record; All TASK type records abstracted;
C...Change label LABIF to avoid unnecessary testing
C
    ASSIGN 120 TO LABIF
...Write screen message on FIRST link record only

130 CALL TMOVETO((NLTL+6),(NCTL+15))
CALL OUTEXT(STATUS(2))
CALL SPCHAR(1)
CALL TMOVETO((NLTL+7),(NCTL+15))
CALL OUTEXT(STATUS(1))
ASSIGN 150 TO LABLW

...Update counter; Abstract data and read next record

150   IL=IL+1
CALL ABLINK
GO TO 70
ENDIF

...Check if RESOURCE record

160 IF(C1.EQ.'R')THEN

...RESOURCE record; All TASK/LINK records abstracted;
...Change label LABIF to avoid unnecessary testing;

ASSIGN 160 TO LABIF

...Write screen message on FIRST resource record only

170 CALL TMOVETO((NLTL+7),(NCTL+15))
CALL OUTEXT(STATUS(2))
CALL SPCHAR(1)
CALL TMOVETO((NLTL+8),(NCTL+15))
CALL OUTEXT(STATUS(1))
ASSIGN 190 TO LABRW

...Update counter; Abstract data and read next record

190   IR=IR+1
CALL ABRES
GO TO 70

...Check if (unrequired) RESOURCE DEMAND record

ELSEIF(C1.EQ.'Q'.OR.C1.EQ.'S')THEN

...Unrequired record; Read next record

GO TO 70

...Check if SUB-RESOURCE record

ELSEIF(C1.EQ.'U')THEN
C...S-R record; Update counter; Abstract data and read next record
C
    IU=IU+1
    CALL ABSUBRES
    GO TO 70
ENDIF
C
C...Check if CALENDAR record
C
200 IF(C1.EQ.'C')THEN
C
C...CALENDAR record; All TASK/LINK/RESOURCE records abstracted;
C...Change label LABIF to avoid unnecessary testing
C
    ASSIGN 200 TO LABIF
C
C...Write screen message on FIRST calendar record only
C
    GO TO LABCW
210 CALL TMOVETO((NLTL+8),(NCTL+15))
    CALL OUTEXT(STATUS(2))
    CALL TMOVETO((NLTL+9),(NCTL+15))
    CALL OUTEXT(STATUS(1))
    ASSIGN 230 TO LABCW
C
C...Update counter; Abstract data and read next record
C
230 IC=IC+1
    CALL ABCAL
    GO TO 70
ELSE
C
C...All data abstracted; Close file
C
    CLOSE(3)
ENDIF
C
C...Display concluding message
C
    CALL TMOVETO((NLTL+9),(NCTL+15))
    CALL OUTEXT(STATUS(2))
    CALL TXTCLR(15,1)
    CALL TMOVETO((NLTL+12),NCTL)
    CALL OUTEXT(' ALL DATA ABSTRACTED ')
    CALL SPCHAR(1)
    CALL TXTCLR(0,4)
    CALL TMOVETO((NLTL+13),NCTL)
    CALL OUTEXT(' Hit any key to continue ')
    CALL TXTCLR(0,0)
    CALL TMOVETO(25,70)
    CALL OUTEXT(' ')
    CALL TMOVETO(25,70)
    CALL KBREAD(0,C1)
Appendix C.5: Subroutine RECCOM

SUBROUTINE RECCOM(NCOMMA)

C **********************
C** Subroutine RECCOM **
C** Determines comma positions in PMA export file record REC **
C** NCOMMA = Number of commas in record **
C**********************

COMMON/RECORD/REC,ICOMMA(16)
CHARACTER*185 REC

J=1
DO 10 I=1,185
   IF(REC(I:I).NE.',')GO TO 10
   ICOMMA(J)=I
   J=J+1
   IF(J.GT.NCOMMA)GO TO 20
10 CONTINUE

20 RETURN
END
SUBROUTINE ABTASK

Subroutine ABTASK

Abstracts TASK data from export file record REC

Abstracts START/FINISH code from OUTPUT record

START task: SFCODE = 0
NORMAL task: SFCODE = 1
FINISH task: SFCODE = 2

COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400),TRES(400,10),
&TRESQ(400,10),TCAL(400),NTRES(400),SFCODE(400),TWK(400),
&CTRES(400,10),TCOST(400,9),TWBS(400)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IDM

CHARACTER*1 TTYPE,SCODE,FCODE
CHARACTER*3 CTRES
CHARACTER*8 TCAL
CHARACTER*9 TNAME
CHARACTER*10 FMT
CHARACTER*12 TWBS
CHARACTER*185 REC

INTEGER SFCODE,TWK,TRES

C...Determine comma positions in record
CALL RECCOM(14)

C...Zero character variables
TNAME(IT)=''
TCAL(IT)=''
SFCODE(IT)=''N''

C...Abstract character variables
C
C...Task Type
IF(REC(1:1).EQ.'K')THEN
   TTYPE(IT)=''H''
ELSE
   TTYPE(IT)=REC(1:1)
ENDIF
C...Task Name
C    TNAME(IT)=REC((ICOMMA(2)+2):(ICOMMA(3)-2))
C
C...Task Work Breakdown Structure
C    TWBS(IT)=REC((ICOMMA(4)+2):(ICOMMA(5)-2))
C
C...Task Calendar
C    TCAL(IT)=REC((ICOMMA(14)+1):(ICOMMA(14)+8))
C
C...Abstract numerical variables
C
C...Task Duration
C    NX=ICOMMA(6)+1
    NC=ICOMMA(7)-(1)-(NX+1)
    WRITE(FMT,10)NX,NC
10 FORMAT('(','12,'X','I','13,''))
    READ(REC,FMT)IDURN
    TDURN(IT)=REAL(IDURN)
C
C...Read OUTPUT record
C    READ(3,20)REC
20 FORMAT(A185)
C
C...Abstract START/FINISH code
C    CALL RECCOM(16)
    SCODE=REC((ICOMMA(15)+2):(ICOMMA(15)+2))
    FCODE=REC((ICOMMA(16)+2):(ICOMMA(16)+2))
    SFCODE(IT)=1
    IF(SCODE.EQ.'1')THEN
        SFCODE(IT)=0
    ELSEIF(FCODE.EQ.'1')THEN
        SFCODE(IT)=2
    ENDIF
C
C...Exit
C
RETURN
END
Appendix C.7: Subroutine ABTRD

SUBROUTINE ABTRD

**************************************************************************
** Subroutine ABTRD
**************************************************************************

** Abstracts TASK RESOURCE DEMAND data from export file record REC **
**************************************************************************

COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400),TRES(400,10),
&TRESQ(400,10),TCAL(400),NTRES(400),SFCODE(400),TWK(400),
&CTRES(400,10),TCOST(400,9),TWBS(400)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IDM

CHARACTER*1 TTYPE
CHARACTER*3 CTRES
CHARACTER*8 TCAL
CHARACTER*9 TNAME
CHARACTER*11 NUMB
CHARACTER*12 TWBS
CHARACTER*185 REC

INTEGER SFCODE,TWK,TRES

...Determine comma positions in record
CALL RECCOM(4)

...Zero character variables
CTRES(IT,ID)="'

...Abstract character variables

...Resource Name
CTRES(IT,ID)=REC((ICOMMA(3)+2):(ICOMMA(4)-2))

...Abstract numerical variables

...Resource Quantity
C...Check whether INTEGER or FLOATING POINT number
NUMB=REC((ICOMMA(4)+1):(ICOMMA(4)+11))
DO 30 I=1,11
   IF(NUMB(I:I).EQ."'"'.')THEN
C
C...FLOATING POINT number; Abstract using 'F' format
C
READ(NUMB,10)TRESQ(IT,ID)
10 FORMAT(F11.4)
GO TO 40
ELSEIF(NUMB(1:1).EQ. ' ' ) THEN

C
C...INTEGER number; Abstract using 'I'
C...format and convert to FP number
C
READ(NUMB,20)IRESQ
20 FORMAT(I6)
TRESQ(IT,ID)=REAL(IRESQ)
GO TO 40
ENDIF
30 CONTINUE

C
C...Exit
C
40 RETURN
END
Appendix C.8: Subroutine ABLINK

SUBROUTINE ABLINK

C**************************************************************************
C** Subroutine ABLINK **
C** Abstracts LINK data from export file record REC **
C** Outputs SOURCE task (LFROM) and TARGET task (LTO) as task INDEX **
C** Outputs Link Type (LTYPE) in INTEGER form **
C** F-S: LTYPE = 1 **
C** S-S: LTYPE = 2 **
C** F-F: LTYPE = 3 **
C** S-F: LTYPE = 4 **
C**************************************************************************

C**************************************************************************
C COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400), TRES(400,10),
& TRESQ(400,10), TCAL(400), NTRES(400), SFCODE(400), TWK(400),
& CTRES(400,10), TCOST(400,9), TWBS(400)
C**************************************************************************

C**************************************************************************
C COMMON/LINKS/LFROM(500,2), LTO(500,2), LTYPE(500,2),
& LDURN(500,2) COMMON/RECORD/REC, ICOMMA(16)
C COMMON/COUNTS/IT, ID, IL, IR, IU, IC, IDM
C**************************************************************************

CHARACTER*1  TTYPE
CHARACTER*3 CLTYPE, CTRES
CHARACTER*8 FMT, TCAL
CHARACTER*9 TNAME, CLFROM, CLTO
CHARACTER*12 TWBS
CHARACTER*185 REC

REAL LDURN
INTEGER SFCODE, TWK, TRES

C...Determine comma positions in record
CALL RECCOM(5)

C...Zero character variables
CLFROM=''
CLTO=''

C...Abstract character variables

C...Source task
CLFROM=REC((ICOMMA(2)+2):(ICOMMA(3)-2))
C...Target task
C
    CLTO=REC((ICOMMA(3)+2):(ICOMMA(4)-2))
C
C...Link type
C
    CLTYPE=REC((ICOMMA(4)+2):(ICOMMA(5)-2))
C
C...Abstract link duration as a numerical variable
C
    NX=ICOMMA(5)+1
    J=NX+3
    WRITE(FMT,20)NX
20 FORMAT(12,'X,13')
21 READ(REC,FMT,ERR=22)IDURN
    GO TO 23
22 IF(REC(J:J).EQ.' 'THEN
    FMT(7:7)=121
    ELSE
    FMT(7:7)=11
    ENDIF
    GO TO 21
23 LDURN(IL,1)=REAL(IDURN)
C
C...Change SOURCE/TARGET task names to task indices
C
    DO 30 I=1,IT
    IF(CLFROM.EQ.TNAME(I))THEN
        LFROM(IL,1)=I
    GO TO 40
    ENDIF
30 CONTINUE
40 DO 50 I=1,IT
    IF(CLTO.EQ.TNAME(I))THEN
        LTO(IL,1)=I
    GO TO 60
    ENDIF
50 CONTINUE
C
C...Change Link Type to an integer
C
    IF(CLTYPE.EQ.'F-S')THEN
        LTYPE(IL,1)=1
    GO TO 70
    ELSEIF(CLTYPE.EQ.'S-S')THEN
        LTYPE(IL,1)=2
    GO TO 70
    ELSEIF(CLTYPE.EQ.'F-F')THEN
        LTYPE(IL,1)=3
    GO TO 70
    ELSE
        LTYPE(IL,1)=4
    ENDIF
Appendix C.9: Subroutine ABRES

SUBROUTINE ABRES

** Subroutine ABRES **

** Abstracts RESOURCE data from export file record REC **

COMMON/RES/RNAME(100),RTYPE(100),RCOST(100)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IDM

COMMON/RES/RNAME(100),RTYPE(100),RCOST(100)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IDM

CALL RECCOM(5)

CALL RECCOM(5)

C...Zero character variables

RNAME(IR)=''

C...Resource name

RNAME(IR)=REC((ICOMMA(1)+2):(ICOMMA(2)-2))

C...Resource type

RTYPE(IR)=REC((ICOMMA(3)+1):(ICOMMA(3)+1))

C...Exit

RETURN

END

RETURN

END
Appendix C.10: Subroutine ABSUBRES

SUBROUTINE ABSUBRES
C******************************************************************
C** Subroutine ABSUBRES **
C** Abstracts SUB-RESOURCE data from export file record REC **
C******************************************************************
C
COMMON/SUBRES/CRTOP(110),CRSUB(110),RSUBQ(110),RTOP(110),
&RSUB(110)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IM
C
CHARACTER*3 CRTOP,CRSUB
CHARACTER*11 NUMB
CHARACTER*185 REC
C
INTEGER RTOP,RSUB
C
C...Determine comma positions in record
C
CALL RECCOM(3)
C
C...Zero character variables
C
CRTOP(IU)=''
CRSUB(IU)=''
C
C...Abstract character variables
C
C...Resource
C
CRTOP(IU)=REC((ICOMMA(1)+2):(ICOMMA(2)-2))
C
C...Sub-resource
C
CRSUB(IU)=REC((ICOMMA(2)+2):(ICOMMA(3)-2))
C
C...Abstract numerical variable
C
C...Sub-resource quantity
C...Check whether INTEGER or FLOATING POINT number
C
NUMB=REC((ICOMMA(3)+1):(ICOMMA(3)+11))
DO 30 I=1,11
  IF(NUMB(I:I).EQ.'')THEN
C
C...FLOATING POINT number; Abstract using 'F' format
C
READ(NUMB,10)RSUBQ(IU)
10 FORMAT(F11.4)
GO TO 40
ELSEIF(NUMB(I:I).EQ.' ')THEN
C
C...INTEGER number; Abstract using 'I'
C...format and convert to FP number
C
READ(NUMB,20)ISUBQ
20 FORMAT(16)
RSUBQ(IU)=REAL(ISUBQ)
GO TO 40
ENDIF
30 CONTINUE
C
C...Exit
C
40 RETURN
END

Appendix C.11: Subroutine ABCAL

SUBROUTINE ABCAL
**
** Subroutine ABCAL
**
** Abstracts CALENDAR data from export file record REC **
**
**
COMMON/CALS/CAL(5),ICWK(5)
COMMON/RECORD/REC,ICOMMA(16)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IDM
C
CHARACTER*8 CAL
CHARACTER*185 REC
C
C...Zero character variable
C
CAL(IC)=''
C
C...Abstract character variable
C
DO10 I=4,20
   IF(REC(I:I).EQ.'''')GO TO 20
10 CONTINUE
20 CAL(IC)=REC(4:(I-1))
C
C...Exit
C
RETURN
END

230
SUBROUTINE CALWKS

** Subroutine CALWKS **
** Keyboard input of working week length **
** for each calendar; **
** Determination of working week length **
** for each task; **

** ....................................................................... **

COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400), TRES(400,10),
&TRESQ(400,10), TCAL(400), NTRES(400), SFCODE(400), TWK(400),
&CRES(400,10), TCOST(400,9), TWBS(400)
COMMON/CALS/CAL(5), ICWK(5)
COMMON/COUNTS/IT, ID, IL, IR, IU, IC, IDM

CHARACTER*1  TTYPE, CHR
CHARACTER*3  CTRES
CHARACTER*8  CAL, TCAL
CHARACTER*9  TNAME
CHARACTER*12 TWBS
CHARACTER*27 TABREC(12), BLANK

INTEGER  SFCODE, TWK, TRES

...Initialise display table records

DO 20 I=1,12
   TABREC(1)(1:1)=CHAR(186)
   TABREC(1)(27:27)=CHAR(186)
   DO 10 J=2,26
      TABREC(1)(J:J)=T  
10 CONTINUE
20 CONTINUE

DO 30 J=2,26
   TABREC(1)(J:J)=CHAR(205)
   TABREC(12)(J:J)=CHAR(205)
   BLANK(J:J)= ' 
30 CONTINUE

TABREC(1)(1:1)=CHAR(201)
TABREC(1)(6:22)='Working Week data'
TABREC(1)(27:27)=CHAR(187)
TABREC(3)(3:10)='CALENDAR'
TABREC(3)(14:25)='WORKING WEEK'
TABREC(4)(17:22)='(Days)'
TABREC(12)(1:1)=CHAR(200)
TABREC(12)(27:27)=CHAR(188)

DO 40 I=1,IC
   J=I+5
   TABREC(J)(3:10)=CAL(I)
40 CONTINUE
C
C...Draw display table (BRIGHT WHITE on BLUE)
C
NLTL=8
NCTL=27
NLTOP=NLTL-1
CALL TXTCLR(7,0)
CALL SCLEAR
CALL TXTCLR(15,1)
DO 50 I=1,12
   CALL TMOVETO((NLTL+I-1),NCTL)
   CALL OUTEXT(TABREC(I))
50 CONTINUE
C
C...Define option highlight delay
C
NDEL=30
C
C...Input calendar working week data from keyboard
C
60 DO 80 I=1,IC
C
C...Highlight entry slot in table (BROWN)
C
CALL TXTCLR(0,6)
CALL TMOVETO((NLTL+I+4),(NCTL+18))
CALL OUTEXT('•')
C
C...Display input line (BRIGHT WHITE)
C
CALL TXTCLR(15,0)
CALL TMOVETO(NLTOP,NCTL)
CALL OUTEXT(CAL(I)//': Week length [ ]')
CALL TMOVETO(NLTOP,(NCTL+23))
C
C...Read working week length from keyboard
C
70 CALL KBREAD(0,CHR)
ICHR=ICHAR(CHR)
IF(ICHR.LT.49.OR.ICHR.GT.57)THEN
   CALL SPCHAR(1)
   GO TO 70
ENDIF
C
C...Inappropriate character; try again
C
CALL SPCHAR(1)
GO TO 70
ENDIF
C
C...Appropriate character; update table and variab'e ICWK
C
CALL TXTCLR(7,0)
DO 75 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTL+23))
   CALL OUTEXT(CHR)
75 CONTINUE
ICWK(I)=ICHR-48
CALL TXTCLR(7,1)
CALL TMVETO((NLTL+I+4),(NCTL+18))
CALL OUTEXT(CHR)
80 CONTINUE
C
C...Display CONTINUATION message;
C...highlight 1st character of options
C
CALL TXTCLR(15,0)
CALL TMVETO(NLTOP,NCTL)
CALL OUTEXT(BLANK(2:26))
CALL TMVETO(NLTOP,NCTL)
CALL OUTEXT('All OK? YES NO')
CALL TXTCLR(4,0)
CALL TMVETO(NLTOP,(NCTL+9))
CALL OUTEXT('Y')
CALL TMVETO(NLTOP,(NCTL+14))
CALL OUTEXT('N')
CALL TXTCLR(0,0)
CALL TMVETO(25,70)
CALL OUTEXT('')
CALL TMVETO(25,70)
C
C...Read options from keyboard
C
90 CALL KBREAD(0,CHR)
CALL CHRCNV(CHR)
IF(CHR.EQ.'N')THEN
  CALL TXTCLR(0,7)
  DO 110 M=1,NDEL
    CALL TMVETO((NLTL+I+4),(NCTL+18))
    CALL OUTEXT('NO')
  CONTINUE
  CALL TXTCLR(0,1)
  DO 120 1=1,1C
    CALL TMVETO((NLTL+I+4),(NCTL+18))
    CALL OUTEXT(' ') 
  CONTINUE 
  GO TO 60  
ELSEIF(CHR.EQ.'Y')THEN
  ELSE
C
C...Option is UNSATISFACTORY; Erase table data and try again;
C
CALL TXTCLR(0,7)
DO 110 M=1,NDEL
  CALL TMVETO((NLTL+I+4),(NCTL+18))
  CALL OUTEXT('NO')
110 CONTINUE
CALL TXTCLR(0,1)
DO 120 1=1,1C
  CALL TMVETO((NLTL+I+4),(NCTL+18))
  CALL OUTEXT(' ') 
120 CONTINUE 
GO TO 60 
ELSEIF(CHR.EQ.'Y')THEN
  ELSE
C
C...Option is SATISFACTORY; Continue
C
CALL TXTCLR(0,7)
DO 130 M=1,NDEL
  CALL TMVETO(NLTOP,(NCTL+9))
  CALL OUTEXT('YES')
130 CONTINUE 
GO TO 140 
ELSE
CALL SPCHAR(1)
GO TO 90
ENDIF

CALL TXTCLR(0,0)
CALL TMOVTO(NLTOP,NCTL)
CALL OUTEXT(BLANK(2:26))
CALL TXTCLR(15,0)
CALL TMOVTO((NLTL+12),NCTL)
CALL OUTEXT('Commencing data initialisation')

DO 160 I=1,IT
   DO 150 J=1,IC
      IF(TCAL(I).EQ.CAL(J))THEN
         TWK(I)=ICWK(J)
         GO TO 160
      ENDIF
   150 CONTINUE
160 CONTINUE

RETURN
END
SUBROUTINE TCOSTS

C****************************************************
COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400), TRES(400,10),
&TRESQ(400, 10), TCAL(400), NTRES(400), SF CODE(400), TWK(400),
&TCTRES(400,10), TCOST(400, 9), TWBS(400)
COMMON/RES/RNAME(100), RTYPE(100), RCOST(100)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC, IDM
COMMON/COSTS/CCODE(9), CPAY(9), CARR(9), CARRQ(9)
C
CHARACTER*1 TTYPE, RTYPE, CPAY, CARR, CHR
CHARACTER*3 RNAME, CTRES, CCODE
CHARACTER*4 FMT, CHR4
CHARACTER*8 TCAL
CHARACTER*9 TNAME
CHARACTER*10 CRDEL
CHARACTER*12 TWBS
CHARACTER*54 TABREC(16),BLANK
C
INTEGER SFCODE,TWK,TRES,CARRQ
C...Initialise display table strings
C
DO 20 I=1,16
   TABREC(I)(1:1)=CHAR(186)
   TABREC(I)(54:54)=CHAR(186)
   DO 10 J=2,53
      TABREC(I)(J:J)=' '
   10 CONTINUE
20 CONTINUE
DO 30 J=2,53
   TABREC(I)(J:J)=CHAR(205)
   TABREC(16)(J:J)=CHAR(205)
   BLANK(J:J)=' '
30 CONTINUE
TABREC(1)(1:1)=CHAR(201)
TABREC(1)(18:36)="Cost Component Data'
TABREC(1)(54:54)="CHR(187)
TABREC(3)(3:37)="COMPONENT LOADING CODE PAYMENT'
TABREC(3)(40:52)="PAYMENT DELAY'
TABREC(5)(3:21)="LABOUR Normal'
TABREC(6)(3:21)="MATERIALS Front '
TABREC(7)(3:21)="PLANT Normal'
TABREC(8)(3:21)="PLANT Front '

APPENDIX C.13: Subroutine TCOSTS
TABREC(9)(3:21)="PLANT Back"
TABREC(10)(3:21)="SUBCONTRACT Spread"
TABREC(11)(3:21)="OVERHEADS Normal"
TABREC(12)(3:21)="OVERHEADS Front"
TABREC(14)(3:21)="TASK PRICE Spread"
TABREC(16)(1:1)=CHAR(200)
TABREC(16)(54:54)=CHAR(188)

C...Initialise task cost component arrays
C
DO 50 I=1,IT
  DO 40 J=1,9
    TCOST(I,J)=0.0
  40 CONTINUE
50 CONTINUE

C...Specify option highlight delay
C
  NDEL=30

C...Specify table position parameters
C
  NLTL=4
  NCTL=14
  NLTOP=NLTL-1
  NCTOP=NCTL
  NLBOT=NLTL+16
  NCBOT=NCTL

C...Draw display table (BRIGHT WHITE on BLUE)
C
  CALL TXTCLR(7,0)
  CALL SCLEAR
  CALL TXTCLR(15,1)
  DO 60 I=1,16
    CALL TMOTETO((NLTL+I-1),NCTL)
    CALL OUTEXT(TABREC(I))
  60 CONTINUE

C...Input cost component data from keyboard
C
  DO 240 I=1,9
  65 NLTAB=NLTL+I+3
  IF(I.GT.8)NLTAB=NLTAB+1

C...Display CODE input line;
C
  CALL TXTCLR(15,0)
  CALL TMOTETO(NLTAB,NCTOP)
  CALL OUTEXT('Code: [ ]')
  CALL OUTEXT('(Hit <CR> if component not applicable)')
C...Highlight table entry position;
C...Input component code from keyboard;
C
CALL TXTCLR(0,0)
CALL TMVETO(NLTOP,NCTOP)
CALL OUTEXT(BLANK(2:53))
CALL TXTCLR(7,1)
CALL TMVETO(NLTAB,(NCTL+24))
CALL OUTEXT(' - ')
CALL TMVETO(NLTAB,(NCTL+32))
CALL OUTEXT(' - ')
CALL TMVETO(NLTAB,(NCTL+43))
CALL OUTEXT(' - ')
CCODE(I)='000'
CPAY(I)='0'
CARR(I)=10
CARRQ(I)=0
GO TO 205
ENDIF
C
C...Component applicable; convert code to upper case if required;
C...Search for cost component index
C
CALL CHRCNV(CCODE(I))
DO 80 J=1,IR
   IF(RNAME(J).EQ.CCODE(I))THEN
      ICOST=J
      GO TO 90
   ENDIF
80 CONTINUE
C
C...Cost component index not found; display error message;
C...try again;
C
CALL TXTCLR(15,0)
CALL TMVETO(NLBOT,NCBOT)
CALL OUTEXT('Component code not found; Try again!!')
CALL SPCHAR(1)
CALL TMOVETO(NLTOP,(NCTOP+7))
CALL OUTEXT('   ')
GO TO 70

C

C...Cost component code found; Update table (WHITE on BLUE)

C

90 CALL TXTCLR(7,1)
CALL TMOVETO(NLTAB,(NCTL+24))
CALL OUTEXT(CCODE(I))

C

C...Hide cursor while calculating

C

CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTEXT('   ')
CALL TMOVETO(25,70)

C

C...Calculate appropriate cost component for each task;
C...Avoid calculation if a task has no resource demands;
C

CALL RESCOST(ICOST)
DO 110 J=1,IT
   IF(NTRES(J).GT.0)THEN
      DO 100 K=1,NTRES(J)
         L=TRES(J,K)
         TCOST(J,I)=TCOST(J,I)+RCOST(L)*TRESQ(J,K)
      100 CONTINUE
   ENDIF
110 CONTINUE

C

C...Avoid repetitive input for PLANT and OVERHEADS components
C

IF(I.NE.4.AND.I.NE.5.AND.I.NE.8)GO TO 115
IF(CCODE(I-1).EQ.'000')GO TO 115
CALL TMOVETO(NLTOP,NCTOP)
CALL OUTEXT(BLANK(2:53))
CPAY(I)=CPAY(I-1)
CARR(I)=CARR(I-1)
CARRQ(I)=CARRQ(I-1)
CALL TXTCLR(7,1)
IF(CPAY(I).EQ.'W')THEN
   CALL TMOVETO(NLTAB,(NCTL+30))
   CALL OUTEXT('WEEKLY ' )
ELSE
   CALL TMOVETO(NLTAB,(NCTL+30))
   CALL OUTEXT('MONTHLY')
ENDIF
IF(CARR(I).EQ.'O')THEN
   CALL TMOVETO(NLTAB,(NCTL+40))
   CALL OUTEXT('  - ')
   GO TO 205
ENDIF
GO TO 202
CALL TXTCLR(15,0)
CALL TMOVETO(NLTOP,NCTOP)
CALL OUTEXT(BLANK(2:53))
CALL OUTEXT('Payment: WEEKLY MONTHLY')
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+10))
CALL OUTEXT('W')
CALL TMOVETO(NLTOP,(NCTOP+18))
CALL OUTEXT('M')

CALL TXTCLR(0,6)
CALL TMOVETO(NLTAB,(NCTL+30))
CALL OUTEXT('  ')
CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTEXT('  ')
CALL TMOVETO(25,70)

CALL KBREAD(0,CPAY(I))
CALL CHRCNV(CPAY(I))
IF(CPAY(I).EQ.'W')THEN
DO 130 M=1,NDEL
CALL TMOVETO(NLTOP,(NCTOP+10))
CALL OUTEXT('WEEKLY')
CONTINUE
CALL TXTCLR(7,1)
CALL TMOVETO(NLTAB,(NCTL+30))
CALL OUTEXT('WEEKLY ')
ELSEIF(CPAY(I).EQ.'M')THEN
DO 140 M=1,NDEL
CALL TMOVETO(NLTOP,(NCTOP+18))
CALL OUTEXT('MONTHLY')
CONTINUE
CALL TXTCLR(7,1)
CALL TMOVETO(NLTAB,(NCTL+30))
CALL OUTEXT('MONTHLY')
ELSE
CALL SPCHAR(1)
GO TO 120
ENDIF

CALL TXTCLR(15,0)
CALL TMVETO(NLTOP,NCTOP)
CALL OUTEXT('Payment delay: [   ] ')
CALL OUTEXT('('Hit <CR> if no payment delay)')
CALL TMVETO(NLBOT,NCBOT)
CALL OUTEXT('[e.g. 28d ' //CHAR(240)//' 28 days; ')
CALL OUTEXT('6w ' //CHAR(240)//' 6 weeks; ')
CALL OUTEXT('lm ' //CHAR(240)//' 1 month;]')

CALL TXTCLR(0,6)
CALL TMVETO(NLTAB,(NCTL+40))
CALL OUTEXT('  ')

CALL CREAD(NLTOP,(NCTOP+16),CHR4,NCHR)
CALL TMVETO(NLTOP,NCTOP)
CALL OUTEXT(BLANK(2:53))
CALL TMVETO(NLBOT,NCBOT)
CALL OUTEXT(BLANK(2:53))
IF(NCHR.EQ.0)THEN

...Payment delay NOT APPLICABLE; Update table;
...Update CARR and CARRQ accordingly;
CALL TXTCLR(7,1)
CALL TMVETO(NLTAB,(NCTL+40))
CALL OUTEXT('  - ')
CARR(I)='0'
CARRQ(I)=0
GO TO 205
ENDIF

CALL CHRCNV(CHR4)
DO 160 J=1,NCHR
CHR=CHR4(J:J)
K=ICHAR(CHR)
IF(K.GT.47.AND.K.LT.58)GO TO 160
IF(CHR.EQ.'D'.OR.CHR.EQ.'W'.OR.CHR.EQ.'M')THEN
L = J - 1
GO TO 180
ENDIF
160 CONTINUE

C...Unsuitable input; display error message and try again;
C
CALL TMVETO(NLBOT, NCBOT)
CALL OUTEXT(BLANK(2:53))
CALL SPCHAR(1)
CALL TXTCLR(15, 0)
DO 170 M = 1, NDEL
   CALL TMVETO(NLBOT, NCBOT)
   CALL OUTEXT('Inappropriate input; try again!!')
170 CONTINUE
GO TO 150

C...Suitable input; abstract data
C
180 WRITE(FMT,190) L
190 FORMAT((I,11,))
READ(CHR4,FMT)CARRQ(I)
CARR(I)=CHR4(J:J)
C
C...Define table display string
C
CRDEL=''
WRITE(CRDEL(1:3),200)CARRQ(I)
200 FORMAT(13)
IF(CARR(I).EQ.'D')THEN
   CRDEL(5:7)='day'
   IF(CARRQ(I).GT.1)CRDEL(8:8)='s'
ELSEIF(CARR(I).EQ.'W')THEN
   CRDEL(5:8)='week'
   IF(CARRQ(I).GT.1)CRDEL(9:9)='s'
ELSE
   CRDEL(5:9)='month'
   IF(CARRQ(I).GT.1)CRDEL(10:10)='s'
ENDIF
C
C...Update table
C
202 CALL TXTCLR(7,1)
CALL TMVETO(NLTAB,(NCTL+40))
CALL OUTEXT(CRDEL)
C
C...Display CONTINUE? input line
C...Highlight first character of options
C
205 CALL TXTCLR(15,0)
CALL TMVETO(NLTOP,NCTOP)
CALL OUTEXT('Continue? YES NO')
CALL TXTCLR(4,0)
CALL TMVETO(NLTOP,(NCTOP+11))
CALL OUTEXT('Y')
CALL TMOVETO(NLTOP, (NCTOP+16))
CALL OUTEXT('N')

C...Hide cursor
C
CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTEXT(' ')
CALL TMOVETO(25,70)

C...Read option from keyboard; convert to upper case if required;
C
210    CALL KBREAD(0,CHR)
    CALL CHRCNV(CHR)
    IF(CHR.EQ.'Y')THEN
    C...Option is CONTINUE; Highlight option;
    C...Proceed to next component;
    C
    CALL TXTCLR(0,7)
    DO 220 M=1, NDEL
       CALL TMOVETO(NLTOP,(NCTOP+11))
       CALL OUTEXT('YES')
    220   CONTINUE
       GO TO 240
    ELSEIF(CHR.EQ.'N')THEN
    C...Option is DISCONTINUE; Highlight option;
    C
    CALL TXTCLR(0,7)
    DO 230 M=1, NDEL
       CALL TMOVETO(NLTOP,(NCTOP+16))
       CALL OUTEXT('NO')
    230  CONTINUE
    C...Erase table entries and input new entries
C
    CALL TXTCLR(7,1)
    CALL TMOVETO(NLTAB,(NCTL+24))
    CALL OUTEXT(BLANK(25:50))
    GO TO 65
    ELSE
    C...Inappropriate character; try again
C
    CALL SPCHAR(1)
    GO TO 210
    ENDF

240 CONTINUE

C...Erase any input or error lines; Display COMPLETION message
C
CALL TXTCLR(15,0)
CALL TMOVETO(NLTOP,NCTOP)
CALL OUTEXT(BLANK(2:53))
CALL TMOVETO(NLBOT, NCBOT)
CALL OUTEXT(BLANK(2:53))
CALL TMOVETO(NLBOT, NCBOT)
CALL OUTEXT('Completing data initialisation')

C
C...Exit
C
RETURN
END
Appendix C.14: Subroutine QUIT

SUBROUTINE QUIT(FLNAME, NCHR)
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c** **
c** Subroutine QUIT **
c** **
c** Concluding display for program PERTRA1 **
c** **
c** Writes data to data files for subsequent **
c** use by PERTRA2 and PERTRA3 **
c** **
c* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400), TRES(400,10),
&TRESQ(400,10), TCAL(400), NTRES(400), SFCODE(400), TWK(400),
&CTRES(400,10), TCOST(400,9), TWBS(400)
COMMON/LINKS/LFROM(500,2), LTO(500,2), LTYP(500,2),
&LDURN(500,2)
COMMON/PROJ/PHEA(3), PVER, PNAME, PSTART
COMMON/COSTS/CCODE(9), CPAY(9), CARR(9), CARRQ(9)
COMMON/COUNTS/IT, ID, IL, IR, IU, IC, IDM
C
CHARACTER*1 TTYPE, PVER, CPAY, CARR
CHARACTER*3 CTRES, CCODE
CHARACTER*8 TCAL, PNAME
CHARACTER*9 TNAME, PSTART
CHARACTER*12 FLNAME, TWBS
CHARACTER*30 PHEAD
CHARACTER*37 TABREC(20)
C
REAL LDURN
INTEGER SFCODE, TWK, TRES, CARRQ
C
C...Initialise display table strings
C
DO 20 I=1,20
  TABREC(I)(1:1)=CHAR(186)
  TABREC(I)(37:37)=CHAR(186)
DO 10 J=2,36
  TABREC(I)(J:J)=*'
10 CONTINUE
20 CONTINUE
TABREC(1)(1:1)=CHAR(201)
TABREC(1)(37:37)=CHAR(187)
TABREC(5)(1:1)=CHAR(204)
TABREC(5)(37:37)=CHAR(185)
TABREC(20)(1:1)=CHAR(200)
TABREC(20)(37:37)=CHAR(188)
DO 30 J=2,36
  TABREC(1)(J:J)=CHAR(205)
  TABREC(5)(J:J)=CHAR(205)
  TABREC(20)(J:J)=CHAR(205)
30 CONTINUE
TABREC(15)=TABREC(5)
TABREC(3)(10:28)="Quitting PERTRA1 !!"
TABREC(7)(5:33)="The following data files have"
TABREC(8)(5:17)="been created:"

C
C...Define table position
C
NLTL=3
NCTL=22

C...Clear screen; draw table
C
CALL TXTCLR(0,0)
CALL SCLEAR
CALL TXTCLR(15,1)
DO 40 I=1,20
   CALL TMOVETO((NLTL+I-1),NCTL)
   IF(I.LT.2.OR.I.GT.4)THEN
      CALL OUTEXT(TABREC(I))
   ELSE
      CALL OUTEXT(TABREC(I)(1:1))
      CALL TXTCLR(0,4)
      CALL OUTEXT(TABREC(I)(2:36))
      CALL TXTCLR(15,1)
      CALL OUTEXT(TABREC(I)(37:37))
   ENDIF
40 CONTINUE

C...Write data to datafiles
C
C...Define GENERAL file
C
FLNAME((NCHR+1):(NCHR+4))='.GEN'

C...Open file on Unit 4; Write GENERAL data
C
OPEN(UNIT=4,FILE=FLNAME)
WRITE(4,50)PNAME,PVER
50 FORMAT(A8,IX,A2)
WRITE(4,60)(PHEAD(I),I=1,3)
60 FORMAT(A30)
WRITE(4,70)PSTART
70 FORMAT(A9)
CLOSE(4)

C...Update table
C
CALL TXTCLR(15,1)
CALL TMOVETO((NLTL+9),(NCTL+5))
CALL OUTEXT('GENERAL data: ')
CALL TXTCLR(14,1)
CALL OUTEXT(FLNAME(1:(NCHR+4)))

245
C...Define TASKS data file
C
FLNAME((NCHR+1):(NCHR+4))='TSK'
C
C...Open file on Unit 4; Write TASKS data;
C
OPEN(UNIT=4, FILE=FLNAME)
WRITE(4,80)IT
80 FORMAT(14)
DUM=0.0
IDUM=1
WRITE(4,90)(TTYPE(I), TNAME(I), TWBS(I), TWK(I), SFCODE(I), IDUM, 
&DUM, TDURN(I), DUM, I=1, IT)
90 FORMAT(IX, A1, IX, A9, IX, A12, 312, 3F7.2)
CLOSE(4)

C...Update table
C
CALL TXTCLR(15,1)
CALL TMOVETO((NLTL+10),(NCTL+5))
CALL OUTEXT('TASK data: ')
CALL TXTCLR(14,1)
CALL OUTEXT(FLNAME(1:(NCHR+4)))

C...Define LINKS data file
C
FLNAME((NCHR+1):(NCHR+4))='LNK'
C
C...Open file on Unit 4; Write LINKS data
C
OPEN(UNIT=4, FILE=FLNAME)
WRITE(4,100)((LFROM(I,J), LTO(I, J), LTYPE(I, J), 
&LDURN(I,J), J=1,2), I=1, IL)
100 FORMAT(2(314,F7.2))
CLOSE(4)

C...Update table
C
CALL TXTCLR(15,1)
CALL TMOVETO((NLTL+11),(NCTL+5))
CALL OUTEXT('LINK data: ')
CALL TXTCLR(14,1)
CALL OUTEXT(FLNAME(1:(NCHR+4)))

C...Define COSTS data file
C
FLNAME((NCHR+1):(NCHR+4))='CST'
C
C...Open file on Unit 4; Write COSTS data
C
OPEN(UNIT=4, FILE=FLNAME)
WRITE(4,110)(CCODE(I), CPAY(I), CARR(I), CARRQ(I), I=1, 9)
110 FORMAT(A3, IX, A1, IX, A1, I4)
WRITE(4,120)((TCOST(I,J),J=1,9),I=1,IT)
120 FORMAT(9F10.2)
CLOSE(4)
C
C...Update table
C
CALL TXTCLR(15,1)
CALL TMOVETO((NLTL+12),(NCTL+5))
CALL OUTEXT('COST data: ') 
CALL TXTCLR(14,1)
CALL OUTEXT(FLNAME(1:(NCHR+4)))
C
C...Write FLNAME to file PERTRA.FLS on Unit 4
C
OPEN(UNIT=4,FILE='PERTRA.FLS')
WRITE(4,130)FLNAME(1:NCHR)
130 FORMAT(A8)
CLOSE(4)
C
C...Display concluding message
C
CALL TXTCLR(15,1)
CALL TMOVETO((NLTL+16),(NCTL+2))
CALL OUTEXT('Run PERTRA2 for interactive input')
CALL TMOVETO((NLTL+17),(NCTL+8))
CALL OUTEXT('of TASK DURATION data')
CALL TXTCLR(7,0)
CALL TMOVETO(23,22)
CALL OUTEXT('Program terminated')
C
C...Exit routine
C
RETURN
END
SUBROUTINE RESCOST(ICOST)
C******************************************************************************
C** Subroutine RESCOST **
C** Calculates individual cost components of resources **
C** ICOST - Index of cost component under consideration **
C******************************************************************************

COMMON/RES/RNAME(100), RTYPE(100), RCOST(100)
COMMON/SUBRES/CRTOP(110), CRSUB(110), RSUBQ(110), RTOP(110), &RSUB(110)
COMMON/COUNTS/IT, ID, IL, IR, IU, IC, IDM

CHARACTER*1 RTYPE
CHARACTER*3 RNAME, CRTOP, CRSUB

INTEGER RTOP, RSUB

C...Initialise cost array
DO 10 I=1,IR
   RCOST(I)=0.0
10 CONTINUE
RCOST(ICOST)=1.0

C...Initialise control statement LAB
ASSIGN 35 TO LAB

C...Initialise calculation indices
IND=ICOST
I1=1
I2=0

C...Rearrange SUB-RESOURCE arrays
20 ISTART=I1
DO 30 I=ISTART, IU
   IF(RSUB(I).EQ.IND)THEN
      CALL SRSWOP(1,11)
      I1=I1+1
   ENDIF
30 CONTINUE

C...Test (First time only) if cost component exists as a sub-resource
GO TO LAB
35 IF(I1.LT.2)THEN
C...It does not exist; Exit subroutine;
C
   GO TO 50
   ELSE
   C...It exists; Update control statement
   C...to avoid unnecessary testing
   C
      ASSIGN 37 TO LAB
      ENDIF
37 I2=I2+1
   IND=RTOP(I2)
   IF(I2.LT.11)GO TO 20
C...Calculate cost component for each resource
C
   DO 40 I=1,(I2-1)
      J=RTOP(I)
      K=RSUB(I)
      RCOST(J)=RCOST(J)+RCOST(K)*RSUBQ(I)
   40 CONTINUE
C...Exit
C
50 RETURN
END
SUBROUTINE SRSWOP(I,II)

C******************************************************************************
C** Subroutine SRSWOP **
C** Sub-resource swopping routine used by Subroutine RESCOST **
C******************************************************************************

COMMON/SUBRES/CRTOP(110),CRSUB(110),RSUBQ(110),RTOP(110),RSUB(110)

C
CHARACTER*3 CRTOP,CRSUB
C
INTEGER RTOP,RSUB,TEMPI,TEMP2
C
TEMPI=RTOP(I)
TEMP2=RSUB(I)
TEMP3=RSUBQ(I)
RTOP(I)=RTOP(I1)
RSUB(I)=RSUB(I1)
RSUBQ(I)=RSUBQ(I1)
RTOP(I1)=TEMPI
RSUB(I1)=TEMP2
RSUBQ(I1)=TEMP3
C
...Exit
C
RETURN
END
SUBROUTINE LSORT

C******************************************************************************
C** Subroutine LSORT **
C** **
C** Rearranges links into the appropriate order for **
C** the forward/backward pass through the network **
C** **
C******************************************************************************

COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400), TRES(400,10),
&TRESQ(400,10), TCAL(400), NTRES(400), SFCODE(400), TWK(400),
&CTRES(400,10), TCOST(400,9), TWBS(400)
COMMON/LINKS/LFROM(500,2), LTO(500,2), LTYPE(500,2),
&LDURN(500,2)

CHARACTER*1 TTYPE
CHARACTER*3 CTRES
CHARACTER*8 TCAL
CHARACTER*9 TNAME
CHARACTER*12 TWBS

REAL LDURN
INTEGER SFCODE, TWK, TRES

C...Rearrange links in FORWARD PASS order
C
C...Zero link counter
C
II=1
C
C...Search for START tasks
C
DO 10 I=1,IL
   J=LFROM(1,1)
   IF(SFCODE(J).EQ.0)THEN
      CALL LSWOP(I,II,1)
      II=II+1
   ENDIF
10 CONTINUE
C
C...Sort remaining links into order
C
DO 30 I2=1,IL
   ISTART=II
   IF(ISTART.GE.IL)GO TO 40
   DO 20 I=ISTART,IL
      IF(LFROM(I,1).EQ.LTO(I2,1))THEN
         CALL LSWOP(I,II,1)
         II=II+1
      ENDIF
20 CONTINUE

251
C CONTINUE
C Set up BACKWARD PASS array
C 40 DO 50 I=1,IL
      LFROM(I,2)=LFROM(I,1)
      LTO(I,2)=LTO(I,1)
      LTYPE(I,2)=LTYPE(I,1)
      LDURN(I,2)=LDURN(I,1)
50 CONTINUE
C Rearrange links in BACKWARD PASS order
C Zero link counter
C 11=1
C Search for FINISH tasks
C DO 60 I=1,IL
      J=LTO(I,2)
      IF(SFCODE(J).EQ.2)THEN
         CALL LSWOP(I,II,2)
         11=11+1
      ENDIF
60 CONTINUE
C Sort remaining links into order
C DO 80 12=1,IL
      ISTART=II
      IF(ISTART.GE.IL)GO TO 90
      DO 70 1=ISTART,IL
         IF(LTO(I,2).EQ.LFROM(12,2))THEN
            CALL LSWOP(I,11,2)
            11=11+1
         ENDIF
70 CONTINUE
80 CONTINUE
C Exit
C 90 RETURN
END
Appendix C.18: Subroutine LSWOP

SUBROUTINE LSWOP(I,II,J)
C----------------------------------------------------------------------
C** **
C** Subroutine LSWOP **
C** **
C** Link swopping routine used by Subroutine LSORT **
C** **
C----------------------------------------------------------------------
C
COMMON/LINKS/LFROM(500,2),LTO(500,2),LTYPE(500,2),
&LDURN(500,2)
COMMON/COUNTS/IT,ID,IL,IR,IU,IC,IDM
C
REAL LDURN
INTEGER TEMP1,TEMP2,TEMP3
C
TEMP1=LFROM(I,J)
TEMP2=LTO(I,J)
TEMP3=LTYPE(I,J)
TEMP4=LDURN(I,J)
LFROM(I,J)=LFROM(I1,J)
LTO(I,J)=LTO(I1,J)
LTYPE(I,J)=LTYPE(I1,J)
LDURN(I,J)=LDURN(I1,J)
LFROM(I1,J)=TEMP1
LTO(I1,J)=TEMP2
LTYPE(I1,J)=TEMP3
LDURN(I1,J)=TEMP4
C
C...Exit
C
RETURN
END
Appendix C.19: Subroutine CHRCNV

SUBROUTINE CHRCNV(CHRVAR)

C * *  **
C** Subroutine CHRCNV **
C** **
C** Converts any lower case characters in CHRVAR to upper case **
C** **
C***********************************************************************
C
CHARACTER CHRVAR(*)

NCHR=LEN(CHRVAR)
DO 10 I=1,NCHR
   N=ICHAR(CHRVAR(I:I))
   IF(N.GT.96.AND.N.LT.123)THEN
      N=N-32
      CHRVAR(I:I)=CHAR(N)
   ENDIF
10 CONTINUE
C
C...Exit
C
RETURN
END
SUBROUTINE CREAD(NL, NC, CHRVAR, NCHR)
C*******************************************************************
C** Subroutine CREAD **
C** Reads a character variable CHRVAR, of maximum length NCHR **
C** characters, from the keyboard at screen position NL,NC. **
C** Returns the actual number of characters NCHR in the **
C** variable. **
C** Uses 'C' subroutines. **
C*******************************************************************
C
CHARACTER CHRVAR*(*)
CHARACTER*1 CHR
C
CALL TXTCLR(7,0)
C
C...Move cursor to required position
C
CALL TMOVETO(NL,NC)
I=1
10 IF(I.GT.NCHR)GO TO 30
J=IC-NC+I-1
C
C...Read input character (no echo)
C
20 CALL KBREAD(0, CHR)
J=ICHAR(CHR)
IF(J.EQ.13)THEN
C
C...Character is CARRIAGE RETURN; Determine length of character variable; Exit routine;
C
N=IC-NC
GO TO 50
ELSEIF(J.EQ.8)THEN
C
C...Character is BACKSPACE; Delete preceding character; Protect against over-deletion
C
IC=IC-1
I=I-1
IF(I.GT.0)GO TO 25
I=I+1
IC=I+1
CALL SPCHAR(1)
GO TO 20
25 CALL TMOVETO(NL,IC)
CALL OUTEXT(' ')  
CALL TMOVETO(NL, IC)  
GO TO 20  
ELSEIF(J.EQ.32)THEN  
C  
C...Character is SPACE; Ignore  
C  
CALL SPCHAR(1)  
GO TO 20  
ELSE  
C  
C...Character is acceptable; Update  
C...character variable and continue  
C  
N=(IC-NC)+1  
CHRVAR(N:N)=CHR  
CALL OUTEXT(CHR)  
ENDIF  
I=I+1  
GO TO 10  
C  
C...Hide cursor  
C  
30 CALL TXTCLR(0,0)  
CALL TMOVETO(25,70)  
CALL OUTEXT(' ')  
CALL TMOVETO(25,70)  
40 CALL KBREAD(0,CHR)  
J=ICHAR(CHR)  
IF(J.EQ.13)THEN  
GO TO 50  
ELSEIF(J.EQ.8)THEN  
CALL TXTCLR(7,0)  
CALL TMOVETO(NL, IC)  
CALL OUTEXT(' ')  
CALL TMOVETO(NL, IC)  
I=NCHR  
GO TO 10  
ELSE  
CALL SPCHAR(1)  
GO TO 40  
ENDIF  
C  
C...Exit  
C  
50 NCHR=N  
RETURN  
END
APPENDIX D: Program PERTRA2 and Subroutines

Appendix D.1: Program PERTRA2

C ***************************************************************
C ** Program PERTRA2.FOR **
C ** Interactive input of TASK DURATIONS for use by **
C ** Risk Analysis Program PERTRA3.FOR **
C ** TDURN(1,1): OPTIMISTIC duration; **
C ** TDURN(1,2): MOST LIKELY duration; **
C ** TDURN(1,3): PESSIMISTIC duration; **
C ** Reads initial data from user-specified file: FLNAME.TSK **
C ** generated by initialisation program: PERTRA1.FOR **
C ** Appends FLNAME.TSK to include data input from keyboard **
C ***************************************************************

COMMON/TASKS/TNAME(400),TWBS(400),TTYPE(400),TDURN(400,3),
&TWK(400),SFCODE(400),FTASK(400),PCODE(400),DFACT(2)
COMMON/FILDAT/CODE(6),POS(6),OPR(6),SELECT(6),NCHR(6),
&LOCIC,IFLT,NTEST
C
CHARACTER*1 TTYPE,YES
CHARACTER*9 TNAME,SELECT
CHARACTER*12 TWBS,CVAR,FLNAME
C
REAL TDURN
INTEGER FTASK,PCODE,CODE,POS,OPR,NCHR,SFCODE,TWK,IFLT,NTEST
C
C...Read FLNAME from file: PERTRA.FLS on Unit 3
C
OPEN(UNIT=3,FILE='PERTRA.FLS')
READ(3,10)FLNAME
10 FORMAT(A8)
CLOSE(3)
C
C...Set TEXT mode; display program introduction
C
CALL SETMODE(0)
CALL HEADER(FLNAME)
C...Read TASK data from FLNAME on Unit 3

C

FLNAME(9:12)='TSK'
OPEN(UNIT=3,FILE=FLNAME)
READ(3,*), NTASK
READ(3,20)(TTYPE(I),TNAME(I),TWBS(I),TWK(I),SFCODE(I),
&PCODE(I),(TDURN(I,J),J=1,3),I=1,NTASK)
20 FORMAT(1X,A1,1X,A9,1X,A12,312,3F7.2)
CLOSE(3)

C...Set HAMMOCK/MILESTONE task codes set to zero

C

DO 30 I=1,NTASK
   IF(TTYPE(I).EQ.'H'.OR.TTYPE(I).EQ.'M') PCODE(I)=0
30 CONTINUE

C...Display screen enquiry

C

40 CALL QDISPLAY(JCODE,FLNAME,NTASK)

C...Quit program if so desired; Otherwise continue

C

IF(JCODE.GT.1)GO TO 999

C...Input filtration parameters from keyboard

C

50 CALL FDISPLAY(JCODE)

C...Filter tasks in accordance with filtration parameters

C

CALL FILTER(NTASK)

C...Display filtered tasks on screen

C

CALL TDISPLAY(ICODE,JCODE)

C...If filtration unsatisfactory, input new filtration parameters

C

IF(ICODE.GT.0)GO TO 50

C...Filtration satisfactory; Update the appropriate variables

C

IF(JCODE.EQ.0)THEN

C...INVARIABLE duration tasks; Update task codes PCODE

C

DO 60 I=1,IFLT
   J=FTASK(I)
   PCODE(J)=0
60 CONTINUE

ELSE
C...VARIABLE duration tasks; Update OPTIMISTIC/PESSIMISTIC durations
C
   DO 70 I=1,IFLT
       J=FTASK(I)
       TDURN(J,1)=TDURN(J,2)*DFACT(1)
       TDURN(J,3)=TDURN(J,2)*DFACT(2)
    70 CONTINUE
   ENDIF
C
C...Repeat for next set of tasks
C
   GO TO 40
C
C...Program ends
C

999 END
SUBROUTINE HEADER(FLNAME)

** Subroutine HEADER **
** Outputs program introductory message **
** to screen **

CHARACTER*1 CHR
CHARACTER*12 FLNAME
CHARACTER*61 TXTREC(45)
CHARACTER*65 TABREC(21)

C...Initialise TABLE records
C
DO 20 I=1,23
   TABREC(1)(1:1)=CHAR(186)
   TABREC(1)(65:65)=CHAR(186)
   DO 10 J=2,64
      TABREC(1)(J:J)=CHAR(205)
   10 CONTINUE
20 CONTINUE
   TABREC(1)(1:1)=CHAR(201)
   TABREC(1)(65:65)=CHAR(187)
   TABREC(5)(1:1)=CHAR(204)
   TABREC(5)(65:65)=CHAR(185)
   TABREC(21)(1:1)=CHAR(200)
   TABREC(21)(65:65)=CHAR(188)
   DO 30 J=2,64
      TABREC(1)(J:J)=CHAR(205)
      TABREC(5)(J:J)=CHAR(205)
      TABREC(21)(J:J)=CHAR(205)
30 CONTINUE
   TABREC(3)(24:38)="Program PERTRA2"

C...Initialise TEXT records
C
DO 40 I=1,35
   DO 35 J=1,61
      TXTREC(1)(J:J)=CHAR(205)
   35 CONTINUE
40 CONTINUE
   TXTREC(1)(1:43)="Interactive input of TASK DURATION data for"
   TXTREC(1)(45:59)="the Monte-Carlo"
   TXTREC(2)(1:42)="simulation carried out by Program PERTRA3."
   TXTREC(4)(1:44)="Basic task data is read from the appropriate"
   TXTREC(4)(46:54)=".TSK file"
   TXTREC(5)(1:29)="generated by Program PERTRA1."
   TXTREC(5)(31:57)="Following input of the task"
duration data, this file will be updated, thus providing the facility for performing intermediate checks on the data file. For convenience, PERTRA2 facilitates 'SELECTIVE display of' tasks on the basis of:

(a) Task NAME;
(b) Task WORK BREAKDOWN STRUCTURE;
(c) Task TYPE;

For the purpose of data input, tasks may be 'categorised as' follows:

Type 1: Those whose durations are VARIABLE:
   (a) OPTIMISTIC duration;
   (b) MOST LIKELY duration;
   (c) PESSIMISTIC duration;

Type 2: Those whose durations are INVARIABLE:
   subject to a probability distribution as above;
   Note 1: MILESTONE tasks will automatically be included within category 2;
   Note 2: By virtue of their nature, HAMMOCK tasks will also be automatically included within category 2;

For convenience, PERTRA2 facilitates SELECTIVE display of tasks on the basis of:

(a) Task NAME;
(b) Task WORK BREAKDOWN STRUCTURE;
(c) Task TYPE;

For the purpose of data input, tasks may be categorised as follows:

Type 1: Those whose durations are VARIABLE:
   (a) OPTIMISTIC duration;
   (b) MOST LIKELY duration;
   (c) PESSIMISTIC duration;

Type 2: Those whose durations are INVARIABLE:
   subject to a probability distribution as above;
   Note 1: MILESTONE tasks will automatically be included within category 2;
   Note 2: By virtue of their nature, HAMMOCK tasks will also be automatically included within category 2;
CALL SCLEAR
CALL SETWIN(1,1,25,80)
CALL TXTCLR(15,1)
DO 80 I=1,21
   CALL TMOVETO(1,8)
   IF(I.LT.2.OR.I.GT.4)THEN
      CALL OUTEXT(TABREC(I))
   ELSE
      CALL OUTEXT(TABREC(I)(1:1))
      CALL TXTCLR(0,6)
      CALL OUTEXT(TABREC(I)(2:64))
      CALL TXTCLR(15,1)
      CALL OUTEXT(TABREC(I)(65:65))
   ENDIF
80 CONTINUE

CALL SETWIN(6,10,20,71)

DO 90 I=1,45
   IF(MOD(I,15).GT.0)THEN
      CALL OUTEXT(TXTREC(I)//CHAR(10))
   ELSE
      CALL OUTEXT(TXTREC(I))
      CALL SETWIN(1,1,25,80)
      CALL TMOVETO(22,8)
      CALL TXTCLR(0,4)
      CALL OUTEXT(' Hit any key to continue ')
      CALL TXTCLR(0,0)
      CALL TMOVETO(25,75)
      CALL OUTEXT(' ')
      CALL TMOVETO(25,75)
      CALL KBREAD(0,CHR)
      IF(I.EQ.45)GO TO 100
      CALL TXTCLR(15,0)
      CALL TMOVETO(22,8)
      CALL OUTEXT(TABREC(6)(2:64))
      CALL TXTCLR(15,1)
      CALL SETWIN(6,10,20,71)
      CALL TMOVETO(15,62)
      CALL OUTEXT(CHAR(10))
   ENDIF
90 CONTINUE
...All text output; Display concluding message and exit routine

100 CALL TXTCLR(7,0)
   DO 110 I=1,8
       J=8-(I-1)
       IF(FLNAME(J:J).NE.' ')GO TO 120
   110 CONTINUE
   120 CALL TMOVETO(22,8)
       CALL OUTEXT('Reading Task Duration data from file: ')
       CALL TXTCLR(15,0)
       CALL OUTEXT(FLNAME(1:J)//'. TSK')
   RETURN
END
Appendix D.3: Subroutine QDISPLAY

SUBROUTINE QDISPLAY(JCODE,FLNAME,NTASK)

C******************************************************************************
C** Subroutine QDISPLAY **
C** Intermediate inquiry display **
C**
C******************************************************************************
C
COMMON/TASKS/TNAME(400),TWBS(400),TTYPE(400),TDURN(400,3),
&TWK(400),SFCODE(400),FTASK(400),PCODE(400),DFACT(2)
C
CHARACTER*1 CHR1,TTYPE
CHARACTER*9 TNAME
CHARACTER*12 FLNAME,TWBS
CHARACTER*40 TABREC(22)
C
INTEGER FTASK,PCODE,SFCODE,TWK
C
C...Specify highlight delay
C
NDEL=30
C
C...Initialise TABLE records
C
DO 20 I=1,9
    TABREC(I)(1:1)=CHAR(186)
    TABREC(I)(38:38)=CHAR(186)
    DO 10 J=2,37
        TABREC(I)(J : J)=1
    10 CONTINUE
20 CONTINUE
    TABREC(I)(1:1)=CHAR(201)
    TABREC(I)(38:38)=CHAR(187)
    TABREC(5)(1:1)=CHAR(204)
    TABREC(5)(38:38)=CHAR(185)
    TABREC(9)(1:1)=CHAR(200)
    TABREC(9)(38:38)=CHAR(188)
    DO 30 J=2,37
        TABREC(I)(J : J)=CHAR(205)
        TABREC(5)(J : J)=CHAR(205)
        TABREC(9)(J : J)=CHAR(205)
    30 CONTINUE
    TABREC(3)(11:28)='TASK DURATION DATA'

264
CALL SCLEAR
CALL SETWIN(1,1,25,80)
CALL TXTCLR(15,1)
NLTL=7
NCTL=22
DO 40 I=1,9
   CALL TMOVETO((NLTL+I-1),NCTL)
   IF(I.LT.2.OR.I.GT.4)THEN
      CALL OUTEXT(TABREC(I)(1:38))
   ELSE
      CALL OUTEXT(TABREC(I)(1:1))
      CALL TXTCLR(0,6)
      CALL OUTEXT(TABREC(I)(2:37))
      CALL TXTCLR(15,1)
      CALL OUTEXT(TABREC(I)(38:38))
   ENDIF
40 CONTINUE
45 CALL TXTCLR(15,1)
CALL TMOVETO((NLTL+6),(NCTL+2))
CALL OUTEXT('Task Duration: ')
CALL TXTCLR(7,1)
CALL OUTEXT('VARIABLE/INVARIABLE')

CALL TMOVETO((NLTL-1),NCTL+1)
CALL TXTCLR(15,0)
CALL TMOVETO(NLTOP,NCTOP)
CALL OUTEXT('Option: INPUT QUIT')
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+9))
CALL OUTEXT('I')
CALL TMOVETO(NLTOP,(NCTOP+16))
CALL OUTEXT('Q')
CALL TXTCLR(0,0)
CALL TMOVETO(25,75)
CALL OUTEXT('')
CALL TMOVETO(25,75)

CALL KBREAD(0,CHR1)
CALL CHRCNV(CHR1)
IF(CHR1.EQ.'I')THEN
   CALL TXTCLR(0,7)
   DO 60 M=1,NDEL
      CALL TMOVETO(NLTOP,(NCTOP+9))
      CALL OUTEXT('INPUT')
   60 END
ELSEIF(CHR1.EQ.'Q')THEN

C...Option is QUIT; Highlight; Update JCODE
C...Move to 'quitting' routine;
C
CALL TXTCLR(0,7)
DO 70 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+16))
   CALL OUTEXT('QUIT')
70 CONTINUE
JCODE=2
GO TO 140
ELSE

C...Inappropriate character; sound warning and try again;
C
CALL SPCHAR(1)
GO TO 50
ENDIF

C...Display TASK DURATION message with highlights
C
CALL TXTCLR(15,0)
CALL TMOVETO(NLTOP,NCTOP)
CALL OUTEXT('Task Duration: VARIABLE INVARIABLE')
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+16))
CALL OUTEXT('V')
CALL TMOVETO(NLTOP,(NCTOP+26))
CALL OUTEXT('I')
CALL TXTCLR(0,0)
CALL TMOVETO(25,75)
CALL OUTEXT(' V')
CALL TMOVETO(25,75)

C...Read input from keyboard; Convert to upper case if required;
C
80 CALL KBREAD(0,CHR1)
CALL CHRCNV(CHR1)
IF(CHR1.EQ.'V')THEN

C...Option is VARIABLE; Highlight; Update JCODE
C
CALL TXTCLR(0,7)
DO 90 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+16))
   CALL OUTEXT('VARIABLE')
90 CONTINUE
JCODE=1
CALL TXTCLR(0,4)  
CALL TMOVETO((NLTL+6),(NCTL+17))  
CALL OUTEXT('VARIABLE')  
ELSEIF(CHR1.EQ.'I')THEN

DO 100 M=1,NDEL  
   CALL TMOVETO(NLTOP,(NCTOP+26))  
   CALL OUTEXT('INVARIABLE')  
100 CONTINUE
   JCODE=0

CALL TXTCLR(0,4)  
CALL TMOVETO((NLTL+6),(NCTL+26))  
CALL OUTEXT('INVARIABLE')  
ELSE

CALL SPCHAR(1)  
GO TO 80
ENDIF

CALL TXTCLR(0,0)  
CALL TMOVETO(NLTOP,NCTOP)  
CALL OUTEXT(TABREC(2)(2:37))  
CALL TXTCLR(15,0)  
CALL TMOVETO(NLTOP,NCTOP)  
CALL OUTEXT('Continue? YES NO')  
CALL TXTCLR(4,0)  
CALL TMOVETO(NLTOP,(NCTOP+11))  
CALL OUTEXT('Y')  
CALL TMOVETO(NLTOP,(NCTOP+16))  
CALL OUTEXT('N')  
CALL TXTCLR(0,0)  
CALL TMOVETO(25,75)  
CALL OUTEXT(' ')  
CALL TMOVETO(25,75)

CALL KBREAD(0,CHR1)  
CALL CHRCNV(CHR1)  
IF(CHR1.EQ.'N')THEN
C...Option is DISCONTINUE; Highlight and start again
C
CALL TXTCLR(0,7)
DO 120 M=1, NDEL
   CALL TMOVETO(NLTOP, (NCTOP+16))
   CALL OUTEXT('NO')
120 CONTINUE
GO TO 45
ELSEIF(CHR1.EQ.'Y')THEN
C...Option is CONTINUE; Highlight and exit routine
C
CALL TXTCLR(0,7)
DO 130 M=1, NDEL
   CALL TMOVETO(NLTOP, (NCTOP+11))
   CALL OUTEXT('YES')
130 CONTINUE
CALL TXTCLR(7,0)
GO TO 999
ELSE
C...Inappropriate character; sound warning and try again;
C
CALL SPCHAR(1)
GO TO 110
ENDIF
C...Quitting routine
C
140 CALL TXTCLR(7,0)
CALL SCLEAR
C...Initialise display table records
C
DO 160 I=1,22
   TABREC(I)(1:1)=CHAR(186)
   TABREC(I)(40:40)=CHAR(186)
   DO 150 J=2,39
      TABREC(I)(J:J)=' '
150 CONTINUE
160 CONTINUE
   TABREC(1)(1:1)=CHAR(201)
   TABREC(1)(40:40)=CHAR(187)
   TABREC(5)(1:1)=CHAR(200)
   TABREC(5)(40:40)=CHAR(188)
   TABREC(6)(1:1)=CHAR(201)
   TABREC(6)(40:40)=CHAR(187)
   TABREC(12)(1:1)=CHAR(199)
   TABREC(12)(40:40)=CHAR(182)
   TABREC(17)(1:1)=CHAR(199)
   TABREC(17)(40:40)=CHAR(182)
   TABREC(22)(1:1)=CHAR(200)
   TABREC(22)(40:40)=CHAR(188)
   DO 170 J=2,39
   268
TABREC(1)(J:J)=CHAR(205)
TABREC(5)(J:J)=CHAR(205)
TABREC(6)(J:J)=CHAR(205)
TABREC(12)(J:J)=CHAR(196)
TABREC(17)(J:J)=CHAR(196)
TABREC(22)(J:J)=CHAR(205)

CONTINUE
TABREC(3)(11:30)='Quitting PERTRA2 !!!'
TABREC(8)(3:38)='You may now check your input data in'
DO 180 I=1,8
   J=8-(I-1)
   IF(FLNAME(J:J).NE. '  ')THEN
      GO TO 190
   ENDIF
180 CONTINUE
190 TABREC(10)(12:(J+21))='File: //FLNAME(1:J)\\.TSK'
TABREC(14)(4:36)='Task Duration Data may be amended'
TABREC(15)(4:36)='or updated by re-running PERTRA2;'
TABREC(19)(4:36)='The Monte-Carlo simulation may be'
TABREC(20)(4:34)='carried out by running PERTRA3;'

C
C...Draw 'quitting' display; write data to FLNAME on Unit 4
C
CALL TXTCLR(15,4)
NLTL=1
NCTL=21
DO 200 I=1,5
   CALL TMOVETO((NLTL+I-1),NCTL)
   IF(I.LT.2.OR.I.GT.4)THEN
      CALL OUTEXT(TABREC(I))
   ELSE
      CALL OUTEXT(TABREC(I)(1:1))
      CALL TXTCLR(0,4)
      CALL OUTEXT(TABREC(I)(2:39))
      CALL TXTCLR(15,4)
      CALL OUTEXT(TABREC(I)(40:40))
   ENDIF
200 CONTINUE
CALL TXTCLR(0,0)
CALL TMOVETO(24,75)
CALL OUTEXT('  ')
OPEN(UNIT=4, FILE=FLNAME)
WRITE(4,210)NTASK
210 FORMAT(14)
WRITE(4,220)(TTYPE(I), TNAME(I), TWBS(I), TWK(I), SFCODE(I),
&PCODE(I),(TDURN(I, J), J=1,3), I=1,NTASK)
CLOSE(4)
CALL TXTCLR(15,1)
DO 230 I=6,22
   CALL TMOVETO((NLTL+I),NCTL)
   CALL OUTEXT(TABREC(I))
230 CONTINUE
CALL TXTCLR(7,0)
CALL TMVETO((NLTL+23), NCTL)
CALL OUTEXT('Program terminated')

C
C...Exit routine
C
999 RETURN
   END
Appendix D.4: Subroutine FDISPLAY

SUBROUTINE FDISPLAY(JCODE)
C*****************************************************************************************
C** Subroutine FDISPLAY
C** Screen display for input of task filtration data
C*****************************************************************************************
COMMON/FILDAT/ CODE(6),POS(6),OPR(6),SELECT(6),
&NCHR(6),LOGIC,IFLT,NTEST
C
CHARACTER*1 CHR, INTCHR
CHARACTER*9 SELECT, BLANK
CHARACTER*38 TABREC(11), TOPREC(7)
C
INTEGER CODE, POS, OPR, NCHR, LOGIC, IFLT, NTEST
C
C...Specify display colours
C
C...Table borders and headings (BRIGHT WHITE)
C
ICLR1=15
C
C...Table background (BLUE)
C
ICLR2=1
C
C...Table text (WHITE)
C
ICLR3=7
C
C...Table location highlight (BROWN)
C
ICLR4=6
C
C...'Backspace' display (RED)
C
ICLR5=4
C
C...Specify 'reverse video' delay
C
NDEL=30
C
C...Initialise display strings
C
TOPREC(1)= 'FILTER ; CODE: NAME WBS TYPE QUIT'
TOPREC(2)= 'FILTER ; POSITION: [ ]'
TOPREC(3)= 'FILTER ; OPERAND: EQ NE'
TOPREC(4)= 'FILTER ; SELECTION: [ ]'
TOPREC(5)= 'CONTINUE ? YES NO'
TOPREC(6)= 'LOGIC: AND OR'
C
TOPREC(7)= 'GO CANCEL'
C
C...initialise table strings
C
DO 20 I=1,11
   TABREC(I)(1:1)=CHAR(186)
   TABREC(I)(38:38)=CHAR(186)
   DO 10 J=2,37
      TABREC(I)(J:J)=' '  
   10 CONTINUE
   20 CONTINUE
   TABREC(1)(1:1)=CHAR(201)
   TABREC(1)(38:38)=CHAR(187)
   TABREC(9)(1:1)=CHAR(199)
   TABREC(9)(38:38)=CHAR(182)
   TABREC(11)(1:1)=CHAR(200)
   TABREC(11)(38:38)=CHAR(188)
   DO 30 I=2,37
      TABREC(I)(1:1)=CHAR(205)
      TABREC(9)(I:I)=CHAR(196)
      TABREC(11)(I:I)=CHAR(205)
   30 CONTINUE
   TABREC(1)(17:22)= 'Filter'
   TABREC(2)(3:36)= 'FILTER CODE POS OPRN SELECTION'
   TABREC(10)(15:20)= 'LOGIC:'
   DO 60 I=1,6
      TABREC(I+2)(5:5)=INTCHR(I)
   60 CONTINUE
   BLANK=''
C
C...define text window
C
   CALL SETWIN(1,1,25,80)
C
C...define position of display
C
   NLTL=6
   NCTL=22
C
C...draw appropriate heading (BLACK on CYAN)
C
   CALL TXTCLR(7,0)
   CALL SCLEAR
   CALL TXTCLR(0,3)
   IF(JCODE.EQ.0)THEN
      CALL TMVETO(1,(NCTL+3))
      CALL OUTEXT(' Tasks with INVARIABLE durations ')
   ELSE
      CALL TMVETO(1,(NCTL+4))
      CALL OUTEXT(' Tasks with VARIABLE durations ')
   ENDIF
C...Draw filtration table

CALL TXTCLR(ICLR1,ICLR2)
DO 70 I=1,11
   CALL TMOVETO((NLTL+I-1),NCTL)
   CALL OUTEXT(TABREC(I))
70 CONTINUE

C...Write out input line (BRIGHT WHITE on BLACK)

80 NLTOP=NLTL-1
   NCTOP=NCTL
   DO 160 I=1,6
      NLTAB=NLTL+I+1
      DO 150 J=1,4
         TOPREC(J)(8:8)=INTCHR(I)
         CALL TMOVETO(NLTOP,NCTOP)
         CALL TXTCLR(15,0)
         CALL OUTEXT(TOPREC(J))
      IF(J.GT.1)GO TO 100
   DO 150 J=1,4
      TOPREC(J)(8:8)=INTCHR(I)
      CALL TMOVETO(NLTOP,NCTOP)
      CALL TXTCLR(15,0)
      CALL OUTEXT(TOPREC(J))
   IF(J.GT.1)GO TO 100

C...CODE display; Highlight first character of options (RED on BLACK)

NCTAB=NCTL+10
   NB=4
   CALL TXTCLR(4,0)
   CALL TMOVETO(NLTOP,(NCTOP+17))
   CALL OUTEXT('N')
   CALL TMOVETO(NLTOP,(NCTOP+23))
   CALL OUTEXT('W')
   CALL TMOVETO(NLTOP,(NCTOP+28))
   CALL OUTEXT('T')
   CALL TMOVETO(NLTOP,(NCTOP+34))
   CALL OUTEXT('Q')

C...Highlight table location

   CALL TMOVETO(NLTAB,NCTAB)
   CALL TXTCLR(0,ICLR4)
   CALL OUTEXT(BLANK(1:NB))

C...Hide cursor

   CALL TXTCLR(0,0)
   CALL TMOVETO(25,70)
   CALL OUTEXT(' ')
   CALL TMOVETO(25,70)

C...Read keyboard input; convert to upper case if required

90 CALL KBREAD(0,CHR)
   CALL CHRCNV(CHR)
   IF(CHR.EQ.'N')THEN
C Option is 'NAME'; Define CODE(I);
C Highlight option (BLACK on WHITE); Update table;
C
CODE(I)=1
CALL TXTCLR(0,7)
DO 92 M=1,NDEL
   CALL TMVETO(NLTOP,(NCTOP+17))
   CALL OUTEXT('NAME')
92 CONTINUE
CALL TMVETO(NLTAB,NCTAB)
CALL TXTCLR(ICLR3,ICLR2)
CALL OUTEXT('NAME')
ELSEIF(CHR.EQ.'W')THEN
C Option is 'WBS'; Define CODE(I);
C Highlight option (BLACK on WHITE); Update table;
C
CODE(I)=2
CALL TXTCLR(0,7)
DO 94 M=1,NDEL
   CALL TMVETO(NLTOP,(NCTOP+23))
   CALL OUTEXT('WBS')
94 CONTINUE
CALL TMVETO(NLTAB,NCTAB)
CALL TXTCLR(ICLR3,ICLR2)
CALL OUTEXT('WBS')
ELSEIF(CHR.EQ.'T')THEN
C Option is 'TYPE'; Default POSITION is 1
C Highlight option (BLACK on WHITE); Update table;
C
CODE(I)=3
POS(I)=1
CALL TXTCLR(0,7)
DO 96 M=1,NDEL
   CALL TMVETO(NLTOP,(NCTOP+28))
   CALL OUTEXT('TYPE')
96 CONTINUE
CALL TMVETO(NLTAB,NCTAB)
CALL TXTCLR(ICLR3,ICLR2)
CALL OUTEXT('TYPE 1')
ELSEIF(CHR.EQ.'Q')THEN
C Option is 'QUIT'; Highlight option (BLACK on WHITE);
C Move to 'quitting' routine
C
CALL TXTCLR(0,7)
DO 98 M=1,NDEL
   CALL TMVETO(NLTOP,(NCTOP+34))
   CALL OUTEXT('QUIT')
98 CONTINUE
CALL TMVETO(NLTAB,NCTAB)
CALL TXTCLR(0,ICLR2)
CALL OUTEXT(BLANK(1:4))
GO TO 170
ELSE
C...Inappropriate character; try again
C
CALL SPCHAR(1)
GO TO 90
ENDIF
GO TO 150
100 IF(J .GT. 2) GO TO 120
IF(CODE(I).EQ.3) GO TO 150
C...POSITION display;
C
NCTAB=NCTL+17
NB=1
C...Highlight table location
C
CALL TMOVETO(NLTAB,NCTAB)
CALL TXTCLR(0,1CLR4)
CALL OUTEXT(BLANK(1:NB))
C...Read data input from keyboard; Update table
C
CALL TMVETO(NLTOP,(NCTOP+22))
CALL KBREAD(0,CHR)
ICHR=ICHAR(CHR)
IF(ICHR.LT.49. OR .ICHG.T.57)THEN
CALL SPCHAR(1)
GO TO 110
ELSE
CALL OUTEXT(CHR)
ENDIF
POS(I)=ICHR-48
CALL TMOVETO(NLTAB,NCTAB)
CALL TXTCLR(1CLR3,1CLR2)
CALL OUTEXT(CHR)
GO TO 150
120 IF(J .GT. 3) GO TO 140
C...OPERAND display; Highlight first character
C...of options (RED on BLACK)
C
NCTAB=NCTL+22
NB=2
CALL TXTCLR(4,0)
CALL TMVETO(NLTOP,(NCTOP+20))
CALL OUTEXT('E')
CALL TMVETO(NLTOP,(NCTOP+24))
CALL OUTEXT('N')
CALL TMOVETO(NLTAB,NCTAB)
CALL TXTCLR(0,1CLR4)
CALL OUTEXT(BLANK(1:NB))

CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTEXT(' ')
CALL TMOVETO(25,70)

CALL KBREAD(0,CHR)
CALL CHRCNV(CHR)
IF(CHR.EQ.'E')THEN

OPR(I)=1
CALL TXTCLR(0,7)
DO 132 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+20))
   CALL OUTEXT('EQ')
   CONTINUE
 CALL TMOVETO(NLTAB,NCTAB)
 CALL TXTCLR(ICLR3,ICLR2)
 CALL OUTEXT('EQ')
ELSEIF(CHR.EQ.'N')THEN

C...Option is 'NOT EQUAL TO'; Define OPR(I);
C...Highlight option (BLACK on WHITE); Update table;
C
OPR(I)=2
CALL TXTCLR(0,7)
DO 134 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+24))
   CALL OUTEXT('NE')
   CONTINUE
 CALL TMOVETO(NLTAB,NCTAB)
 CALL TXTCLR(ICLR3,ICLR2)
 CALL OUTEXT('NE')
ELSE
C...Inappropriate character; try again
C
CALL SPCHAR(1)
GO TO 130
ENDIF
GO TO 150
C
C...SELECTION display;
C
C...Highlight table location
C
140 NCTAB=NCTL+27
       NB=9
       CALL TMOVETO(NLTAB,NCTAB)
       CALL TXTCLR(0,ICLR4)
       CALL OUTEXT(BLANK(1:NB))

C...Read data input from keyboard; update table
C
       CALL TMOVETO((NLTL+11),NCTL)
       CALL TXTCLR(0,ICLR5)
       CALL OUTEXT(' Use BACKSPACE key to erase  ')
       NCHR(I)=9
       CALL CREAD(NLTOP,(NCTOP+23),SELECT(I),NCHR(I))
       CALL CHRCNV(SELECT(I))
       CALL TMOVETO(1CLR3,ICLR2)
       NB=NB-NCHR(I)
       IF(NB.GT.0)THEN
           CALL OUTEXT(SELECT(I)(1:NCHR(I))/BLANK(1:NB))
       ELSE
           CALL OUTEXT(SELECT(I)(1:NCHR(I)))
       ENDIF
       CALL TMOVETO((NLTL+11),NCTL)
       CALL TXTCLR(7,0)
       CALL OUTEXT('  ')
       CALL TMOVETO((NLTL+11),NCTL)
       CALL TXTCLR(15,0)
       CALL OUTEXT(TOPREC(5))
       CALL TXTCLR(4,0)
       CALL TMOVETO(NLTOP,(NCTOP+12))
       CALL OUTEXT('Y ')
       CALL TMOVETO(NLTOP,(NCTOP+17))
       CALL OUTEXT('N')

C
C...'Quitting' routine; write CONTINUE display (WHITE on BLACK);
C...Highlight first character of options (RED on BLACK)
C
170 NTEST=I-1
       IF(NTEST.LT.2)THEN
           LOGIC=1
           GO TO 200
       ENDIF
       CALL TMOVETO(NLTOP,NCTOP)
       CALL TXTCLR(15,0)
       CALL OUTEXT(TOPREC(5))
       CALL TXTCLR(4,0)
       CALL TMOVETO(NLTOP,(NCTOP+12))
       CALL OUTEXT('Y ')
       CALL TMOVETO(NLTOP,(NCTOP+17))
       CALL OUTEXT('N')

277
CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTEXT( ' ' )
CALL TMOVETO(25,70)

CALL KBREAD(0,CHR)
CALL CHRCNV(CHR)
IF(CHR.EQ.'N')THEN
   CALL TXTCLR(0,7)
   DO 182 M=1,NDEL
      CALL TMOVETO(NLTOP,(NCTOP+17))
      CALL OUTEXT('NO')
   CONTINUE
   CALL TXTCLR(ICLR3,ICLR2)
   DO 183 M=1,NTEST
      CALL TMOVETO((NLTL+M+1),(NCTL+8))
      CALL OUTEXT(BLANK//BLANK//BLANK)
   CONTINUE
   GO TO 80
ELSEIF(CHR.EQ.'Y')THEN
   CALL TXTCLR(0,7)
   DO 184 M=1,NDEL
      CALL TMOVETO(NLTOP,(NCTOP+12))
      CALL OUTEXT('YES')
   CONTINUE
   ELSE
C...Inappropriate character; try again
   CALL SPCHAR(1)
   GO TO 180
ENDIF

CALL TXTCLR(15,0)
CALL OUTEXT(TOPREC(6))
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+8))

C...Write LOGIC display (WHITE on BLACK);
C...Highlight first character of options (RED on BLACK)

NLTAB=NLTL+9
NCTAB=NCTL+21
NB=3
CALL TMOVETO(NLTOP,NCTOP)
CALL TXTCLR(15,0)
CALL OUTEXT(TOPREC(6))
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+8))
CALL OUTTEXT('A')
CALL TMOVETO(NLTOP,(NCTOP+13))
CALL OUTTEXT('O')

C
C...Highlight table location
C
CALL TMOVETO(NLTAB,NCTAB)
CALL TXTCLR(0,ICLR4)
CALL OUTTEXT(BLANK(1:NB))

C
C...Hide cursor
C
CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTTEXT(' ')  
CALL TMOVETO(25,70)

C
C...Read keyboard input; convert to upper case if required
C
190 CALL KBREAD(0,CHR)
CALL CHRCNV(CHR)
IF(CHR.EQ.'A')THEN
C
C...Option is 'AND'; Define LOGIC(I);
C
LOGIC=1
CALL TXTCLR(0,7)
DO 192 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+8))
   CALL OUTTEXT('AND')
192 CONTINUE
CALL TMOVETO(NLTAB,NCTAB)
CALL TXTCLR(1CLR3,ICLR2)
CALL OUTTEXT('AND')
ELSEIF(CHR.EQ.'O')THEN
C
C...Option is 'OR'; Define LOGIC(I);
C
LOGIC=2
CALL TXTCLR(0,7)
DO 194 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+13))
   CALL OUTTEXT('OR')
194 CONTINUE
CALL TMOVETO(NLTAB,NCTAB)
CALL TXTCLR(1CLR3,ICLR2)
CALL OUTTEXT('OR ')
ELSE

279
CALL SPCHAR(1)
GO TO 190
ENDIF

...Write GO/CANCEL display (WHITE on BLACK)
...Highlight first character of options (RED on BLACK)

CALL TMOVETO(NLTOP, NCTOP)
CALL TXTCLR(15, 0)
CALL OUTEXT(TOPREC(7))
CALL TXTCLR(4, 0)
CALL TMOVETO(NLTOP, NCTOP)
CALL OUTEXT('G')
CALL TMOVETO(NLTOP, (NCTOP+4))
CALL OUTEXT('C')

...Hide cursor

CALL TXTCLR(0, 0)
CALL TMOVETO(25, 70)
CALL OUTEXT('  ')
CALL TMOVETO(25, 70)

...Read keyboard input; convert to upper case if required

CALL KBREAD(0, CHR)
CALL CHRCNV(CHR)
IF(CHR.EQ.'C')THEN
...
CALL TXTCLR(0,7)
DO 212 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+4))
   CALL OUTEXT('CANCEL')
CONTINUE
212
CALL TXTCLR(ICLR3,ICLR2)
DO 213 M=1,NTEST
   CALL TMOVETO((NLTL+M+1),(NCTL+8))
   CALL OUTEXT(BLANK//BLANK//BLANK)
213
CONTINUE
CALL TMOVETO(NLTAB,NCTAB)
CALL OUTEXT(BLANK)
GO TO 80
ELSEIF(CHR.EQ.'G')THEN
CALL TXTCLR(0,7)
DO 214 M=1,NDEL
   CALL TMOVETO(NLTOP,NCTOP)
   CALL OUTEXT('GO')
214 CONTINUE
ELSE
   CALL SPCHAR(1)
   GO TO 210
ENDIF

CALL TXTCLR(7,0)
RETURN
END
Appendix D.5: Subroutine FILTER

SUBROUTINE FILTER(NTASK)

C***********************************************************************
C *** Subroutine FILTER ***
C*** Filters NTASK tasks for screen display and data input; ***
C*** Performs a maximum of 6 (NTEST) levels of filtration ***
C*** according to specified characteristics (SELECT) of: ***
C***
C*** Task Name (CODE = 1) ***
C*** Task WBS (CODE = 2) ***
C*** Task Type (CODE = 3) ***
C***
C*** Filtration parameters are: LOGIC: 1 for .AND. ***
C*** 2 for .OR. ***
C***
C*** OPR: 1 for .EQ. ***
C*** 2 for .NE. ***
C**********************************************************************

COMMON/TASKS/TNAME(400),TWBS(400),TTYPE(400),TDURN(400,3),
&TWK(400),SFCODE(400),FTASK(400),PCODE(400),DFACT(2)

COMMON/FILDAT/CODE(6),POS(6),OPR(6),SELECT(6),NCHR(6),
&LOGIC,IFLT,NTEST

CHARACTER*1  TTYPE
CHARACTER*9  TNAME,SELECT
CHARACTER*12  TWBS,CVAR,FLNAME

REAL  TDURN
INTEGER FTASK,PCODE,CODE,POS,OPR,NCHR,SFCODE,
&TWK,IFLT,NTEST

C...Set counter for filtered task array FTASK
C
IFLT=0

C...Carry out filtration
C
DO 90 1=1,NTASK
   DO 70 J=1,NTEST
C
C...Define LH operator in equality/non-equality tests
C
N1=POS(J)
N2=N1+(NCHR(J)-1)
GO TO (10,20,30) CODE(J)
10 CVAR=TNAME(I)
GO TO (50,60) OPR(J)

282
20 CVAR=TWBS(I)
    GO TO (50,60) OPR(J)
30 CVAR=TTYPE(I)
    GO TO (50,60) OPR(J)

C...Perform appropriate test according to value of OPR
C...Post-test path determined by value of LOGIC
C
C...EQUALITY test
C
50 IF(CVAR(N1:N2).EQ.SELECT(J)(1:NCHR(J))) GO TO (70,80) LOGIC
    GO TO (90,70) LOGIC
C
C...NON-EQUALITY test
C
60 IF(CVAR(N1:N2).NE.SELECT(J)(1:NCHR(J))) GO TO (70,80) LOGIC
    GO TO (90,70) LOGIC
70 CONTINUE
    GO TO (80,90) LOGIC
C
C...Task satisfies filtration criteria; Update filtered task array
C
80 IFLT=IFLT+1
    FTASK(IFLT)=I
90 CONTINUE
C
C...Exit
C
RETURN
END
Appendix D.6: Subroutine TDISPLAY

SUBROUTINE TDISPLAY(ICODE,JCODE)

C **********************************************************
C *** Subroutine TDISPLAY ***
C ***
C*** Writes filtered task data to screen in sets of 15 ***
C***
C** ICODE = 0: Satisfactory display of tasks; **
C** ICODE = 1: Unsatisfactory display of tasks; **
C**
C** JCODE = 0: Display of INVARIABLE duration tasks; **
C** JCODE = 1: Display of VARIABLE duration tasks; **
C**
C****************************************************************

COMMON/TASKS/TNAME(400),TWBS(400),TTYPE(400),TDURN(400,3),
&TWK(400),SFCODE(400),FTASK(400),PCODE(400),DFACT(2)

COMMON/FIELDAT/CODE(6),POS(6),OPR(6),SELECT(6),NCHR(6),
&LOGIC,IFLT,NTEST

CHARACTER*1 TTYPE,CHR1
CHARACTER*8 CFACT
CHARACTER*9 TNAME,SELECT,FMT
CHARACTER*12 TWBS
CHARACTER*37 TCHAR
CHARACTER*43 TABREC(21),DBLANK,HEAD

REAL TDURN
INTEGER FTASK,PCODE,SFCODE,TWK,IFLT,CODE,POS,OPR,NCHR,
&LOGIC,NTEST

C...
C...Specify display colours
C
C...Heading (CYAN)
C
ICLR1=3

C...Table borders and headings (BRIGHT WHITE)
C
ICLR2=15

C...Table background (BLUE)
C
ICLR3=1

C...Table text (WHITE)
C
ICLR4=7
C...Specify 'reverse video' delay
C
NDEL=30
C
C...Initialise table strings
C
IF(JCODE.EQ.0)THEN
   HEAD=' Tasks with INVARIABLE durations'  
ELSE
   HEAD=' Tasks with VARIABLE durations'  
ENDIF
DO 20 I=1,21
   TABREC(I)(1:1)=CHAR(186)
   TABREC(I)(43:43)=CHAR(186)
   DO 10 J=2,42
      TABREC(I)(J:J)=' '
   10 CONTINUE
20 CONTINUE
   TABREC(1)(1:1)=CHAR(201)
   TABREC(1)(43:43)=CHAR(201)
   DO 30 I=2,42
      TABREC(I)(1:1)=CHAR(205)
      TABREC(I)(43:43)=CHAR(205)
   30 CONTINUE
   TABREC(1)(15:28)="Filtered Tasks'
   TABREC(2)(3:41)="NAME TYPE WBS DURATION'
IF(JCODE.EQ.0)THEN
   TABREC(18)(1:1)=CHAR(200)
   TABREC(18)(43:43)=CHAR(188)
   DO 40 I=2,42
      TABREC(18)(I:1)=CHAR(205)
   40 CONTINUE
ELSE
   TABREC(18)(1:1)=CHAR(199)
   TABREC(18)(43:43)=CHAR(182)
   DO 50 I=2,42
      TABREC(18)(I:1)=CHAR(196)
   50 CONTINUE
ENDIF
   TABREC(19)(3:29)="OPTIMISTIC duration factor:'
   TABREC(20)(3:30)="PESSIMISTIC duration factor:'
   DBLANK=' '
C
C...Clear screen
C
   CALL SETWIN(1,1,25,80)
   CALL SCLEAR
C
C...Define table position
C
   NLTL=4
   NCTL=19
C...Define position of display strings
C
    NLTOP=NLTL-1
    NCTOP=NCTL
C
C...Draw header
C
    CALL TMOVETO(1,NCTL)
    CALL TXTCLR(0,ICLR1)
    CALL OUTFEXT(HEAD)
C
C...Draw task display table
C
    CALL TXTCLR(ICLR2,ICLR3)
    IREC=18
    IF(JCODE.GT.0) IREC=21
    DO 60 I=1,IREC
        NL=NLTL+I-1
        CALL TMOVETO(NL,NCTL)
        CALL OUTFEXT(TABREC(I))
    60 CONTINUE
C
C...Set text window for task scrolling
C
    NLWTL=NLTL+2
    NCWTL=NCTL+2
    NLWBR=NLWTL+14
    NCWBR=NCWTL+37
    CALL SETWIN(NLWTL,NCWTL,NLWBR,NCWBR)
C
C...Scroll filtered tasks in sets of 15
C
    CALL TXTCLR(ICLR4,ICLR3)
    DO 100 I=1,IFLT
        K=FTASK(I)
    100 CONTINUE
C
C...Convert task records to CHARACTER string
C
    DO 70 J=1,37
        TCHAR(J:J)=' '
    70 CONTINUE
    WRITE(TCHAR,80)TDURN(K,2)
70 FORMAT(31X,F6.2)
    TCHAR(1:9)=TNAME(K)
    TCHAR(13:13)=TTYPE(K)
    TCHAR(18:29)=TWBS(K)
    IF(I.EQ.IFLT)THEN
        CALL OUTFEXT(TCHAR)
        GO TO 100
    ELSEIF(MOD(I,15).GT.0)THEN
C...Continue until 15th record
C
CALL OUTEXT(TCHAR//CHAR(10))
GO TO 100
ENDIF
C
C...15th record; pause, hide cursor, and display CONTINUE message;
C...Highlight first character of options (RED on BLACK);
C
CALL OUTEXT(TCHAR)
CALL SETWIN(1,1,25,80)
CALL TMOVETO(NLTOP,NCTOP)
CALL TXTCLR(15,0)
CALL OUTEXT('Continue? YES NO')
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+11))
CALL OUTEXT('Y')
CALL TMOVETO(NLTOP,(NCTOP+16))
CALL OUTEXT('N')
CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTEXT(' ') 
CALL TMOVETO(25,70)
C
C...Read input from keyboard; Convert to upper case
C
90 CALL KBR EAD(0,CHR1)
CALL CHRCNV(CHR1)
IF(CHR1.EQ.'Y')THEN
C
C...Option is CONTINUE; reset window and continue scrolling tasks;
C
CALL TXTCLR(0,7)
DO 92 M=1,NDE L
   CALL TMOVETO(NL TOP,(NCTOP+11))
   CALL OUTEXT('YES')
   CONTINUE
92 CALL TXTCLR(7,0)
CALL TMOVETO(NL TOP,NCTOP)
CALL OUTEXT(DBLANK)
CALL SETWIN(NLWTL,NCWTL,NLWBR,NCWBR)
CALL TXTCLR(ICLR4,ICLR3)
CALL TMOVETO(15,38)
CALL OUTEXT(CHAR(10))
ELSEIF(CHR1.EQ.'N')THEN
C
C...Option is DISCONTINUE; Set ICODE and exit routine;
C
CALL TXTCLR(0,7)
DO 94 M=1,NDEL
   CALL TMOVETO(NLTOP,(NCTOP+16))
   CALL OUTEXT('NO')
94 CONTINUE
ICODE=1
287
GO TO 999
ELSE
C
C...Inappropriate character; try again;
C
CALL SPCHAR(1)
GO TO 90
ENDIF
100 CONTINUE
C
C...Display concluding message;
C...Highlight first character of options (RED on BLACK); Hide cursor;
C
CALL SETWIN(1,1,25,80)
CALL TMOVETO(NLTOP,NCTOP)
CALL TXTCLR(15,0)
CALL OUTTEXT('Filtration OK? YES NO')
CALL TXTCLR(4,0)
CALL TMOVETO(NLTOP,(NCTOP+16))
CALL OUTTEXT('Y')
CALL TMOVETO(NLTOP,(NCTOP+21))
CALL OUTTEXT('N')
CALL TXTCLR(0,0)
CALL TMOVETO(25,70)
CALL OUTTEXT(' ')  
CALL TMOVETO(25,70)
C
C...Read input from keyboard; Convert to upper case
C
110 CALL KREAD(0,CHR1)
CALL CHRCNV(CHR1)
IF(CHR1.EQ.'N')THEN
C
C...Option is NOT OK; Set ICODE and exit routine
C
CALL TXTCLR(0,7)
DO 112 M=1,NDEL  
   CALL TMOVETO(NLTOP,(NCTOP+21))
   CALL OUTTEXT('NO')
112 CONTINUE
ICODE=1
GO TO 999
ELSEIF(CHR1.EQ.'Y')THEN
C
C...Option is OK; Set ICODE;
C
CALL TXTCLR(0,7)
DO 114 M=1,NDEL  
   CALL TMOVETO(NLTOP,(NCTOP+16))
   CALL OUTTEXT('YES')
114 CONTINUE
ICODE=0
ELSE

288
C
C...Inappropriate character; try again;
C
CALL SPCHAR(1)
GO TO 110
ENDIF
C
C...Check if DURATION displays required; If not, exit routine;
C
IF(JCODE.LT.1)THEN
CALL TMOVEO(NLTOP,NCTOP)
CALL TXTCLR(0,0)
CALL OUTEXT(DBLANK)
CALL TXTCLR(0,ICLR1)
CALL TMOVEO((NLTL+18),(NCTL+4))
CALL OUTEXT(' Tasks will be coded accordingly ')
CALL TMOVEO((NLTL+19),(NCTL+4))
CALL TXTCLR(0,4)
CALL OUTEXT(' Hit any key to continue ')
CALL TXTCLR(0,0)
CALL TMOVEO(25,70)
CALL OUTEXT(' ')
CALL TMOVEO(25,70)
CALL KREAD(0,CHR1)
GO TO 999
ENDIF
C
C...Display DURATION strings and read data from keyboard;
C...Convert to REAL NUMBER if input as INTEGER;
C
120 DO 180 I=1,2
CALL TXTCLR(15,0)
CALL TMOVEO(NLTOP,NCTOP)
CALL OUTEXT('Factor ')
IF(I.EQ.1)THEN
CALL OUTEXT('(0 < f '//CHAR(243)//' 1): ') IPOS=21
ELSE
CALL OUTEXT('(1 '//CHAR(243)//' f < 1000): ') IPOS=24
ENDIF
CALL OUTEXT('['
CALL TXTCLR(0,6)
CALL TMOVEO((NLTL+17+1),(NCTL+31))
CALL OUTEXT('')
125 NCHAR=8
CALL CREAD(NLTOP,(NCTOP+IPOS),CFACT,NCHAR)
CALL TMOVEO((NLTL+21),NCTL)
CALL OUTEXT(DBLANK)
FMT=''
DO 140 K=1,NCHAR
   IF(CFACT(K:K).EQ.'.')THEN
      L=NCHAR-K
      WRITE(FMT,130)NCHAR,L
   ENDIF
130 FORMAT('(F',I1,'.',I1,'',I1,'')')
READ(CFACT,FMT)DFACT(I)
GO TO 160
ENDIF
140 CONTINUE
WRITE(FMT,150)NCHAR
150 FORMAT('(I',11,')')
READ(CFACT,FMT)IDFACT
DFACT(I)=REAL(IDFACT)
160 WRITE(CFACT,170)DFACT(I)
170 FORMAT(F8.4)
C
C...Check if factor within limits
5
IF(I.EQ.1.AND.DFACT(I).GT.0.0.AND.DFACT(I).LE.1.0)GO TO 17

175
IF(I.EQ.2.AND.DFACT(I).GT.1.0.AND.DFACT(I).LE.1000.0)GO TO 17
C
C...Factor out of range; display error message and re-enter
C
CALL TXTCLR(15,0)
CALL TMOVE((NLT.I+21),NCTOP)
CALL OUTEXT('Factor out of range; Try again!!')
CALL SPCHAR(1)
CALL TMOVE((NLTOP),(NCTOP+IPOS))
CALL OUTEXT('')
GO TO 125
C
C...Duration within limits; update table
C
175 CALL TMOVE((NLT.I+17+I),(NCT.I+31))
CALL TXTCLR(1CLR4,1CLR3)
CALL OUTEXT(CFACT(1:8))
180 CONTINUE
C
C...Check whether input is satisfactory
C
CALL TMOVE((NLTOP,NCTOP)
CALL TXTCLR(0,0)
CALL OUTEXT(DBLANK)
CALL TMOVE((NLTOP,NCTOP)
CALL TXTCLR(15,0)
CALL OUTEXT('Factors OK? YES NO')
CALL TXTCLR(4,0)
CALL TMOVE((NLTOP),(NCTOP+13))
CALL OUTEXT('Y')
CALL TMOVE((NLTOP),(NCTOP+18))
CALL OUTEXT('N')
CALL TXTCLR(0,0)
CALL TMOVE(25,70)
CALL OUTEXT('')
CALL TMOVE(25,70)
C...Read input from keyboard; Convert to upper case
C
190 CALL KBREAD(0,CHR1)
   CALL CHRCONV(CHR1)
   IF(CHR1.EQ.'N')THEN
C...Option is NOT OK; Input new durations
C
   CALL TXTCLR(0,7)
   DO 200 M=1,NDEI
      CALL TMVETO(NLTOP,(NCTOP+18))
      CALL OUTEXT('NO')
   200 CONTINUE
   CALL TXTCLR(1CLR4,1CLR3)
   CFACI(1:8)=''
   DO 205 I=1,2
      CALL TMVETO((NLTL+17+1),(NCTL+31))
      CALL OUTEXT(CFACT(1:8))
   205 CONTINUE
   CALL TXTCLR(7,0)
   CALL TMVETO(NLTOP,NCTOP)
   CALL OUTEXT(DBLANK)
   GO TO 120
ELSEIF(CHR1.EQ.'Y')THEN
C...Option is OK; Exit routine
C
   CALL TXTCLR(0,7)
   DO 210 M=1,NDEI
      CALL TMVETO(NLTOP,(NCTOP+13))
      CALL OUTEXT('YES')
   210 CONTINUE
ELSE
C...Inappropriate character; try again;
C
   CALL SPCHAR(1)
   GO TO 190
ENDIF
C...Exit routine
C
999 CALL TXTCLR(7,0)
RETURN
END
Appendix D.7: Subroutine CHRCNV

SUBROUTINE CHRCNV(CHRVAR)
C**********************************************************************************************************************
C**                                                                 **
C** Subroutine CHRCNV **
C**                                                                 **
C** Converts any lower case characters in CHRVAR to upper case **
C**                                                                 **
C**********************************************************************************************************************
C
CHARACTER CHRVAR(*)

NCHR=LEN(CHRVAR)
DO 10 I=1,NCHR
   N=ICHAR(CHRVAR(I:1))
   IF(N.GT.96.AND.N.LT.123)THEN
      N=N-32
      CHRVAR(I:1)=CHAR(N)
   ENDIF
10 CONTINUE
C
C...Exit
C
RETURN
END
FUNCTION INTCHR(I)
C**********************************************************************
C**                      **
C**    Function INTCHR       **
C**                      **
C** Converts a single digit NUMERIC INTEGER (I) **
C**    into an ASCII CHARACTER        **
C**                      **
C**********************************************************************
C
CHARACTER*1 INTCHR
C
IF(I.LT.10)THEN
    N=I+48
    INTCHR=CHAR(N)
ENDIF
RETURN
END
Appendix D.9: Subroutine CREAD

SUBROUTINE CREAD(NL,NC,CHRVAR,NCHR)
C** Subroutine CREAD **
C** Reads a character variable CHRVAR, of maximum length **
C** NCHR characters, from the keyboard at screen **
C** position NL,NC. **
C** Returns the actual number of characters NCHR in the **
C** variable. **
C** Uses 'C' subroutines. **
C**
C***-----------------------------------------------***

CHARACTER CHRVAR*(*)
CHARACTER*1 CHR

CALL TXTCLR(7,0)
C...Move cursor to required position
CALL TMOVETO(NL,NC)
I=1
10 IF(I.GT.NCHR)GO TO 30
IC=NC+I-1

C...Read input character (no echo)
20 CALL KBREAD(0,CHR)
J=ICHAR(CHR)
IF(J.EQ.13)THEN
C...Character is CARRIAGE RETURN; 
C...Determine length of character variable; Exit routine;
N=IC-NC
GO TO 50
ELSEIF(J.EQ.8)THEN
C...Character is BACKSPACE; Delete preceding character;
C...Protect against over-deletion
IC=IC-1
I=I-1
IF(I.GT.0)GO TO 25
I=I+1
IC=I+1
CALL SPCHAR(1)
GO TO 20
25 CALL TMOVETO(NL,IC)
CALL OUTEXT(' ')  
CALL TMOVETO(NL,IC)  
GO TO 20  
ELSEIF(J.EQ.32)THEN  
C  
C...Character is SPACE; Ignore  
C  
CALL SPCHAR(1)  
GO TO 20  
ELSE  
C  
C...Character is acceptable; Update character variable and continue  
C  
N=(IC-NC)+1  
CHRVAR(N:N)=CHR  
CALL OUTEXT(CHR)  
ENDIF  
i=i+1  
GO TO 10  
C  
C...Hide cursor  
C  
30 CALL TXTCLR(0,0)  
CALL TMOVETO(25,70)  
CALL OUTEXT(' ')  
CALL TMOVETO(25,70)  
40 CALL KBREAD(0,CHR)  
J=ICHAR(CHR)  
IF(J.EQ.13)THEN  
GO TO 50  
ELSEIF(J.EQ.8)THEN  
CALL TXTCLR(7,0)  
CALL TMOVETO(NL,IC)  
CALL OUTEXT(' ')  
CALL TMOVETO(NL,IC)  
i=NCHR  
GO TO 10  
ELSE  
CALL SPCHAR(1)  
GO TO 40  
ENDIF  
C  
C...Exit  
C  
50 NCHR=N  
RETURN  
END
Appendix E: Sample of the Updated Task Data Written to the File
FLNAME.TSK

(Number of tasks in the project)

(1)  (2)  (3)  (4)  (5)  (6)  (7)  (8)  (9)

T CLEAR1  GN-CLEAR  5  1  1  1.00  2.00  4.00
T CLEAR2  GN-CLEAR  5  1  1  2.50  5.00 10.00
H CMPRESS  GN-CMPRESS  5  1  0  0.00 136.00  0.00
T CPC1CO  PR-CONC  5  1  1  5.00  1.00 20.00
T CPC1D1  PR-DELAY  7  1  0  0.00  1.00  0.00

Definition of Columns are:

(1): Task Type.
(2): Task Name.
(3): Task Work Breakdown Structure.
(4): Number of Working Days/Week.
(5): Task Code (Start, Finish or Normal Task).
(6): Variable/Invariable duration Task Codes (1= Invariable; 0= Variable).
(7): Optimistic Duration (Days).
(8): Most Likely Duration (Days).
(9): Pessimistic Durations (Days).

296
Appendix F: Program PERTRA3 and Subroutines

Appendix F.1: Program PERTRA3

C*************************************************************************
C                  Program PERTRA3.FOR
C*************************************************************************
C**                    Reads selected data from the files generated by the
C**                      programs PERTRA1.FOR; PERTRA2.FOR and perform
C**                      RISK ANALYSIS CALCULATIONS (Monte-Carlo simulation)
C*************************************************************************

C               COMMON/FREQ1/ PCFREQ(100),PREMPT(100),IPBAND(100),PFREQ(100),
&                RCPWPD(100)
C               COMMON/FREQ2/CFREQ(100),BANMPT(100),IBAND(100),FREQ(100),
&                ROPWRR(100)
C               COMMON/FREQ3/CCFREQ(100),CBMPT(100),ICBAND(100),CAFREQ(100),
&                ROPWCAP(100)
C               COMMON/PDIST/DLIM(400,2),FMODE(400)
C               COMMON/STATS/MINPDU,MXPDU,PRMEAN,PSKEW,MEDIAN,PROSIG
C               COMMON/STAT2/RMIN,RMAX,RRMEAN,RRSKEW,RRMEDR,RRSIG
C               COMMON/STAT3/CAPMIN,CAPMAX,CAMEDN,CBSKEDN,CAPSIG
C               COMMON/PDCOST/DCOSTS(500,9),COSPAY(500,9),COSDLY(500,9),
&               TIDDATE(500),MACDAY(50),TTCOS(500),ACTOTC(500),ACCTPR(500),
&               RCPTITM(1000),RATERN(1000)
C               COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400,3),SFCODE(400),
&               TWK(400),TCOST(400,9),TWBS(400),PCCODE(400),IDURN(400)
C               COMMON/LINKS/LFROM(500,2),LTO(500,2),LTYPE(500,2),LDURN(500,2)
C               COMMON/FINSTA/ISTART(400),IFINISH(400),NDAYOW(1000)
C               COMMON/HEADS/PHEAD(3)
C               COMMON/COSTS/CICODE(9),CPAY(9),CARR(9),CARRQ(9)
C               COMMON/CDATES/IDAYS(400,4),MODE(400,4),IDYDATE(400,4)
C               COMMON/DAYS/IESDNO(400),IESDOW(400),IDURN2(400),LFDN(400),
&               LFDW(400),IDURN3(400),ILDURN(500,2),ICRIT(400),ACRIT(400),
&               IPPRD(1000),IEFDNO(400),IEFDW(400),LSDNO(400),LSDW(400)
C               COMMON/STAFIN/NSTART,NFINISH,IPSDNO,IPFDNO,IPSDOW,NLINK,NTASK

C               CHARACTER*1 TTYPE,RCCODE,CPAY,CARR
C               CHARACTER*3 PSMOTH,CCODE,MODATE,PFMOTH
C               CHARACTER*8 PNAME
C               CHARACTER*9 TNAME
C               CHARACTER*12 FLNAME,TWBS
C               CHARACTER*30 PHEAD

C               REAL TDURN,LDURN,TCOST,MEDIAN,ACRIT
C               INTEGER SFCODE,TWK,PCCODE,PVER,CARRQ,LFROM,LTO,LTYPE,ICRIT,N
**Input data from terminal**

```
WRITE(6,10)
10 FORMAT('///','PROGRAM PERTRA3.FOR','/','9X','Reads selected data &from the files generated by Programs','/','9X','PERTRA1.FOR ; PERTRA2. &FOR and perform Risk Analysis calculations')
20 WRITE(6,30)
30 FORMAT('///','Select the type of random generation ','&','(R) for a Repetable OR (N) for a non-repetable ','\')
READ(5,40)RGCODE
40 FORMAT(A1)
```

**Check random generation code**

```
IF(RGCODE.EQ.'R'. OR. RGCODE. EQ.'r'. OR. RGCODE. EQ.'N'. OR. RGCODE. EQ.'n &' )G0T055
WRITE(6,50)RGCODE
50 FORMAT('///','CODE WERE WRITING AS ( ',A1,' ),/',' UNIDENTIFIED CODE &',/',' TRY AGAIN')
GOTO20
```

**Input number of runs**

```
55 WRITE(6,60)
60 FORMAT('///','Number of runs required ?? ','\')
READ(5,70)NRUNS
70 FORMAT(I4)
```

**Read FLNAME from file: FILES.PRT on unit 4**

```
OPEN(UNIT=4, FILE='PERTRA.FLS')
WRITE(6,80)
80 FORMAT('///','Reading Project Name ')
READ(4,90)FLNAME
```

**Define GENERAL data file**

```
FLNAME(9:12)='.GEN'
```

**Open GENERAL data file on unit 4 and read the data.**

```
OPEN(UNIT=4, FILE=FLNAME)
WRITE(6,100)FLNAME
100 FORMAT('///','Reading GENERAL data from file ',A12)
```

```
READ(4,110)PNAME,PVER
```

```
READ(4,120)(PHEAD(I),1=1,3)
```

```
READ(4,130)IPSDAY,PSMOTH,IPSYR
```

```
FORMAT(12,1X,A3,1X,I2)
CLOSE(4)
```
c...Adjust Project Start Year for day numbers calculations
c  IPSYR=IPSYR+1900
c...Define TASKS data file
c  FLNAME(9:12)='.TSK'
c...Open TASKS data file on unit 4 and read the data
c  OPEN(UNIT=4,FILE=FLNAME)  
  WRITE(6,140)FLNAME
  140 FORMAT///,' Reading TASKA data from file ','A12)  
  READ(4,150)(TTYPE(I),TNAME(I),TWBS(I),TWK(I),SFCODE(I),PCCODE(I),  
  &TDURN(I,J),J=1,3),I=1,NTASK)
  150 FORMAT(IX,A1,IX,A9, IX,A12,312,3F7.2)  
  CLOSE(4)
c...Define LINKS data file
c  FLNAME(9:12)='.LNK'
c...Open LINKS file on unit 4 and read the data.
c  OPEN(UNIT=4,FILE=FLNAME)  
  WRITE(6,160)FLNAME
  160 FORMAT///,' Reading LINKS data from file ','A12)
  READ(4,170)((LFROM(I,J),LTO(I,J),LTYPE(I,J),LDURN(I,J),J=1,2),I=1,  
  NLINK)
  170 FORMAT(2(3I4,F7.2))
  CLOSE(4)
c...Define COSTS data file
c  FLNAME(9:12)='.CST'
c...Open COSTS file on unit 4 and read data.
c  OPEN(UNIT=4,FILE=FLNAME)  
  WRITE(6,180)FLNAME
  180 FORMAT///,' Reading COSTS data from file ','A12)  
  READ(4,190)(CCODE(I),CPAY(I),CARR(I),CARRQ(I),I=1,9)
  READ(4,200)((TCOST(I,J),J=1,9),I=1,NTASK)
  190 FORMAT(A3,IX,Al,IX,Al,14)
  200 FORMAT(9F10.2)  
  CLOSE(4)
c...Initialise Criticality Index
c  DO210,I=1,NTASK  
     ICRIT(I)=0  
  210 CONTINUE
...Set run without probability analysis to run number one (Using Most Likely Task Duration)
N=1
...Change tasks duration to Integers
DO220, I=1,NTASK
   IDURN(I)=INT(TDURN(I,2))
   IF(TDURN(I,2).GT.INT(TDURN(I,2)))IDURN(I)=IDURN(I)+1
220 CONTINUE

...Calculate project start day number
CALL CALEND(1,IPSDAY,PSMOTH,IPSYR,IPSDNO)
...Calculate Project Start Day of the Week Number
IPSDOW-IDOTW(IPSDNO)
...Check that the starting date of the project is not a weekend and adjust if so ........
IF(IPSDOW.EQ.7) THEN
   IPSDNO=IPSDNO+1
   IPSDOW=1
ELSEIF(IPSDOW.EQ.6) THEN
   IPSDNO=IPSDNO+2
   IPSDOW=1
ENDIF

...Perform network Time analysis
WRITE(6,230)
   230 FORMAT(/// ' Performing network TIME ANALYSIS ')
   CALL ANALYS(N)

...Change Project Finish Day Number to Calendar Date
CALL CALEND(2,IPFDAY,PFMOTH,IPFYR,IPFDNO)

...Distribute the costs on the project days, calculate IRR and CAPTIM............
WRITE(6,250)
   250 FORMAT(///, ' Performing network COST ANALYSIS')
   CALL PCOSTS(NSTART,NTASK,ICDAYS,N)
c...Quit the program if so desired; Otherwise continue
  OPEN(UNIT=4,FILE='PERTRA3.RES')
  WRITE(4,251)IPSDAY,PSMOTH,IPSYR
  WRITE(4,252)IPFDAY,PFMOTH,IPFYR
  WRITE(4,255)IPRDRN(N)
  WRITE(4,253)CAPTIM(N)
  WRITE(4,254)RATERN(N)
  IF(NRUNS.EQ.1)GOTO900
  WRITE(6,260)
  CALL TRIDIS(I)
  CONTINUE
  CALL GETTIM(IHR,IMIN,I SEC,I HUN)
  SEED=REAL(IHUN*2592+1296)
  ENDIF
  CALL GETTIM(IHR,IMIN,I SEC,I HUN)
  SEED=REAL(IHUN*2592+1296)
  ENDIF
  IF(PCCODE(1).EQ.0.OR.TTYPE(1).EQ.'H')GOTO290
  WRITE(6,280)N
  CALL GETTIM(IHR,IMIN,I SEC,I HUN)
  SEED=REAL(IHUN*2592+1296)
  ENDIF
  IF(PCCODE(1).EQ.0.OR.TTYPE(1).EQ.'H')GOTO290

  IF(NRUNS.EQ.1)GOTO900
  WRITE(6,260)
  CALL TRIDIS(I)
  CONTINUE
  CALL GETTIM(IHR,IMIN,I SEC,I HUN)
  SEED=REAL(IHUN*2592+1296)
  ENDIF
  IF(PCCODE(1).EQ.0.OR.TTYPE(1).EQ.'H')GOTO290

251 FORMAT(//,1X,'PROJECT START DATE: ',I2,'/',A3,'/',I4)
252 FORMAT(//,1X,'PROJECT FINISH DATE: ',I2,'/',A3,'/',I4)
253 FORMAT(//,1X,'Negative CAPTIM = ',F10.2,' .Day')
254 FORMAT(//,1X,'Internal Rate of Return (IRR) =',F10.2,' %',//)
255 FORMAT(//,1X,'Project Duration =',16,2X,'Days')
  WRITE(6,280)N
  CALL GETTIM(IHR,IMIN,I SEC,I HUN)
  SEED=REAL(IHUN*2592+1296)
  ENDIF
c
   ...Randomly select (INTEGER) Tasks durations
   CALL DSELECT(I,SEED)
   CONTINUE

   ...Perform network time analysis for the selected durations
   CALL ANALYS(N)

   ...Distribute the costs on the project days,
   ...calculate IRR and CATIM
   CALL PCOSTS(NSTART,NTASK,ICDAYS,N)
   CONTINUE

   ...Change criticality index into percentages
   DO410,1=1,NTASK
      ACRIT(I)=FLOAT(ICRIT(I))/FLOAT(NRUNS)*100.0
   CONTINUE

   ...Write Screen Message
   WRITE(6,500)
   500 FORMAT(/,lx,'Performing Statistical Analysis on the Results')

   ...Sort project durations into rank order
   CALL PSORT(NRUNS)

   ...Calculate minimum, maximum, mean, median project durations and skewnesses
   CALL PRSTAT(NRUNS)

   ...Divide the project durations into bands and calculate frequency
   CALL FREQUA(NRUNS,NBANDS)

   ...Sort IRR and CAPTIM into rank order
   CALL RRSORT(NRUNS)

   ...Calculate minimum, maximum, mean, median IRR and skewnesses
   CALL RRSTAT(NRUNS)

   ...Calculate number of bands and frequency for the sorted IRR
   CALL FRQIRR(NRUNS,INBAND)

   ...Calculate minimum, maximum, mean, median CAPTIM and skewnesses
   CALL CASTAT(NRUNS)

   302
c...Calculate number of bands and frequency for the sorted CAPTIM
c
CALL FRQCAP(NRUNS,ICNBAND)
c
Write the results into the file PERTRA3.RES
c
WRITE(6,600)
600 FORMAT(///,1X,'Writing Data to the File PERTRA3.RES',///)
OPEN(UNIT=3,FILE='PERTRA3.RES')
WRITE(3,700)NRUNS
700 FORMAT(///,' Number of runs =',14)
WRITE(3,710)
710 FORMAT(///,2X,'TASK',3X,'TASK',6X,'TASK',3X,'OPTIMISTIC',2X,'MOST L
&IKELY',2X,'PESSIMISTIC',2X,'CRITICALITY',1X,'NUMBER',6
.&X,'TYPE',4X,'DURATION',4X,'DURATION',5X,'DURATION',5X,'INDEX (%)
')
WRITE(3,720)(I,TNAME(I),TTYPE(I),TDURN(I,1),TDURN(I,2),TDURN(I,3),
&ACRIT(I),I=1,NTASK)
720 FORMAT(///,IX,'Number of runs =',14)
WRITE(3,725)
725 FORMAT(///,1X,'PROJECT DURATIONS')
WRITE(3,730)(MINPDU,MXPDU,PRMEAN,PRMEDN,PRSIG,PRSKEW
730 FORMAT(//,'Minimum project duration =',14,4X,'DAYS',/,'Maximum p
&ject duration =',14,4X,'DAYS',/,'Mean Project duration =',F7
.&2,1X,'DAYS',/,'Median project duration =',F7.2,1X,'DAYS',/,'St
&andard Deviation =',F7.2,1X,'DAYS',/,'Skewness =',F7.4,///)
WRITE(3,740)
740 FORMAT(///,5X,'MIDPOINT',5X,'FREQUENCY',3X,'CUM. FREQUENCY',8X,'f(x)
&','/')
WRITE(3,750)(PRBMPT(I),PFREQ(I),PCFREQ(I),RCPWPD(I),I=1,(NBANDS-1)
&)
750 FORMAT(5X,F7.2,5X,F7.2,8X,F7.2,10X,F7.2)
WRITE(3,760)
760 FORMAT(///,1X,'INTERNAL RATE OF RETURN (IRR)')
WRITE(3,770)(RRMIN,RRMAX,RRMEAN,RRMEDN,RRSIG,RRSKEW
770 FORMAT(//,'Minimum IRR =',F12.3,IX,'%',/,'Maximum IRR =',F12.3,1
&X,'%',/,'Mean IRR =',F12.3,1X,'%',/,'Median IRR =',F12.3,1X,
&'%',/,'IRR Standard Deviation =',F12.5,1X,'%',/,'Skewness =',F9
.&5,///)
WRITE(3,780)(BANMPT(I),FREQ(I),CFREQ(I),ROPWRR(I),I=1,(INBAND-1))
780 FORMAT(2X,F10.2,5X,F7.2,8X,F7.2,11X,F7.2)
WRITE(3,781)
781 FORMAT(///,1X,'NEGATIVE CAPTIM')
WRITE(3,782)(CABMPT(I),CAFREQ(I),CCFREQ(I),ROPWCAP(I),I=1,(ICNBAND
&-1))
782 FORMAT(///,' Minimum CAPTIM =',F15.2,1X,'.DAY ',/,' Maximum CAPTI
&M =',F15.2,1X,'.DAY ',/,' Mean CAPTIM =',F15.2,1X,'.DAY ',
&/',,' Median CAPTIM =',F15.2,1X,'.DAY ',/,' CAPTIM Standard Devi
&ation =',F15.2,1X,'.DAY ',/,' Skewness =',F9.5,///)
WRITE(3,784)
784 FORMAT(///,1X,'STOP END
900 STOP
SUBROUTINE CALEND(ICODE, IDAY, MONTH, IYEAR, IDAYNO)

C**************************************************************************
C** Subroutine CALEND **
C** ICODE=1; Changes Project Start Date into Day Number **
C** ICODE=2; Changes Project Finish Day Number into Calendar Date **
C**************************************************************************

CHARACTER*3 MONTH

PARAMETER(A=365.25,B=30.6001,C=122.1)

C...Check ICODE and perform Calculation Accordingly

IF(ICODE.EQ.1)THEN

C...Change the Month from Character Format to Integer

IF(MONTH.EQ.'JAN')THEN
  IMONTH=1
ELSEIF(MONTH.EQ.'FEB')THEN
  IMONTH=2
ELSEIF(MONTH.EQ.'MAR')THEN
  IMONTH=3
ELSEIF(MONTH.EQ.'APR')THEN
  IMONTH=4
ELSEIF(MONTH.EQ.'MAY')THEN
  IMONTH=5
ELSEIF(MONTH.EQ.'JUN')THEN
  IMONTH=6
ELSEIF(MONTH.EQ.'JUL')THEN
  IMONTH=7
ELSEIF(MONTH.EQ.'AUG')THEN
  IMONTH=8
ELSEIF(MONTH.EQ.'SEP')THEN
  IMONTH=9
ELSEIF(MONTH.EQ.'OCT')THEN
  IMONTH=10
ELSEIF(MONTH.EQ.'NOV')THEN
  IMONTH=11
ELSEIF(MONTH.EQ.'DEC')THEN
  IMONTH=12
ENDIF
...Calculate the Day Number

IF(IMONTH.GE.1.AND.IMONTH.LE.2)THEN
   IYEAR1=IYEAR-1
   IMONTH1=IMONTH+13
ELSE
   IYEAR1=IYEAR
   IMONTH1=IMONTH+1
ENDIF
IDAYNO=INT(A*IYEAR1)+INT(B*IMONTH1)+IDAY

...ICODE is G.T. One; Changing from Day Number to Date is Required
ELSE

...Calculate the approximate year and month
IYEAR2=INT((IDAYNO-C)/A)
IMTH2=INT((IDAYNO-INT(A*IYEAR2))/B)

...Calculate the exact day of the date
IDAY=IDAYNO-INT(A*IYEAR2)-INT(B*IMTH2)

...Calculate the exact month of the date
IF(IMTH2.LT.14)THEN
   IMONTH=IMTH2-1
ELSE
   IMONTH=IMTH2-13
ENDIF

...Calculate the exact year
IF(IMONTH.LE.2)THEN
   IYEAR=IYEAR2+1
ELSE
   IYEAR=IYEAR2
ENDIF
IF (IMONTH.EQ.1) THEN
   MONTH='JAN'
ELSEIF(IMONTH.EQ.2)THEN
   MONTH='FEB'
ELSEIF(IMONTH.EQ.3)THEN
   MONTH='MAR'
ELSEIF(IMONTH.EQ.4)THEN
   MONTH='APR'
ELSEIF(IMONTH.EQ.5)THEN
   MONTH='MAY'
ELSEIF(IMONTH.EQ.6)THEN
   MONTH='JUN'
ELSEIF(IMONTH.EQ.7)THEN
   MONTH='JUL'
ELSEIF(IMONTH.EQ.8)THEN
   MONTH='AUG'
ELSEIF(IMONTH.EQ.9)THEN
   MONTH='SEP'
ELSEIF(IMONTH.EQ.10)THEN
   MONTH='OCT'
ELSEIF(IMONTH.EQ.11)THEN
   MONTH='NOV'
ELSEIF(IMONTH.EQ.12)THEN
   MONTH='DEC'
ENDIF
ENDIF
RETURN
END

Appendix F.3: Function IDOTW

INTEGER FUNCTION IDOTW(IDAYNO)
C******************************************************************************
C**    FUNCTION RETURNS THE DAY OF THE WEEK NUMBER FROM THE DAY NUMBER
C**    IDAYNO - DATE DAY NUMBER
C**    IDOTW - DATE DAY OF THE WEEK NUMBER
C******************************************************************************
C
M=IDAYNO+5
IDOTW=MOD(M,7)
IF(IDOTW.EQ.0)IDOTW=7
RETURN
END
Appendix F.4: Subroutine ANALYS

SUBROUTINE ANALYS(N)

Performs forward pass to determine tasks Earliest Start day numbers (IESDNO), Earliest Start day of the week numbers (IESDOW), Earliest finish day numbers (IEFDNO) and Earliest Finish day of the week numbers (IEFDOW).

Performs backward pass to calculate Latest Finish day numbers (LFDNO), Latest finish day of the week number (LFDOW), Latest Start day number (LSDNO) and latest start day of the week numbers (LSDOW).

Calculate Hammock tasks durations and Identify Critical tasks in the network (ICRIT).

NSSTART - Number of the first task in the network

NFINISH - Number of the last task in the network

IPSNO - Project start day number

IPSOW - Project start day of the week number

NLINK - Number of links

NTASK - Number of tasks

N - Run number

...Initialise project finish day number (IPFDNO)

IPFDNO=0

...Search for START task number (NSTART)

DO 10, I=1, NTASK
    IF(SFCODE(I).EQ.0) NSTART=I
10 CONTINUE
c...Assign, project start, day number (IPSDNO) and day of
the week number (IPSDOW) to tasks (IESDNO), (IEFDNO) and
(IESDOW), (IEFDOW) respectively

DO20, I=1, NTASK
IESDNO(I)=IPSDNO
IEFDNO(I)=IPSDNO
IESDOW(I)=IPSDOW
IEFDOW(I)=IPSDOW
20 CONTINUE

c...Set the number of non-working days
...for the start task to zero

IDURN2(NSTART)=0

30 K=0

DO50, I=1, NLINK
IFROM=LFROM(1,1)
ITO=LTO(1,1)

c...Ignore Hammock tasks
IF(TTYPE(IFROM).EQ. 'H' . OR. TTYPE(ITO).EQ. 'H') COTO50

M=1

c...Calculate the absolute integer link durations
ILDURN(I,1)=LINKDN(I, M, IFROM)

IF(LTYPE(I,1).EQ.1)THEN

IADD=SFCODE(IFROM)

ITIME=IEFDNO(IFROM)+IADD
ITEMP=IDOTW(ITIME)

308
Using the Function (ISADED), Calculate 

the additional non working days

\[
ISADD = ISADED(ITO,ITEMP)
\]

Find out the early start day of the week number of the link duration

\[
IESLD = ITIME + ISADD \\
IDOWL = IDOTW(IESLD)
\]

Calculate the additional non working days to the link duration

\[
ILADD = (\text{INT}((ILDURN(I,1)+IDOWL-2)/TWK(IFROM)))*(7-TWK(IFROM))
\]

Recalculate (ITIME), (ITEMP)

\[
ITIME = IESLD + ILDURN(I,1) + ILADD \\
ITEMP = IDOTW(ITIME)
\]

Calculate the additional number of non-working days

\[
ISADD = ISADED(ITO,ITEMP)
\]

Adjust (ITIME)

\[
ITIME = ITIME + ISADD
\]

For (S-S) link type

ELSEIF (LTYPE(I,1).EQ.2) THEN

Calculate the additional number of non-working days to the link duration

\[
ILADD = (\text{INT}((ILDURN(I,1)+IESDOW(IFROM)-2)/TWK(IFROM)))*(7-TWK(IFROM))
\]

Calculate (ITIME), (ITEMP)

\[
ITIME = IESDNO(IFROM) + ILDURN(I,1) + ILADD \\
ITEMP = IDOTW(ITIME)
\]

Calculate the additional number of non-working days

\[
ISADD = ISADED(ITO,ITEMP)
\]

Adjust (ITIME)

\[
ITIME = ITIME + ISADD
\]

For (F-F) or (S-F) link types

ELSE
...No calculation for these type of links
ITIME=IESDNO(ITO)
ENDIF

...Compare (ITIME) with succeeding task (IESDNO) and divert the program
IF(ITIME.LE.IESDNO(ITO))GOTO40

...Update the repeating loop counter
K=K+1

...Assign ITIME to the succeeding task (IESDNO)
IESDNO(ITO)=ITIME

...Update project finish day number (IPFDNO)
IF(ITIME.GT.IPFDNO)IPFDNO=ITIME

...Calculate the early start day of the week number (IESDOW) and the non working days (IDURN2) for the succeeding task
IESDOW(ITO)=IDOW(ITIME)
IDURN2(ITO)=(INT((IDURN(ITO)+IESDOW(ITO)-2)/TWK(ITO)))*(7-TWK(ITO))

...For each type of link, calculate tasks (IEFDNO),(IEFDOW)
...For 'F-S' or 'S-S' link types
40 IF(LTYPE(1,1).EQ.1.OR.LTYPE(1,1).EQ.2)THEN
...Calculate number of non-working days to be subtracted...(Because of the use of day numbers)
ITNADD=2-SFCODE(ITO)
...Using the earliest start day number calculate...the temporary earliest finish day number (ITIME2)
ITIME2=IESDNO(ITO)+IDURN(ITO)+IDURN2(ITO)-ITNADD
...For 'F-F' link type
ELSEIF(LTYPE(1,1).EQ.3)THEN
...Calculate the additional non working days to the link duration
ILADD=(INT((IEFDOW(IFROM)+ILDURN(1,1)-2)/TWK(ITO)))*(7-TWK(ITO))
c...Calculate the temporary earliest finish day number (ITIME2)
c...and earliest finish day of the week number (ITEMP2)
c
   ITIME2=IEFDNO(IFROM)+ILDURN(I,1)+ILADD
   ITEMP2=IDOTW(ITIME2)
c
   c...Check if any more days to be added and calculate it
   c
   ISADD=ISADED(ITO,ITEMP2)
c
   c...Adjust (ITIME2)
c
   ITIME2=ITIME2+ISADD
   c
   c...For 'S-F' link type
   c
   ELSE
   c
   c...Calculate the additional non-working days to the link duration
   c
   ILADD=\left(\left\lfloor \frac{IESDOW(IFROM)+ILDURN(I,1)-2}{TWK(ITO)} \right\rfloor \times (7-TWK(ITO)) \right)
   c
   c...Calculate (ITIME2)
c
   ITIME2=IESDNO(IFROM)+ILDURN(I,1)+ILADD-1
   ENDIF
   c
   c...Compare (ITIME2) with succeeding task (IEFDNO); Divert
   c... program to store the greatest and calculate (IEFDOW)
c
   IF(ITIME2.LE.IEFDNO(ITO))GOTO50
   IEFDNO(ITO)=ITIME2
   IEFDOW(ITO)=IDOTW(ITIME2)
   50 CONTINUE
   c
   c...If counter updated; Repeat the loop
   c
   IF(K.GT.0)GOTO30
   c
   c...Identify FINISH task number (NFINISH)
c
   DO60,1=1,NTASK
   IF(SFCODE(I).EQ.2)NFINISH=I
   60 CONTINUE
   311
..Initialise tasks latest finish day numbers (LFDNO), latest start day numbers (LSDNO), latest finish day of the week numbers (LFDOW) and latest start day of the week numbers (LSDOW)

```
DO70,1=1,NTASK
   LFDNO(I)=IPFDNO
   LSDNO(I)=IPFDNO
   LFDOW(I)=IDOTW(IPFDNO)
   LSDOW(I)=IDOTW(IPFDNO)
   CONTINUE
```

..Initialise repeating loop counter

```
80 K=0
```

..Carry out backward pass to calculate tasks (LFDNO), (LFDOW) and (LSDNO), (LSDOW)

```
DO100,1=1,NLINK
   IFROM=LFROM(I,2)
   ITO=LTO(1,2)
```

..Ignore Hammock tasks

```
    IF(TTYPE(IFROM).EQ.'H'.OR.TTYPE(ITO).EQ.'H')GOT0100
```

..Name the backward pass

```
   M=2
```

..Using the function (LINKDN) calculate the Integer link durations

```
   ILDURN(1,2)=LINKDN(I,M,IFROM)
```

..Select the link type and perform calculations; For (F-S) link type

```
   IF(LTYPE(I,2).EQ.1)THEN
```

..Identify the number of days to deducted

```
   IDDCT=SFCODE(IFROM)
```

..Calculate temporary latest finish day number

```
   ITIME=LSDNO(ITO)-IDDCT
   ITEMP=IDOTW(ITIME)
```

..Using the Function (ISDEDC), Calculate

```
   ISDDCT=ISDEDC(IFROM,ITEMP)
```
c...Adjust (ITIME) and Recalculate (ITEMP)

c    ITIME=ITIME-ISDDCT
    ITEMP=IDOTW(ITIME)

c...Calculate the non-working days for the link duration

c    ILDDCT=INT((ILDURN(I,2)-ITEMP+TWK(IFROM)-1)/TWK(IFROM))*7-TWK(IFROM)

c...Recalculate (ITIME) and (ITEMP)

c    ITIME=ITIME-ILDURN(I,2)-ILDDCT
    ITEMP=IDOTW(ITIME)

c...Calculate the number of non-working days to be deducted

c    ISDDCT=ISDED(IFROM,ITEMP)

c...Adjust (ITIME)

c    ITIME=ITIME-ISDDCT

c...For (F-F) link type

c    ELSEIF(LTYPE(I,2).EQ.3)THEN

c...Calculate the non-working days of the link duration

c    ILDDCT=INT((ILDURN(I,2)-LFDOW(ITO)+TWK(ITO)-1)/TWK(ITO))*7-TWK(ITO)

c...Calculate (ITIME) and (ITEMP)

c    ITIME=LFDNO(ITO)-ILDURN(I,2)-ILDDCT
    ITEMP=IDOTW(ITIME)

c...Calculate the additional non-working days to be subtracted

c    ISDDCT=ISDED(IFROM,ITEMP)

c...Adjust the temporary latest finish day number (ITIME)

c    ITIME=ITIME-ISDDCT

c...For (S-S) or (S-F) link types

c    ELSE

c...NO calculation for this types of link is required

c    ITIME=LFDNO(IFROM)

ENDIF
c...Compare (ITIME) with the stored (LFDNO) and divert the program
   IF (ITIME.GE.LFDNO(IFROM)) GOTO 0

   c
   c...Update (LFDNO) and calculate (LFDOW)
   LFDNO(IFROM) = ITIME
   LFDOW(IFROM) = IDOTW(ITIME)

   c...Update Repeating loop counter
   K = K + 1

   c...Calculation of latest start day number (LSDNO)
   c...and latest start day of the week number (LSDOW)
   c...For 'F-S' or 'F-F' link types
   90 IF (LTYPE(I,2).EQ.1.OR.LTYPE(I,2).EQ.3) THEN
   c...Calculate the non-working days
   IDDUCT = (INT((IDURN(IFROM) - LFDOW(IFROM) - TWK(IFROM) - 1) / TWK(IFROM) &)) * (7 - TWK(IFROM))
   c...Calculate additional days 1 or 0 (because of using day numbers)
   ITNADD = SFCODE(IFROM)
   c...Using task (LFDNO); Calculate the temporary latest start
day number (ITIME2) and day of the week number (ITEMP2)
   c
   c...Check if any non-working days to be added and calculate it
   c
   ISADD = ISADED(IFROM,ITEMP2)
   c...Adjust the temporary earliest start day number
   c
   c...For a 'S-S' link type
   ELSE IF (LTYPE(I,2).EQ.2) THEN

   c...Calculate the number of non-working days
   c
   IDDUCT = (INT((ILDURN(I,2) - LSDOW(ITO) + TWK(IFROM) - 1) / TWK(IFROM)) &*) (7 - TWK(IFROM))
Calculate (ITIME2) and (ITEMP2)

\[
\begin{align*}
\text{ITIME2} & = \text{LSDNO}(\text{ITO}) - \text{ILDURN}(1,2) - \text{IDDUCT} \\
\text{ITEMP2} & = \text{IDOTW}(\text{ITIME2})
\end{align*}
\]

Calculate the number of non-working days

\[
\text{ISDDCT} = \text{ISDEDC}(\text{IFROM}, \text{ITEMP2})
\]

Adjust the temporary latest start day number (ITIME2)

\[
\text{ITIME2} = \text{ITIME2} - \text{ISDDCT}
\]

For 'S-F' link type

ELSE

Calculate the number of non-working days

\[
\text{IDDUCT} = \left(\text{INT}\left(\left(\text{ILDURN}(1,2) - \text{LFDOW}(\text{ITO}) + \text{TWK}(\text{IFROM}) - 1\right) / \text{TWK}(\text{IFROM})\right)\right) \\
& \times (7 - \text{TWK}(\text{IFROM}))
\]

Calculate (ITIME2)

\[
\text{ITIME2} = \text{LFDNO}(\text{ITO}) - \text{ILDURN}(1,2) - \text{IDDUCT} + 1
\]

ENDIF

Compare the temporary with the stored (LSDNO) and divert program

IF (ITIME2 .GE. LSDNO(IFROM)) GOTO 100

Store (ITIME2) as preceding task (LSDNO) and calculate (LSDOW)

\[
\begin{align*}
\text{LSDNO}(\text{IFROM}) & = \text{ITIME2} \\
\text{LSDOW}(\text{IFROM}) & = \text{IDOTW}(\text{ITIME2})
\end{align*}
\]

CONTINUE

Counter updated; Repeat the loop

IF (K.GT.0) GOTO 80

Counter equal to zero (not updated)

Calculate the project finish day of the week number

\[
\text{IPFDOW} = \text{IDOTW}(\text{IPFDNO})
\]

Project finish day is the first day of the week

IF (IPFDOW.EQ.1) THEN

315
Deduct the number of non-working days to calculate the exact project finish day and day of the week numbers

\[ \text{IPFDNO} = \text{IPFDNO} - (8 - \text{TWK(\text{NFINISH})}) \]
\[ \text{IPFDOW} = \text{IDOTW(\text{IPFDNO})} \]

Project finish day is not the first day in the week

ELSE

Calculate the exact project finish day and day of the week numbers

\[ \text{IPFDNO} = \text{IPFDNO} - 1 \]
\[ \text{IPFDOW} = \text{IDOTW(\text{IPFDNO})} \]
ENDIF

Calculate the project duration (IPRDRN) in days

\[ \text{IPRDRN(N)} = \text{IPFDNO} - \text{IESDNO(\text{NSTART})} + 1 \]

Initialise (IEFDNO) for hammock tasks

DO110, 1=1, NTASK

Select hammock task

IF(TTYPE(I) .NE. 'H') GOTO110
IEFDNO(I) = IPFDNO
110 CONTINUE

Calculation of HAMMOCK tasks durations

DO140, 1=1, NTASK

Select the hammock task

IF(TTYPE(I) .NE. 'H') GOTO140
DO120, J=1, NLINK

Using forward pass; Select the preceding task

IF(LTO(J,1) .NE. I) GOTO120
IFROM = LFROM(J,1)
ITO = LTO(J,1)

Identify the forward pass number

M = 1

Calculate the absolute Integer link duration

ILDURN(J,1) = LINKDN(J,M,IFROM)
c...Perform calculations for 'S-S' link type

IF(LTYPE(J,1).EQ.2)THEN

c...Calculate the non-working days to be added to the link duration

ILADD=(INT(((ILDURN(J,1)+IESDOW(IFROM)-2)/TWK(IFROM)))*(7-TWK
&IFROM))

c...Calculate temporary Earliest start day number

ITIME=IESDNO(IFROM)+ILDURN(J,1)+ILADD

c...Perform calculations for 'F-S' link type

ELSEIF(LTYPE(J,1).EQ.1)THEN

c...Calculate the temporary earliest start day number

ITSDNO=IESDNO(IFROM)+ILDURN(IFROM)+ILDURN2(IFROM)
ITSDOW=IDOTW(ITSDNO)
ISADD=ISADED(I,ITSDOW)
IESLD=ITSDNO+ISADD
IOWLIN=IDOTW(IESLD)

c...Calculate the additional days to the link duration

ILAD=(INT(((ILDURN(J,1)+IOWLIN-2)/TWK(IFROM)))*(7-TWK(IFROM)))

c...Recalculate the temporary earliest start day number

ITIME=ITSDNO+ILDURN(J,1)+ILAD
ENDIF

c...Calculate the temporary earliest start day of the week number

ITEMP=IDOTW(ITIME)

c...Check if any days to be added and calculate

ISADD=ISADED(ITO,ITEMP)

c...Adjust (ITIME)

ITIME=ITIME+ISADD

c...Calculate the Earliest Start day number (IESDNO)
c...and Earliest start day of the week number (IESDOW)

IF(ITIME.LE.IESDNO(I))GOTO120
IESDNO(I)=ITIME
IESDOW(I)=IDOTW(ITIME)
120 CONTINUE
c...Using the forward pass; Select the succeeding task

c DO130,K=1,NLINK
    IF(LFROM(K,1).NE.1)GOTO130
    LFROM=LFROM(K,1)
    ITO=LTO(K,1)
    M=1

  c...Calculate the Integer link duration.
  c  ILDURN(K,1)=LINKDN(K,M,IFROM)

  c...Perform calculations for 'F-F' link type
  c  IF(LTYPE(K,1).EQ.3)THEN
  c...If the succeeding task is 'FINISH' task
  c    IF(SFCODE(ITO).EQ.2)THEN
  c      c...Adjust the Earliest finish day number
  c      ITEMPE2=IEFDNO(ITO)-1
  c      ITEMPE3=IDOTW(ITEMPE2)
  c      ISDDCT=ISDEDC(ITO,ITEMPE3)
  c      ITEMPE2=ITEMPE2-ISDDCT
  c      IDOWT=IDOTW(ITEMPE2)\
  c      c...Calculate the additional non working days to the link duration
  c      IDDUCT=(INT((ILDURN(K,1)-1DOWT+TWK(ITO)-1)/TWK(ITO)))*(7-TWK(IITO))\
  c      c...Calculate the temporary earliest finish day number
  c      c...and earliest finish day of the week number
  c      IEFDNTE=ITEMPE2-ILDURN(K,1)-IDDUCT
  c      IEFDWT=IDOTW(IEFDNTE)
  c
  c...For any other type of succeeding task
  c    ELSE
  c    c...Calculate the additional non working days to link duration
  c    IDDUCT=(INT((ILDURN(K,1)-IEFDOW(ITO)+TWK(ITO)-1)/TWK(ITO)))*7-TWK(IITO))\
  c    c...Calculate the temporary earliest finish day number
  c...and earliest finish day of the week number
  c    IEFDNTE=IEFDNO(ITO)-ILDURN(K,1)-IDDUCT
  c    IEFDWT=IDOTW(IEFDNTE)\
  c
ENDIF
...For 'F-S' link type

ELSEIF(LTYPE(K,1).EQ.1)THEN

...Calculate temporarily earliest finish day number
...and earliest finish day of the week number

ITIME=IESDNO(ITO)-1
ITEMP=IDOTW(ITIME)

...Check if any number of days to be subtracted and calculate it

ISDDCT=ISDEDC(IFROM,ITEMP)

...Adjust earliest finish day and day of the week numbers

IEFDNT=ITEMP-ISDDCT
IEFDWT=IDOTW(IEFDNT)

...calculate the additional number of non working days

IDDUCT=(INT((ILDURN(K,1)-IEFDWT+TWK(IFROM)-1)/TWK(IFROM)))*((67-TWK(IFROM))

...Recalculate temporarily day and day of the week numbers

IEFDNT=IEFDNT-ILDURN(K,1)-IDDUCT
IEFDWT=IDOTW(IEFDNT)
ELSE
IEFDNT=IEFDNO(IFROM)
IEFDWT=IDOTW(IEFDNT)
ENDIF

...Check if any non working days to be subtracted and calculate it

ISDDCT=ISDEDC(IFROM,IEFDWT)

...Calculate (IEFDNO), (IEFDOW)

IEFDNT=IEFDNT-ISDDCT
IF(IEFDNT.GE.IEFDNO(IFROM))GOTO130
IEFDNO(I)=IEFDNT
IEFDOW(I)=IDOTW(IEFDNO(I))

130 CONTINUE

...Calculate the hammock task total duration
...(including non working days)

IHTDRN=IEFDNO(I)-IESDNO(I)+1

...Calculate the number of non working days

IDURN2(I)=(INT((IHTDRN+IESDOW(I)-1)/7))*(7-TWK(I))
c...Calculate Hammock task duration
  c
  IDURN(I)=IHTDRN-IDURN2(I)
140 CONTINUE
  c...Update criticality Index (ICRIT)
  c
  DO150,1=1,NTASK
  c...Ignore Hammock tasks
  c
  IF(TTYPE(I).EQ.'H')GOTO150
  c...Calculate the float
  c
  IFLOAT=LFDNO(I)-(IESDNO(I)+IDURN(I)+IDURN2(I))+1
  c...Check if task is critical
  c
  IF(IFLOAT.LE.0)ICRIT(I)=ICRIT(I)+1
150 CONTINUE
  c...Exit
  c
  RETURN
END


Appendix F.5: Function LINKDN

INTEGER FUNCTION LINKDN(I, M, IFROM)

FUNCTION LINKDN

** Changes (S-S) link duration to absolute durations **
** Changes all types of link durations to Integers **

**

COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400, 3), SFCODE(400),
& TWK(400), TCOST(400, 9), TWBS(400), PCCODE(400), IDURN(400)
COMMON/LINKS/LFROM(500, 2), LTO(500, 2), LTYPE(500, 2), LDURN(500, 2)

CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWBS

REAL TDURN, LDURN, TCOST
INTEGER SFCODE, TWK, PCCODE

...Change (S-S) link durations from percentage to absolute...

IF(LTYPE(I, M).EQ.2) THEN
  TEMP=LDURN(I, M)*FLOAT(IDURN(IFROM))
ENDIF

...Change link duration to Integer
LINKDN=INT(TEMP)
IF(TEMP.GT.AINT(TEMP))LINKDN=LINKDN+1
ELSE
LINKDN=INT(LDURN(I, M))
IF(LDURN(I, M).GT.AINT(LDURN(I, M)))LINKDN=LINKDN+1
ENDIF

...Exit
RETURN
END
Appendix F.6: Function ISADED

INTEGER FUNCTION ISADED(I, IDOWNO)

C ******************************************************************************
C ** FUNCTION ISADED **
C ** Returns the number of non-working days (weekends) **
C ** to be added to the tasks durations to calculate the **
C ** Earliest start day number of a succeeding task(IESDNO) **
C ******************************************************************************

C COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400,3),SFCODE(400),
& TWK(400),TCOST(400,9),TWBS(400),PCCODE(400),IDURN(400)
C
CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWBS
C
REAL TDURN
INTEGER SFCODE,TWK,PCCODE
C
...Calculate the additional number of days
C
IF(IDOWNO.GT.TWK(I))THEN
  ISADED=8-IDOWNO
ELSE
  ISADED=0
ENDIF
RETURN
END
Appendix F.7: Function ISDEDC

INTEGER FUNCTION ISDEDC(1,IDOWNO)

C ***
C** FUNCTION ISDEDC  **
C** Returns the number of non—working days (week—ends) **
C** to be deducted from the tasks durations to calculate **
C** the Latest Finisht day number of a preceding task (LFDNO) **
C**
C*******
C
COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400,3),SFCODE(400),
&TWK(400),TCOST(400,9),TWBS(400),PCCODE(400),IDURN(400)

C
CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWBS

C
REAL TDURN
INTEGER SFCODE,TWK,PCCODE

C
C...Calculate the additional number of days
C
IF(IDOWNO.GT.TWK(I))THEN
   ISDEDC=IDOWNO—TWK(I)
ELSE
   ISDEDC=0
ENDIF
RETURN
END
SUBROUTINE TRIDIS(I)

Calculates triangular distribution parameters for the Ith task

COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400,3), SFCODE(400), &TWK(400), TCOST(400,9), TWBS(400), PCCODE(400), IDURN(400)
COMMON/PDIST/DLIM(400,2), FMODE(400)

CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWBS

REAL TDURN, TCOST
INTEGER SFCODE, TWK, PCCODE

CONVER=l.0E-6

FMODE(I)=2.0/(DEL1+DEL2+DEL3+DEL4)

IF(DEL2 GT 0.0)THEN
   DEL1=(0.02+SQRT(0.02**2+0.08*FMODE(I)*DEL2))/(2.0*FMODE(I))
ENDIF
DEL4=(0.02+SQRT(0.02**2+0.08*FMODE(I)*DEL3))/(2.0*FMODE(I))
PTOTAL=0.5*FMODE(I)*(DEL1+DEL2+DEL3+DEL4)
IF((ABS(1.0-PTOTAL)).GT. CONVER)GOTO10

DLIM(I,1)=TDURN(I,1)-DEL1
DLIM(I,2)=TDURN(I,3)+DEL4

RETURN
END
SUBROUTINE DSELECT(I, SEED)

SUBROUTINE DSELECT

C ** Randomly selects an INTEGER task duration IDURN on
C ** the basis of the duration distribution parameters
C ** calculated by Subroutine TRIDIS
C

COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400,3), SFCODE(400),
& TWK(400), TCOST(400,9), TWBS(400), PCCODE(400), IDURN(400)
COMMON/PDIST/DLIM(400,2), FMODE(400)

CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWBS

REAL TDURN, TCOST
INTEGER SFCODE, TWK, PCCODE

c...Calculate RH slope of distribution
Sl=FMODE(I)/(DLIM(I,2)-TDURN(I,2))

c...Check whether duration can be less than modal duration
DLEFT=TDURN(I,2)-DLIM(I,1)
IF(DLEFT.GT.0.0)THEN

S0=FMODE(I)/DLEFT
A0=0.5*DLEFT**2*FMODE(I)
ELSE
A0=0.0
ENDIF

A=RAND(SEED)
IF(A.LT.A0)THEN
DURN=DLIM(I,1)+SQRT(2.0*A/S0)
ELSE
DURN=DLIM(I,2)-SQRT(2.0*(1.0-A)/SI)
ENDIF
IDURN(I)=INT(DURN)
IF(DURN.GT.AI(N))IDURN(I)=IDURN(I)+1
Appendix F.10: Function RAND

FUNCTION RAND(SEED)
C************************************************************
C** FUNCTION RAND **
C** Returns a random REAL number in the range 0. to 1.0 **
C** Requires initialisation by 0.0 < seed < 259200 **
C** **
C************************************************************
C
PARAMETER(IA=7141,IC=54773,IM=259200)
C
SEED=REAL(MOD(INT(SEED)*IA+IC,IM))
RAND=SEED/REAL(IM)
RETURN
END
Appendix F.11: Subroutine PCOSTS

SUBROUTINE PCOSTS(NSTART, NTASK, ICDAYS, N)

C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
C******************************************************************************
COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400, 3), SFCODE(400),
& TWK(400), TCOST(400, 9), TWBS(400), PCCODE(400), IDURN(400)
COMMON/DAYNOS/IESDNO(400), IESDOW(400), IDURN2(400), LFDNO(400),
& LFDOW(400), IDURN3(400), LDURN(500, 2), IDRIT(400), ACRIT(400),
& IPRDRN(1000), IEFDNO(400), IEFDOW(400), LSDNO(400), LSDOW(400)
COMMON/FINSTA/ISTART(400), IFINISH(400), NDAYOW(IO00)
COMMON/PDCOST/DCOSTS(500, 9), COSPAY(500, 9), COSDLY(500, 9),
& IPRDRN(500), MACDAY(50), TOTCOS(500), ACTOTC(500), ACCTPR(500),
& CAPTIM(1000), RATERN(IO00)
COMMON/COSTS/CCODE(9), CPAY(9), CARR(9), CARRQ(9)
C
CHARACTER*1 TTYPE, CPAY, CARR
CHARACTER*3 CCODE
CHARACTER*9 TNAME
CHARACTER*12 TWBS
C
REAL TDURN, TCOST, X
INTEGER SF CODE, TWK, PCCODE, CARRQ
C
..Calculate the maximum payment delay
C
MAXDLY=0
D05, I=1, 9
IF(CARR(I).EQ.'D')THEN
  IF(MAXDLY.LT.CARRQ(I))MAXDLY=CARRQ(I)
ELSEIF(CARR(I).EQ.'W')THEN
  NDAYS=CARRQ(I)*7
  IF(MAXDLY.LT.NDAYS)MAXDLY=NDAYS
ELSEIF(CARR(I).EQ.'M')THEN
  NDAYS=CARRQ(I)*30
  IF(MAXDLY.LT.NDAYS)MAXDLY=NDAYS
ENDIF
5 CONTINUE
C
..Add maximum payment delay to the project duration
C
..Add 30 Days to cover the payment method........
C
ICDAYS=IPRDRN(N)+MAXDLY+30

327
c...Calculate start (ISTART) and finish 
c...(IFINISH) day number for each task 
c
DO10,1=1,NTASK
  ISTART(I)=IESDNO(I)-IESDNO(NSTART)+1
  IFinish(I)=ISTART(I)+IDURN(I)+IDURN2(I)-1
10 CONTINUE

c...Calculate the day of the week number (NDAYOW) 
c...and date (IDDATE) for each day 
c
IDAYNO=IESDNO(NSTART)
DO20,1=1,ICDAYS
  NDAYOW(I)=IDOTW(IDAYNO)
  IDDATE(I)=IDAY(IDAYNO)
  IDAYNO=IDAYNO+1
20 CONTINUE

c...Find out the monthly accumulation days (MACDAY)
c
J=1
DO30,1=2,ICDAYS
  IF(IDDATE(I).NE.1)GOTO30
  MACDAY(J)=I-1
  J=J+1
30 CONTINUE

c...Initialise daily costs arrays 
c
DO50,K=1,9
  DO40,1=1,ICDAYS
    DCOSTS(I,K)=0.0
    COSPAY(I,K)=0.0
    COSDLY(I,K)=0.0
40 CONTINUE
50 CONTINUE

c...Distribute tasks daily labour cost on 
c...the project days (Normal loading) 
c
DO70,1=1,ICDAYS
  DO60,J=1,NTASK
    IF(NDAYOW(I).GT.TWK(J))GOTO60
    IF(I.GE.ISTART(J).AND.I.LE.IFINISH(J))DCOSTS(I,1)=DCOSTS(I,1)+
      &TCOST(J,1)
60 CONTINUE
70 CONTINUE
c...Materials cost (Front loading)
cDO90, I=1, ICDAYS
DO80, J=1, NTASK
   IF (I.EQ. ISTART(J)) DCOSTS(I,2) = DCOSTS(I,2) + TCOST(J,2)
  80   CONTINUE
  90   CONTINUE
c...Daily plant costs (Normal loading)
  DO110, I=1, ICDAYS
  DO100, J=1, NTASK
     IF (NDAYOW(I).GT. TWK(J)) GOTO100
     IF (I.GE. ISTART(J) .AND. I.LE. IFINISH(J)) DCOSTS(I,3) = DCOSTS(I,3) +
     & TCOST(J,3)
   100   CONTINUE
  110 CONTINUE
c...Front-end plant cost (Front loading)
  DO130, I=1, ICDAYS
  DO120, J=1, NTASK
     IF (I.EQ. ISTART(J)) DCOSTS(I,4) = DCOSTS(I,4) + TCOST(J,4)
  120 CONTINUE
  130 CONTINUE
c...Back-end plant cost (Back loading)
  DO150, I=1, ICDAYS
  DO140, J=1, NTASK
     IF (I.EQ. IFINISH(J)) DCOSTS(I,5) = DCOSTS(I,5) + TCOST(J,5)
  140 CONTINUE
  150 CONTINUE
c...Subcontracts cost (Spread loading)
  DO170, I=1, ICDAYS
  DO160, J=1, NTASK
     IF (NDAYOW(I).GT. TWK(J)) GOTO160
     IF (TCOST(J,6).EQ.0.0) GOTO160
     IF (I.GE. ISTART(J) .AND. I.LE. IFINISH(J)) DCOSTS(I,6) = DCOSTS(I,6) +
     & (TCOST(J,6)/IDURN(J))
  160 CONTINUE
  170 CONTINUE
c...Daily overheads cost (Normal)
  DO190, I=1, ICDAYS
  DO180, J=1, NTASK
     IF (NDAYOW(I).GT. TWK(J)) GOTO180
     IF (I.GE. ISTART(J) .AND. I.LE. IFINISH(J)) DCOSTS(I,7) = DCOSTS(I,7) +
     & TCOST(J,7)
  180 CONTINUE
  190 CONTINUE
c...Front-end overheads cost (Front loading)

c DO210, I=1, ICDAYS
 DO200, J=1, NTASK
    IF(I.EQ.ISTART(J)) DCOSTS(I,8) = DCOSTS(I,8) + TCOST(J,8)
 200 CONTINUE
 210 CONTINUE

c...Task prices (Spread loading)

c DO230, I=1, ICDAYS
 DO220, J=1, NTASK
    IF(NDAYOW(I).GT.TWK(J)) GOTO220
    IF(TCOST(J,9).EQ.0.0) GOTO220
    IF(I.GE.ISTART(J).AND.I.LE.IFINISH(J)) DCOSTS(I,9) = DCOSTS(I,9) +
      (TCOST(J,9)/IDURN(J))
 220 CONTINUE
 230 CONTINUE

c...Accumulate Daily cost according to the payment method

c DO320, K=1, 9

c...Weekly payment

 IF(CPAY(K).EQ.'W') THEN
    ACWCOS=0.0
    DO240, I=1, ICDAYS
      IF(NDAYOW(I).NE.5) THEN
        ACWCOS=ACWCOS+DCOSTS(I, K)
      ELSE
        COSPAY(I,K)=ACWCOS+DCOSTS(I,K)
        ACWCOS=0.0
      ENDIF
    240 CONTINUE
 ELSE

 c...Monthly payment

 J=1
 ACMCOS=0.0
 DO250, I=1, ICDAYS
    IF(I.NE.MACDAY(J)) THEN
      ACMCOS=ACMCOS+DCOSTS(I, K)
    ELSE
      COSPAY(I,K)=ACMCOS+DCOSTS(I,K)
      J=J+1
      ACMCOS=0.0
    ENDIF
 250 CONTINUE

 ENDIF
c...Arrange Commulative Cost according to the given payment delay

  IF(CARR(K).EQ.'O')THEN
  c...Payment delay not applicable
  DO260, I=1, ICDAYS
    COSDLY(I, K) = COSPAY(I, K)
  260 CONTINUE
  c...Payment delay in days
  ELSEIF(CARR(K).EQ.'D')THEN
   IARR = CARRQ(K)
   DO270, I=1, ICDAYS
     J = I + IARR
     COSDLY(J, K) = COSPAY(I, K)
  270 CONTINUE
  c...Payment delay in weeks
  ELSEIF(CARR(K).EQ.'W')THEN
   IARR = CARRQ(K) * 7
   DO280, I=1, ICDAYS
     J = I + IARR
     COSDLY(J, K) = COSPAY(I, K)
  280 CONTINUE
  c...Payment delay in months
  ELSE
    IF(CPAY(K).EQ.'W')THEN
      IARR = CARRQ(K) * 30
      DO290, I=1, ICDAYS
        J = I + IARR
        COSDLY(J, K) = COSPAY(I, K)
      290 CONTINUE
    ELSE
      J = 1
      M = J + CARRQ(K)
      DO300, I=1, ICDAYS
        IF(I .NE. MACDAY(M)) GOTO300
        COSDLY(I, K) = COSPAY(MACDAY(J), K)
        J = J + 1
        M = M + 1
      300 CONTINUE
    ENDIF
  ENDIF
c...Adjust payments not to be in week-ends

D0310, I=1, ICDAYS
IF(COSDLY(I,K).EQ.0.0) GOTO310
IF(NDAYOW(I).EQ.6) THEN
   COSDLY(I-1,K)=COSDLY(I,K)+COSDLY(I-1,K)
   COSDLY(I,K)=0.0
ELSEIF(NDAYOW(I).EQ.7) THEN
   COSDLY(I-2,K)=COSDLY(I,K)+COSDLY(I-2,K)
   COSDLY(I,K)=0.0
ENDIF
310 CONTINUE

320 CONTINUE

...Calculate Total cost (TOTCOS)

D0330, I=1, ICDAYS
TOTCOS(I)=COSDLY(I,1)+COSDLY(I,2)+COSDLY(I,3)+COSDLY(I,4)+
& COSDLY(I,5)+COSDLY(I,6)+COSDLY(I,7)+COSDLY(I,8)
330 CONTINUE

...Initialise cumulative total cost (ACTOTC)

D0340, I=0, ICDAYS
ACTOTC(I)=0.0
340 CONTINUE

...Accumulate total cost

D0350, I=1, ICDAYS
ACTOTC(I)=ACTOTC(I-1)+TOTCOS(I)
350 CONTINUE

...Initialise cumulative Task prices (ACCTPR)

D0360, I=0, ICDAYS
ACCTPR(I)=0.0
360 CONTINUE

...Accumulate Task Price

D0370, I=1, ICDAYS
ACCTPR(I)=ACCTPR(I-1)+COSDLY(I,9)
370 CONTINUE

...Initialise CAPTIM

CAPTIM(N)=0.0

...Calculate the CAPTIM

D0380, I=1, ICDAYS
CAPTIM(N)=CAPTIM(N)+DIM(ACTOTC(I), ACCTPR(I))
380 CONTINUE

332
c...Initialise annual interest rate (X)
X=-1.0
c

...Set Increasing of annual Interest rate (DELX) to 10%
DELX=0.1
X=X+DELX
c

...Initialise Net Present Value (ACCNPV)
390 ACCNPV=0.0
R=((1.0+X)**(1.0/365.0))-1.0
c

...Calculate Net Present Value (ACCNPV)
D0400,1=1,1CDAYS
ACCNPV=ACCNPV+(COSDLY(1,9)-TOTCOS(I))/((1.0+R)**(I-1))
400 CONTINUE
c
...Net present value is negative
IF(ACCNPV.GT.0.0)THEN

...Increase Annual Intrest rate and Re-calculate (ACCNPV)
X=X+DELX
GOTO390
ELSE
c
...Net Present Value is Positive
...Deduct the previously seted DELX
X=X-DELX
c
...Reset DELX to 10% of the previous one
DELX=DELX/10.0
c
...Recalculate the annual Interest rate
X=X+DELX
ENDIF
IF(DELX.LE.1.0E-5)GOTO410
GOTO390
c
...Calculate Internal Rate of return
410 RATER(N)=X*100.0
c
...Exit
RETURN
END
Appendix F.12: Subroutine PSORT

SUBROUTINE PSORT(NRUNS)
    COMMON/DAYNOS/ 1ESDNO(400), IESDOW(400), IDURN2(400), LFDNO(400),
               & LFDOW(400), IDURN3(400), ILDURN(500,2), ICRIT(400), ACRIT(400),
               & IPRDRN(1000), IEFDNO(400), IEFDOW(400), LSDNO(400), LSDOW(400)
    
    DO20, I=1, (NRUNS-1)
    DO10, J=1, (NRUNS-I)
        K=J+1
        IF(IPRDRN(J).LE.IPRDRN(K))GOTO10
        ITEMP=IPRDRN(J)
        IPRDRN(J)=IPRDRN(K)
        IPRDRN(K)=ITEMP
    10 CONTINUE
20 CONTINUE
    
RETURN
END
Appendix F.13: Subroutine PRSTAT

SUBROUTINE PRSTAT(NRUNS)

C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C * *  * *
c** S U B R O U T I N E  PRSTA T **
c** **
c** C alculates Minimum, Maximum, Median, Mean pro ject durations **
c** and Project Skewness **
c** **
c** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400, 3), SFCODE(400),
&TWK(400), TCOST(400, 9), PCODE(400), TESNO(400),
&LFDNO(400), LFDNOW(400), LFDNOW(400), LFDNOW(500, 2), ICRT(400), ACRT(400),
&IFRDRN(1000), IEFDNO(400), IEFDNOW(400), LFDNOW(400), LFDNOW(400)
COMMON/STATS/MINPDU, MAXPDU, PRMEAN, PSKEW, MEDIAN, PROSIG
C
CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWKS
C
REAL MEDIAN, TDURN, TCOST
C
INTEGER SFCODE, TWK, PCODE
C
...Initialise minimum, maximum, sum and sum square of project durations
C
MINPDU=1E6
MAXPDU=0
ISUM=0
ISUMSQ=0
C
...Calculate minimum, maximum, sum and sum square
C
DO10, I=1, NRUNS
  IF(IPRDRN(I).LT.MINPDU)MINPDU=IPRDRN(I)
  IF(IPRDRN(I).GT.MAXPDU)MAXPDU=IPRDRN(I)
  ISUM=ISUM+IPRDRN(I)
  ISUMSQ=ISUMSQ+IPRDRN(I)**2
10 CONTINUE
C
...Calculate Mean project duration and Standard deviation
C
PRMEAN=FLOAT(ISUM)/FLOAT(NRUNS)
PROSIG=SQRT((FLOAT(ISUMSQ)-FLOAT(ISUM)**2/FLOAT(NRUNS))/FLOAT(NRUNS-1))


c...Calculate the Median project duration
c
    I=(NRUNS+1)/2
    J=I+1
    IF(MOD(NRUNS,2).LT.1)
        MEDIAN=FLOAT(IPRDRN(I)+IPRDRN(J))/2.0
    ELSE
        MEDIAN=FLOAT(IPRDRN(I))
    ENDIF

c...Calculate project durations Skewnees
c
    PSKEW=3.0*(PRMEAN-MEDIAN)/PROSIG

c...Exit
c
    RETURN
    END
Appendix F.14: Subroutine CSTAT

**SUBROUTINE CSTAT(NRUNS)**

Calculates Minimum, Maximum, Median, Mean CAPTIM and CAPTIM Skewness

**COMMON/TASKS/TTYPE(400), TNAME(400), TDURN(400,3), SFCODE(400), &TWK(400), TCOST(400,9), PCCODE(400), IDURN(400), \**
**COMMON/DAYNOS/IESDNO(400), IESDOW(400), IDURN2(400), LFDNO(400), \*
**&LFDOW(400), ILDURN(500,2), ICRT(400), ACRT(400), \*
**&IPRDRN(1000), IEFDO(400), IEFDOW(400), LSDLNO(400), LSDLW(400), \*
**COMMON/STAT/CAPMIN, CAPMAX, CAMEAN, CSKEW, CMEDIAN, \*
**CAPTIM COMMON/PDCOST/DCOSTS(500,9), COSPAY(500,9), COSDLY(500,9), \*
**&IDDATE(500), MACDAY(50), TOTCOS(500), ACTOTC(500), ACCTPR(500), \*
**&CAPTIM(1000), RATERN(1000) \*
**CHARACTER*1 TTYPE \*
**CHARACTER*9 TNAME \*
**CHARACTER*12 TWBS \*
**REAL TDURN, TCOST \*
**INTEGER SFCODE, TWK, PCCODE \*
**c...Initialise minimum, maximum, sum and sum square of CAPTIM \*
**CAPMIN=1.0E12 \*
**CAPMAX=0.0 \*
**SUM=0.0 \*
**SUMSQ=0.0 \*
**c...Calculate CAPTIM minimum, maximum, sum and sum square \*
**DO10, I=1, NRUNS \*
**IF(CAPTIM(I).LT.CAPMIN)CAPMIN=CAPTIM(I) \*
**IF(CAPTIM(I).GT.CAPMAX)CAPMAX=CAPTIM(I) \*
**SUM=SUM+CAPTIM(I) \*
**SUMSQ=SUMSQ+CAPTIM(I)**2 \*
**10 CONTINUE \*
**c...Calculate Mean CAPTIM and Standard deviation \*
**CAMEAN=SUM/FLOAT(NRUNS) \*
**CAPSIG=SQRT((SUMSQ-SUM**2/FLOAT(NRUNS))/FLOAT(NRUNS-1))
c

...Calculate the Median CAPTIM

c
I=(NRUNS+1)/2
J=I+1
IF(MOD(NRUNS,2).LT.1)THEN
   CMEDIAN=(CAPTIM(I)+CAPTIM(J))/2.0
ELSE
   CMEDIAN=CAPTIM(I)
ENDIF

...Calculate CAPTIM Skewnees

CSKEW=3.0*(CAMEAN-CMEDIAN)/CAPSIG

...Exit

RETURN

END
SUBROUTINE RRSTAT(NRUNS)

C**************************************************************************
C** SUBROUTINE RRSTAT **
C** Calculates Minimum, Maximum, Median, Mean for IRR **
C** and IRR Skewness **
C**************************************************************************

COMMON/TASKS/TTYPE(400),TNAME(400),TDURN(400,3),SFCODE(400),
&TWK(400),TCOST(400,9),TWBS(400),PCCODE(400),IDURN(400)
COMMON/DAYNOS/IESDNO(400),IESDOW(400),IDURN2(400),LFDNO(400),
&LFDOW(400),IDURN3(400),ILDURN(500,2),ICRIT(400),ACRIT(400),
&IPRDRN(1000),IEFDNO(400),IEFDOW(400),LSDNO(400),LSDOW(400)
COMMON/STAT2/RRMIN,RRMAX,RRMEAN,RRSKW,RRMEDN,RRSIG
COMMON/PDCOST/DCOSTS(500,9),COSPAY(500,9),COSDLY(500,9),
&IDDATE(500),MACDAY(50),TOTCOS(500),ACTOTC(500),ACCTPR(500),
&CAPTIM(1000),RATERN(1000)

CHARACTER*1 TTYPE
CHARACTER*9 TNAME
CHARACTER*12 TWBS

REAL TDURN,TCOST

INTEGER SFCODE,TWK,PCCODE

c...Initialise minimum, maximum, sum and sum square of IRR
c
RRMIN=1.0E12
RRMAX=0.0
SUM=0.0
SUMSQ=0.0

c...Calculate IRR minimum, maximum, sum and sum square
c
DO10,I=1,NRUNS
  IF(RATERN(I).LT.RRMIN)RRMIN=RATERN(I)
  IF(RATERN(I).GT.RRMAX)RRMAX=RATERN(I)
  SUM=SUM+RATERN(I)
  SUMSQ=SUMSQ+RATERN(I)**2
10 CONTINUE

c...Calculate Mean Internal Rate of Return and Standard deviation
c
RRMEAN=SUM/FLOAT(NRUNS)
RRSIG=SQRTR((SUMSQ-SUM**2/FLOAT(NRUNS))/FLOAT(NRUNS-1))
Calculate the Median IRR

I = (NRUNS + 1) / 2
J = I + 1
IF (MOD(NRUNS, 2) < 1) THEN
  RRMEDN = (RATERN(I) + RATERN(J)) / 2.0
ELSE
  RRMEDN = RATERN(I)
ENDIF

Calculate IRR Skewness

RRSKEW = 3.0 * (RRMEAN - RRMEDN) / RRSIG

Exit

RETURN
END
SUBROUTINE RRSORT(NRUNS)

C**********************************************************************
C**  SUBROUTINE RRSORT **
C**
C** Sorts Number of(NRUNS) Internal rate or return (RATERN),
C** (CAPTIM) into Rank order **
C**
C
COMMON/PDCOST/DCOSTS(500,9),COSPAY(500,9),COSDLY(500,9),
&IDDATE(500),MACDAY(50),TOTCOS(500),ACTOTC(500),ACCTPR(500),
&CAPTIM(1000),RATERN(1000)

C...Sort IRR and corresponding CAPTIM into rank order

DO20,I=1,(NRUNS-1)
DO10,J=1,(NRUNS-I)
    K=J+1
    IF(RATERN(J).LE.RATERN(K))GOTO10
    TEMP=RATERN(J)
    TEMP1=CAPTIM(J)
    RATERN(J)=RATERN(K)
    CAPTIM(J)=CAPTIM(K)
    RATERN(K)=TEMP
    CAPTIM(K)=TEMP1
10 CONTINUE
20 CONTINUE

C...Exit

RETURN
END
SUBROUTINE FREQUA(NRUNS,NBANDS)

Calculates number of bands, Frequency and cumulative in each band for the number of runs project durations

COMMON/DAYNOS/IESDNO(400),IESDOW(400),IDURN2(400),LFDNO(400),
&LFDOW(400),IDURN3(400),ILDURN(500,2),ICRIT(400),ACRIT(400),
&IPRDRN(1000),IEFDNO(400),IEFDOW(400),LSDNO(400),LSDOW(400)
COMMON/FREQU1/PCFREQ(100),PRBMPT(100),IPBAND(100),PFREQ(100),
&RCPWPD(100)
COMMON/STATS/MINPDU,MAXPDU,PRMEAN,PSKEW,MEAN,PROSIG

IBW=10

Set minimum and maximum parameters

AMIN=MINPDU
AMAX=MAXPDU

Change the band width to real number

BANDW=FLOAT(IBW)

Determine upper and lower distribution

A1=AMIN-AMOD(AMIN,BANDW)-BANDW
A2=AMAX-AMOD(AMAX,BANDW)+BANDW

Calculate number of bands and zero each band frequency

NBANDS=(IFIX(A2)-IFIX(A1))/IBW+1
IPBAND(1)=IFIX(A1)
DO40,I=2,NBANDS
   IPBAND(I)=IPBAND(I-1)+IBW
   PFREQ(I-1)=0.0
40 CONTINUE
c...Calculate frequency for each band
c  DO60, I=1, NRUNS  
    DO50, J=1, (NBANDS-1)  
      IF (IPRDRN(I).GT.IPBand(J).AND.IPRDRN(I).LE.IPBand(J+1)) THEN  
        PFREQ(J)=PFREQ(J)+1.0  
        GOTO60  
      ENDIF  
    50 CONTINUE  
  60 CONTINUE  

  c...Initialise cumulative frequency  
  c  DO70, I=1, (NBANDS-1)  
      PCFREQ(I)=0.0  
    70 CONTINUE  

  c...Calculate frequency (\%)  
  c  DO80, I=1, (NBANDS-1)  
      PFREQ(I)=PFREQ(I)*100.0/FLOAT(NRUNS)  
    80 CONTINUE  

  c...Calculate cumulative frequency (\%)  
  c  PCFREQ(1)=PFREQ(1)  
    DO90, I=2, (NBANDS-1)  
      PCFREQ(I)=PCFREQ(I-1)+PFREQ(I)  
    90 CONTINUE  

  c...Calculate bands midpoints  
  c  DO95, I=1, NBANDS  
      PRBMPT(I)=FLOAT(IPBAND(I))+BANDW/2.0  
    95 CONTINUE  

  c...Calculate the Rate of Probability with Project Duration (f(x))  
  c  A=0.39894228/PROSIG  
    DO100, I=1, (NBANDS-1)  
      RCPWPD(I)=(A*EXP(-(((PRBMPT(I)-PRMEAN)**2)/(2.0*PROSIG**2))))*10  
        &0.00*BANDW  
    100 CONTINUE  

  c...Exit  
  c  RETURN  
  END
SUBROUTINE FRQIRR(NRUNS, INBAND)

Calculates number of bands, Frequency and cumulative in each band for the number of runs IRR.

COMMON/PDCOST/DCOSTS(500,9), COSPAY(500,9), COSDLY(500,9),
& IDDATE(500), MACDAY(50), TOTCOS(500), ACTOTC(500), ACCTPR(500),
& CAPTIM(1000), RATERN(1000)
COMMON/STAT2/RRMIN, RRMAX, RRMEAN, RRSEW, RRMEDN, RRSCI
COMMON/FREQU2/CFREQ(100), BANMPT(100), IBAND(100), FREQ(100),
& ROPWRR(100)

IBW = 100

Initialise minimum and maximum parameters

AMIN = RRRMIN
AMAX = RRRMAX

Change the band width to real number

BANDW = FLOAT(IBW)

Determine upper and lower distribution

IF(AMIN.LE.0.0) THEN
  A1 = AMIN - AMOD(AMIN, BANDW) - BANDW
ELSE
  A1 = AMIN - AMOD(AMIN, BANDW)
ENDIF
A2 = AMAX - AMOD(AMAX, BANDW) + BANDW

Calculate number of bands and zero each band

INBAND = (IFIX(A2) - IFIX(A1)) / IBW + 1
IBAND(1) = IFIX(A1)
DO 40, I = 2, INBAND
  IBAND(I) = IBAND(I - 1) + IBW
  FREQ(I - 1) = 0.0
40 CONTINUE
c...Calculate frequency in each band
c
DO60, I=1, NRUNS
DO50, J=1, (INBAND-1)
   IF(RATERN(I).GT.IBAND(J).AND.RATERN(I).LE.IBAND(J+1))THEN
      FREQ(J)=FREQ(J)+1.0
   GOTO60
ENDIF
50 CONTINUE
60 CONTINUE
c...Initialise cumulative frequency
c
DO70, I=1, (INBAND-1)
   CFREQ(I)=0.0
70 CONTINUE
c...Calculate frequency (%)
c
DO80, I=1, (INBAND-1)
   FREQ(I)=FREQ(I)*100.0/FLOAT(NRUNS)
80 CONTINUE
c...Calculate cumulative frequency (%)
c
   CFREQ(1)=FREQ(1)
   DO90, I=2, (INBAND-1)
      CFREQ(I)=CFREQ(I-1)+FREQ(I)
90 CONTINUE
c...Calculate bands midpoints
c
DO95, I=1, INBAND
   BANMPT(I)=FLOAT(IBAND(I))+BANDW/2.0
95 CONTINUE
c...Calculate the Rate of Probability with IRR (f(x))
c
A=0.39894228/RRSIG
DO100, I=1, (INBAND-1)
   ROPWRR(I)=(A*EXP(-(((BANMPT(I)-RRMEAN)**2)/(2.0*RRSIG**2))))*100 &.00*BANDW
100 CONTINUE
c...Exit
c
RETURN
END
Appendix F.19: Subroutine FRQCAP

SUBROUTINE FRQCAP(NRUNS, ICNBAND)

Calculates number of bands, Frequency and cumulative in each band for the number of runs CAPTIM

COMMON/PDCOST/DCOSTS(500,9), COSPAY(500,9), COSDLY(500,9), IDATE(500), MACDAY(50), TOTCOS(500), ACTOTC(500), ACCTPR(500), CAPTIM(1000), RATERN(1000)
COMMON/FREQU3/CCFREQ(100), CABMPT(100), ICBAND(100), CAFREQ(100), ROPWCAP(100)
COMMON/STAT3/CAPMIN, CAPMAX, CAMEAN, CSKEW, CMEDIAN, CAPSIG

IBW=1E6

...Initialise minimum and maximum parameters

AMIN=CAPMIN
AMAX=CAPMAX

...Change the band width to real number

BANDW=FLOAT(IBW)

...Determine upper and lower distribution

A1=AMIN-AMOD(AMIN, BANDW)-BANDW
A2=AMAX-AMOD(AMAX, BANDW)+BANDW

...Calculate number of bands and zero each band

ICNBAND=(IFIX(A2)-IFIX(A1))/IBW+1
ICBAND(1)=IFIX(A1)
DO 40, I=2, ICNBAND
  ICBAND(I)=ICBAND(I-1)+IBW
  CAFREQ(I-1)=0.0
40 CONTINUE
c...Calculate frequency in each band
c
DO60, I=1, NRUNS
   DO50, J=1, (ICNBAND-1)
      IF(CAPTIM(I).GT.ICBAND(J).AND.CAPTIM(I).LE.ICBAND(J+1)) THEN
         CAFREQ(J)=CAFREQ(J)+1.0
         GO TO 60
      ENDIF
   50 CONTINUE
   60 CONTINUE

c...Initialise commulative frequency
c
DO70, I=1, (ICNBAND-1)
   CCFREQ(I)=0.0
70 CONTINUE

c...Calculate frequency (%)
c
DO80, I=1, (ICNBAND-1)
   CAFREQ(I)=CAFREQ(I)*100.0/FLOAT(NRUNS)
80 CONTINUE

c...Calculate cumulative frequency (%)
c
   CCFREQ(1)=CAFREQ(1)
   DO90, I=2, (ICNBAND-1)
      CCFREQ(I)=CCFREQ(I-1)+CAFREQ(I)
90 CONTINUE

c...Calculate bands midpoints
c
DO95, I=1, ICNBAND
   CABMPT(I)=FLOAT(ICBAND(I))+BANDW/2.0
95 CONTINUE

c...Calculate the Rate of Probability with CAPTIM (f(x))
c
   A=0.39894228/CAPSIG
   DO100, I=1, (ICNBAND-1)
      ROPWCAP(I)=(A*EXP(-(((CABMPT(I)-CAMEAN)**2)/(2.0*CAPSIG**2))))*1.00000*BANDW
100 CONTINUE

c...Exit
c
RETURN
END
Appendix F.20: Function IDAY

INTEGER FUNCTION IDAY(IDAYNO)

PARAMETER(A=365.25,B=30.6001,C=122.1)

IYEAR=INT((IDAYNO-C)/A)

IMONTH=INT((IDAYNO-INT(A*IYEAR))/B)

IDAY=IDAYNO-INT(A*IYEAR)-INT(B*IMONTH)

RETURN
END
Appendix G: One Simulation Run Output Written to the File PERTRA3.RES

PROJECT START DATE: 7/JUN/1993

PROJECT FINISH DATE: 12/APR/1994

Project Duration = 310 Days

CAPTIM = 5332846.00 £.DAY

Internal Rate of Return (IRR) = 562.22 %
Number of runs = 1000

<table>
<thead>
<tr>
<th>TASK NAME</th>
<th>TASK TYPE</th>
<th>OPTIMISTIC</th>
<th>MOST LIKELY</th>
<th>PESSIMISTIC</th>
<th>CRITICALITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFFWM</td>
<td>T</td>
<td>1.20</td>
<td>4.00</td>
<td>5.60</td>
<td>.00</td>
</tr>
<tr>
<td>AUSFWM</td>
<td>T</td>
<td>1.20</td>
<td>4.00</td>
<td>5.60</td>
<td>.00</td>
</tr>
<tr>
<td>CLEAR1</td>
<td>T</td>
<td>.60</td>
<td>2.00</td>
<td>2.80</td>
<td>51.90</td>
</tr>
<tr>
<td>CLEAR2</td>
<td>T</td>
<td>1.50</td>
<td>5.00</td>
<td>7.00</td>
<td>100.00</td>
</tr>
<tr>
<td>CPBPD14</td>
<td>T</td>
<td>.00</td>
<td>14.00</td>
<td>.00</td>
<td>.00</td>
</tr>
<tr>
<td>CPFBLD</td>
<td>T</td>
<td>.30</td>
<td>1.00</td>
<td>1.40</td>
<td>.30</td>
</tr>
<tr>
<td>CPFCONC</td>
<td>T</td>
<td>.30</td>
<td>1.00</td>
<td>1.40</td>
<td>.30</td>
</tr>
<tr>
<td>CPFEX</td>
<td>T</td>
<td>.60</td>
<td>2.00</td>
<td>2.80</td>
<td>.30</td>
</tr>
<tr>
<td>CPFRF</td>
<td>T</td>
<td>4.50</td>
<td>15.00</td>
<td>21.00</td>
<td>.30</td>
</tr>
<tr>
<td>CPSCAFF</td>
<td>H</td>
<td>.00</td>
<td>46.00</td>
<td>.00</td>
<td>.00</td>
</tr>
</tbody>
</table>

Note: Criticality index shows the percentage to the number of runs that the task was a critical task.
1. PROJECT DURATIONS

Minimum project duration = 305 DAYS
Maximum project duration = 354 DAYS
Mean Project duration = 332.70 DAYS
Median project duration = 332.00 DAYS
Standard Deviation = 8.85 DAYS
Skewness = .2367

<table>
<thead>
<tr>
<th>MIDPOINT</th>
<th>FREQUENCY</th>
<th>CUM. FREQUENCY</th>
<th>f(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>295.00</td>
<td>.00</td>
<td>.00</td>
<td>.01</td>
</tr>
<tr>
<td>305.00</td>
<td>1.10</td>
<td>1.10</td>
<td>.33</td>
</tr>
<tr>
<td>315.00</td>
<td>7.20</td>
<td>8.30</td>
<td>6.09</td>
</tr>
<tr>
<td>325.00</td>
<td>30.40</td>
<td>38.70</td>
<td>30.88</td>
</tr>
<tr>
<td>335.00</td>
<td>47.80</td>
<td>86.50</td>
<td>43.60</td>
</tr>
<tr>
<td>345.00</td>
<td>12.10</td>
<td>98.60</td>
<td>17.15</td>
</tr>
<tr>
<td>355.00</td>
<td>1.40</td>
<td>100.00</td>
<td>1.88</td>
</tr>
</tbody>
</table>

Median project duration is the duration lies in the middle of the set of project durations when arranged in order of magnitude.

f(x) is the rate of probability with project durations.
2. INTERNAL RATE OF RETURN (IRR)

Minimum IRR = 61.101 %
Maximum IRR = 1291.002 %
Mean IRR = 368.269 %
Median IRR = 289.961 %
IRR Standard Deviation = 236.04650 %
Skewness = .99525

<table>
<thead>
<tr>
<th>MIDPOINT</th>
<th>FREQUENCY</th>
<th>CUM. FREQUENCY</th>
<th>f(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50.00</td>
<td>6.10</td>
<td>6.10</td>
<td>6.81</td>
</tr>
<tr>
<td>150.00</td>
<td>25.20</td>
<td>31.30</td>
<td>11.02</td>
</tr>
<tr>
<td>250.00</td>
<td>20.40</td>
<td>51.70</td>
<td>14.91</td>
</tr>
<tr>
<td>350.00</td>
<td>5.20</td>
<td>56.90</td>
<td>16.85</td>
</tr>
<tr>
<td>450.00</td>
<td>19.60</td>
<td>76.50</td>
<td>15.92</td>
</tr>
<tr>
<td>550.00</td>
<td>11.20</td>
<td>87.70</td>
<td>12.57</td>
</tr>
<tr>
<td>650.00</td>
<td>3.60</td>
<td>91.30</td>
<td>8.29</td>
</tr>
<tr>
<td>750.00</td>
<td>3.50</td>
<td>94.80</td>
<td>4.57</td>
</tr>
<tr>
<td>850.00</td>
<td>2.70</td>
<td>97.50</td>
<td>2.11</td>
</tr>
<tr>
<td>950.00</td>
<td>.00</td>
<td>97.50</td>
<td>.81</td>
</tr>
<tr>
<td>1050.00</td>
<td>1.70</td>
<td>99.20</td>
<td>.26</td>
</tr>
<tr>
<td>1150.00</td>
<td>.00</td>
<td>99.20</td>
<td>.07</td>
</tr>
<tr>
<td>1250.00</td>
<td>.80</td>
<td>100.00</td>
<td>.02</td>
</tr>
</tbody>
</table>
3. Negative CAPTIM

Minimum CAPTIM = 325526.00 £.DAY
Maximum CAPTIM = 8720172.00 £.DAY
Mean CAPTIM = 5385784.00 £.DAY
Median CAPTIM = 5097613.00 £.DAY
CAPTIM Standard Deviation = 1212286.00 £.DAY
Skewness = .71313

<table>
<thead>
<tr>
<th>MIDPOINT</th>
<th>FREQUENCY</th>
<th>CUM. FREQUENCY</th>
<th>f(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500000.00</td>
<td>.00</td>
<td>.00</td>
<td>1.94</td>
</tr>
<tr>
<td>3500000.00</td>
<td>13.00</td>
<td>13.00</td>
<td>9.81</td>
</tr>
<tr>
<td>4500000.00</td>
<td>31.90</td>
<td>44.90</td>
<td>25.20</td>
</tr>
<tr>
<td>5500000.00</td>
<td>26.20</td>
<td>71.10</td>
<td>32.76</td>
</tr>
<tr>
<td>6500000.00</td>
<td>16.10</td>
<td>87.20</td>
<td>21.57</td>
</tr>
<tr>
<td>7500000.00</td>
<td>9.90</td>
<td>97.10</td>
<td>7.19</td>
</tr>
<tr>
<td>8500000.00</td>
<td>2.90</td>
<td>100.00</td>
<td>1.21</td>
</tr>
</tbody>
</table>