ERRORS

The Author regrets that the following omissions and errors have been found in this Thesis. Those marked * make significant corrections to the meaning.

p60 The heading on column 5 should be 'NUMBER' and 'LIST OF ITEMS' should start above column 6.

p65 The action code corresponding to 'Number of lives for an item > 100' should be 3 instead of 2.

*p82 Rule B2 should be:

'If any part of a lesson from requirement r is assigned to occur at period p then cancel PR(p,r) must be performed.'

*p86 After line 2 the following condition should be present:

'and iii) at each member the item may not have a free period.'

*p87 In line 8 for 'R' read '$R_n$'.

*p103 After Rule D3 the following comment should be present:

'If NR < r $\leq$ NTS then Rule D3 applies to a tied set of requirements instead of to a single requirement.'

p113 The penultimate line should be:

'(RIF[i] - RIF[i - 1]) ÷ NOLIVES[i] > 3 * NP ÷ 5'

p128 In line 9 for 'cnacelled' read 'cancelled'.

p137 In line 15 for 'to a decrease in' read 'by a decrease in'.

p170 Line 16 should be:

'$TNA[r_1,r_2] \leq Min\{N[r_1],N[r_2]\}.Min\{M[r_1],M[r_2]\}$'

p176 The last entry in this figure should be '250 7'.

p190 In line 21 for 'quatilies' read 'qualities'.

H.C.J.

# COMPUTER AIDED CONSTRUCTION OF SCHOOL TIME-TABLES

A

Thesis

presented on application for

admission to the degree

of

DOCTOR OF PHILOSOPHY

in the

Faculty of Science

of the

UNIVERSITY OF LONDON

by

Howard Clements Johnston

in

March 1970

## ABSTRACT

This thesis discusses an approach to the problem of making school time-tables with the aid of an electronic digital computer. The mathematical model defined is capable of describing almost all of the features required in the time-table for a normal British Secondary School.

Having defined the model a system for helping the school staff to present their data accurately in the required form and a computer program to check this data are discussed.

Methods are then described for representing the data and the additional information necessary during the construction of the time-table either inside or outside the computer.

The main program, which attempts to make the time-table, makes a sequence of assignments which continues either until a complete solution has been obtained or until the completion of a solution is no longer possible. In the latter case some of the previous assignments are removed and alternative choices made.

A qualitative evaluation of the main program is given which shows that in general the methods used are not capable of producing time-tables for real schools since for most real schools no solution of the type sought exists.

The conclusion is that the constraints must be made more flexible and that this can best be achieved by combining the powers of the computer, to perform operations quickly and accurately, with those of the experienced manual time-tabler, to make decisions based on a flexible and ill-defined set of constraints. This could most effectively be achieved by allowing the manual time-tabler to control the computer using an on-line terminal.

ACKNOWLEDGEMENTS

The work reported in this Thesis was carried out on a part-time basis at the University of London Institute of Computer Science while the Author was employed in the School of Mathematics Computing and Statistics in the City of Leicester Polytechnic.

Sincere thanks are due to:-

Professor R.A.Buckingham, for accepting the Author as a Part-Time Research Student in the Institute of Computer Science.

Mr. Keith Wolfenden, who has supervised this research, for his keen interest, valuable assistance and in particular for his friendship throughout the project.

The Governors and the Director of the City of Leicester Polytechnic, for granting permission to the Author to undertake this research.

Mr. M.Bridger and other members of Staff in the School of Mathematics, Computing and Statistics in the City of Leicester Polytechnic for arranging the Author's duties so that he was able to make regular weekly visits to London during the project.

The following people for their assistance in supplying data and other information: Mr. J.Tipping, of Royal Belfast Academical Institution; Messrs M.J.Roper and J.C.Clarke, of Guthlaxton Upper School; the Headmaster and Staff of Holland Park School and in particular Mr. Lewis Stedman-Jones; Mr.N.D.Thomson, of Gordonstoun School; Dr. J.Lions, formerly of J.Kates and Associates, Toronto; Dr. ir. J.S.Folkers, of Delft University of Technology,Delft; Mr. H.Michalsen, of Norges Tekniske Høgskole, Trondheim; and Dr. K.N.Groom,of the Local Government Operational Research Unit.

Other members of the Staff at the Institute of Computer Science and Atlas Computing Service, for their help in numerous ways and on numerous occasions.

Finally, the Author wishes to express particular gratitude to his wife, Jenny, who has shared the pleasures and the hardships of the last four years of this project, especially for her valuable assistance in typing and checking this Thesis.

CONTENTS


| | | Page |
|---|---|---|
| CHAPTER ONE - AN OUTLINE OF THE PROBLEM | | |
| 1.0 | Introduction | 8 |
| 1.1 | What we mean by a school | 8 |
| 1.2 | What we mean by a time-table | 9 |
| 1.3 | The process of producing a time-table | 9 |
| 1.4 | The advantage of using a computer | 10 |
| 1.5 | Difficulties in using a computer | 11 |
| 1.6 | Outline of the complete system | 12 |
| CHAPTER TWO - THE BASIC MODEL | | |
| 2.0 | Introduction | 14 |
| 2.1 | Definitions | 14 |
| 2.2 | The example problem | 22 |
| 2.3 | The finished time-table | 26 |
| CHAPTER THREE - DESCRIPTION OF A SCHOOL | | |
| 3.0 | Introduction | 32 |
| 3.1 | The Item List | 32 |
| 3.2 | The Time Structure List | 35 |
| 3.3 | The Requirement List | 37 |
| 3.4 | The Fixed Free Period List | 39 |
| 3.5 | The Forbidden Assignment List | 40 |
| 3.6 | The Pre-assignment List | 41 |
| 3.7 | Factors which the model excludes | 42 |
| 3.8 | Conclusion | 43 |
| CHAPTER FOUR - PREPARATION OF THE DATA DOCUMENTS | | |
| 4.0 | Introduction | 44 |
| 4.1 | Document Number One - The Item List | 44 |
| 4.2 | Document Number Two - The Time Structure List | 45 |
| 4.3 | Document Number Three - The Curriculum | 45 |

Page

4.4 Document Number Four - The Teacher Duty Chart
Document Number Five - The Room and Equipment Usage Chart
Document Number Six - The Requirement List 50
4.5 Document Number Seven - The Fixed Free Period List
Document Number Eight - The Forbidden Assignment List
Document Number Nine - The Pre-assignment List 54
4.6 Final Assembly 54

CHAPTER FIVE - CHECKING THE DATA

5.0 Introduction 55
5.1 The Data Checking Routine 55
5.2 Manual Checks 57

CHAPTER SIX - STORAGE AND DISPLAY OF INFORMATION

6.0 Introduction 70
6.1 Single variables 70
6.2 Arrays associated with days and periods 70
6.3 Arrays associated with items 71
6.4 Arrays associated with requirements 72
6.5 Arrays associated with items and requirements 75
6.6 Item bands 75
6.7 Elements in TPA 77

CHAPTER SEVEN - CANCELLATION RULES FOR ARRAY PR

7.0 Introduction 80
7.1 The operation cancel PR(p,r) 80
7.2 To cancel or not to cancel 81
7.3 Some fairly obvious rules 82
7.4 Some less obvious rules 84
7.5 Generalisation of Rules B7, B8 and B9 93
7.6 The perfect set of rules 93

CHAPTER EIGHT - CANCELLATION RULES FOR ARRAY IFM

8.0 Introduction 96
8.1 The operation cancel IFM(p,i) 96
8.2 When to use cancel IFM(p,i) 96

Page

CHAPTER NINE - IMPOSSIBLE, FORCING AND TIGHT SITUATIONS

9.0 Introduction 101
9.1 Definitions 101
9.2 Rules for locating special situations 101

CHAPTER TEN - THE MAIN PROGRAM

10.0 Introduction 108
10.1 Diagrammatic representation of the problem 109
10.2 Chosen assignment strategy 112
10.3 The priority parameter 112
10.4 The tightest element 116
10.5 The lists 118
10.6 Additional arrays 120
10.7 Implementation of the various sets of rules 121
10.8 Retreating 126
10.9 The overall structure 129
10.10 Input data 129
10.11 Output 129
10.12 Comment 130

CHAPTER ELEVEN - EVOLUTION AND TESTING OF THE MAIN PROGRAM

11.0 Introduction 136
11.1 Parallel testing and development 136
11.2 Roots in the literature 137
11.3 The most persistent problem 141
11.4 Other main problem areas 147

CHAPTER TWELVE - THE TEST DATA USED

12.0 Introduction 151
12.1 R.B.A.I. - a Grammar School in Belfast 152
12.2 G.U.S. - an Upper School in Leicestershire 153
12.3 H.C.T. 1965-66 and 1966-67 - a small Canadian School 154
12.4 H.P.S. - a Comprehensive School in London 154
12.5 G.P.S. - a Public School in Scotland 159
12.6 Conclusions 160

Page

CHAPTER THIRTEEN - AN ALTERNATIVE APPROACH

13.0 Introduction 161
13.1 Definitions 162
13.2 Acceptable arrangements 163
13.3 Arrangement counts 164
13.4 Illustrations using the example problem 165
13.5 An arrangement generator program 168
13.6 A group generator program 169
13.7 A modified group generatod program 170
13.8 Conclusion 172

CHAPTER FOURTEEN - ANALYSIS AND SUGGESTIONS

14.0 Introduction 179
14.1 A basic fault in the approach 180
14.2 A basic change in the approach 180
14.3 Relaxation of constraints in other programs 181
14.4 Disadvantages of the methods for relaxing constraints 186
14.5 The need for modifications to completed time-tables 187
14.6 Suggestions for a revised approach 187
14.7 A final comment 190

APPENDIX 192

BIBLIOGRAPHY 199

# CHAPTER ONE

## AN OUTLINE OF THE PROBLEM

### 1.0 Introduction

The basic aim of the project discussed in this thesis has been to produce a system for aiding the construction of school time-tables. By 'system' we mean a sequence of operations, some manual and some performed by an electronic digital computer, which cover the complete time-table construction process. Before we can state what this complete process includes we must define what we mean by 'a school' and 'a time-table'.

### 1.1 What we mean by a school

By 'a school' we mean a normal British Secondary School or any educational establishment with a similar structure. It will have the following properties:-

i) It consists of:-
   a) groups of pupils, to be called classes;
   b) teachers;
   c) rooms, some of which are supplied with special equipment;
   and d) equipment which may be moved from one room to another.

ii) It is open for a certain length of time on each day of its working week, normally 6 to 7 hours on Monday to Friday and in some cases 3 to 4 hours on Saturday.

iii) The week is divided up into about 35 to 45 approximately equal periods each about 35 to 45 minutes in length and each lesson lasts for an integral number of periods, rarely more than 3.

iv) Each class is normally in attendance throughout the time when the school is open, although there may be parts of each day when it is not being taught by a teacher.

v) The majority of the teachers work full time at the school and are actually teaching for about 70% to 90% of the working week but teachers with special responsibilities have a lighter teaching load.

vi) Each room, especially those supplied with special equipment is normally in use for 80% to 100% of the working week.

## 1.2 What we mean by a time-table

In a school the activities of each class and each teacher and the usage of each room and piece of special equipment normally follows a fixed pattern which repeats after a certain period of time, usually one week. By 'a time-table' we mean the document which defines this pattern i.e. it states what each class and each teacher is to do and to what use each room and piece of equipment is to be put at each period of the week.

## 1.3 The process of producing a time-table

The process of producing a time-table has several main phases. They are:-

Phase 1 - Production of the Curriculum i.e. deciding the course of study which will be available to each pupil or group of pupils.

Phase 2 - Assigning a teacher and a room to each lesson.

Phase 3 - Determining when each lesson is to take place.

### 1.3.1 Production of the curriculum

Phase 1 can only be done by a senior member of the school staff. It has to take account of the examinations for which the various groups of pupils are preparing, any regulations which must be observed e.g. about the provision of religious instruction, and is influenced by the teaching staff and the facilities which are currently available e.g. the total amount of Physics which can be taught in a school is controlled by the number of suitable staff and the number of Physics laboratories. The end product is called the Curriculum and normally it changes little from year to year. Although it may be possible to employ a computer to optimize the Curriculum in some way this Phase is not seen, at the moment, as a major problem and we have not attempted to interfere in any way with the current methods of doing it. Our system does, however, include a method of recording the Curriculum in an easily understandable form.

### 1.3.2 Assigning a teacher and a room to each lesson

Most schools are organised on a Departmental basis where each Department teaches a particular subject or related group of

subjects e.g. a Modern Languages Department might teach French, German, Spanish, Italian and Russian. Phase 2 is usually carried out by the Heads of the Departments. The Head of a Department is the best person to decide which of his teachers is most suitable for each lesson e.g. a Mathematics teacher may be excellent at teaching G.C.E. 'A' level Mathematics to the most senior pupils but be almost incapable of explaining elementary Algebra to 11 year-olds. The Head also knows what rooms and equipment are available and can assign them. Obviously in doing this he must make sure that each teacher has his correct teaching load and that no room or piece of equipment is used for more than 100% of the week. In this Phase a computer could definitely be used to optimize the operation but the amount of work involved out-weighs the advantage that could be obtained. Some quantitative method would have to be devised for measuring each teacher's ability to take a particular lesson e.g. the Mathematics teacher mentioned above might be assigned weight 10 for teaching G.C.E. 'A' level Mathematics but only weight 2 for teaching elementary Algebra. Similar weighting would measure the suitability of each room type and the problem would be to assign a teacher and room type for every lesson so that the total of the weights over the whole Department was a maximum. The Head of Department would still have to decide the weights and this would be a much longer and more difficult job than making the actual decisions. Thus our system does not involve the computer in this Phase but it does include methods of documentation which aid the Head of Department in his task.

1.3.3 Determining when each lesson is to take place

Phase 3 really is the head-ache! When performed manually, usually by one unfortunate member of the school staff, it takes weeks of very careful work and one tiny mistake can cause hours of wasted effort. It is not unusual for this Phase to be entered 4 or 5 months before the time-table is needed and few time-tables are perfect even by the time thay have to go into use. This is the Phase where the computer can come to the rescue.

1.4 The advantage of using a computer

There are several reasons for using the computer in Phase

3 of the process. First of all the nature of the task, namely a long sequence of operations which have to be performed with 100% accuracy, is exactly the sort of job a computer is good at. Secondly, because the computer should be able to produce the result in minutes or hours instead of months, the whole operation could be started much nearer to the date when the time-table is required. This would be a great advantage because if a time-table required for September has to be started in April much of the data is not available. Part of the trouble is caused by staff changes. A teacher can normally give in his notice at the end of May and the replacement teacher may not be appointed until July or even August thus when the time-table is first made the school is usually forced to make the assumption that the new teacher will have exactly the same abilities as the old one. Often this cannot be achieved and hurried changes to the time-table have to be made. Another problem is that the choice of subjects made by senior pupils is often governed by their G.C.E. examination results which are not published until mid-August thus a time-table which allows one group of 16 pupils to study Advanced level G.C.E.Chemistry can become useless overnight if 25 pupils decide they want to take this subject and the laboratory cannot hold that number. Again, last minute changes have to be made. If the time-table could be run off the day before the term was due to start these would no longer be problems.

1.5 Difficulties in using a computer

There are also difficulties in using the computer. The people who construct time-tables manually depend to a large extent on experience, often built up over many years, and there are parts of this experience, peculiar to their own schools, which cannot be built into the program or included as part of the data. Also, because the computer program defines a rigid procedure for constructing a time-table subject to certain built in constraints, it is difficult to give it the freedom to overlook the constraints just occasionally as the manual time-tabler invariably does e.g.if the program is such that a double period lesson should not be split by a lunch break then it will never be split by the computer but if he can see no alternative the human will probably split it rather than undo a lot of the work

he has already done. This inflexibility means that there are cases where the computer will say "there is no solution" while the teacher doing the job will break the rules and press on regardless of the consequences.

A further disadvantage is that a member of the school staff has usually a considerable amount of knowledge about the school and will probably notice any errors in the information given to him for constructing the time-table. There are some types of mistakes which the computer cannot detect e.g. a games afternoon might appear on the Curriculum as 1 lesson 3 periods long but the data passed to the time-tabler might say 3 lessons each 1 period long. The human will probably spot this at once while the machine will proceed to time-table 3 uselessly short games lessons. Again in a school where A.Jones is a Mathematics master and B.Jones is a Needlework mistress a computer would quite happily accept a request for B.Jones to teach Differential Calculus to the Fourth Form while the human will make a correction at once.

The advantages in using a computer, however, easily outweigh the disadvantages thus the main effort of this project has been to produce an efficient computer program for performing Phase 3 of the process.

This program aims to produce a time-table for each set of data which it is given. It does not try to produce the best time-table. The reason for this is simply that it is very difficult to measure the quality of a time-table. The only standard which can really be applied is that it should be as good, from the school's point of view, as any time-table which they could make for themselves by manual methods. In order to make sure, however, that the time-table produced has at least some of the properties which a school would expect several constraints which control the quality of the time-table are built into the model.

## 1.6 Outline of the complete system

The complete system has been christened SCATTAR which is derived from the phrase "School, Class and Teacher Time-table Assembly Routine". It consists of the following parts:-

i) Manual procedures for recording the curriculum to be used

by the senior member of the school staff who carries out Phase 1.

ii) Manual procedures, to be used by the Heads of Department in Phase 2, to help them to avoid making mistakes and for recording their decisions.

iii) A computer program, called the Data Checking Routine, which accepts some of the documents produced in ii) and produces listings which enable all the recorded information to be checked.

iv) A computer program, called the Main Program, which accepts the same data as the Data Checking Routine and attempts to produce a complete time-table.

In the chapters which follow we describe the complete system, discuss its evolution and the problems which have been used to test it and conclude by making suggestions for further improvements.

# CHAPTER TWO

## THE BASIC MODEL

### 2.0 Introduction

Let us now forget about schools and think in much more general terms. We wish to consider an organisation, consisting of people and things, which exists so that a certain set of activities can take place. The set of activities taking place at any instant in time will in general be different from those taking place at any other instant and management of the organisation involves making a schedule or time-table for some known period of time. This time-table may be used once or repeatedly. In defining a model we are providing a method of describing those aspects of the organisation and its activities which are relevant to the preparation of its time-table.

### 2.1 Definitions

To describe the people and things which make up the organisation it is, for most purposes, not necessary to distinguish between a person, a group of persons, a thing, a group of things or a combination of persons and things.

#### 2.1.1 A unit

We define a unit to be a fixed combination of m persons and n things where m and n are positive integers or zero but m and n are not both zero.

Since a unit is a fixed combination it remains together throughout the activities of the organisation.

Example

If a vehicle and its driver are defined as a unit in a transport organisation it implies that the vehicle is always driven by the same driver and the driver always drives the same vehicle.

#### 2.1.2 Equal units

We define two units to be equal if, for all relevant activities, each can perform exactly the same part as

the other.

2.1.3 An item

We define an item to be a type of unit.

2.1.4 Number of lives

We define the number of lives possessed by an item to be the number of units of that item present in the organisation.

The use of the word 'lives' comes from the idea than an item with v lives has the ability to do v jobs simultaneously. All the units of a given item must be equal to one another.

2.1.5 Type code

We define the type code for a given item to be a small positive integer which identifies it as a member of one of several basic groups of items in the organisation.

In a school we have used 1 to denote a class, 2 to denote a teacher and 3 to denote a room or piece of special equipment. This type code is only used by the Main Program (Chapter 10). It is needed as an aid to the control of the number of units which, at a given period in the time-table, are not involved in any activity within the organisation. The Author feels that it should be possible to control this factor without the need for a type code but has been unable to find a suitable technique. These matters are also discussed in 7.4.4, 10.7.3 and 11.3.

2.1.6 The Item List

We define the Item List to be the following set of data which completely describes the composition of the organisation:-

| | | |
|---|---|---|
| NI | - | the total number of items; |
| $(EIN_i, NOLIVES_i, TYPE_i,$ $i = 1,NI)$ | - | the name, number of lives and type code for each item. |

For obvious reasons, no two items may have the same name.

2.1.7 An internal activity

We define an internal activity to be an activity of the organisation which is to appear on the time-table.

2.1.8 An external activity

We define an external activity to be an activity, inside or outside the organisation, in which parts of the organisation are to take part but which is not to appear on the time-table.

2.1.9 A period

We define a period to be the largest unit of time such that the duration of each internal activity may be expressed as an integral number of periods.

2.1.10 A day

We define a day to be a consecutive set of periods bounded by natural breaks in the total activity of the organisation such that no activity may start on one day and be completed on the next.

2.1.11 A week

We define a week to be the set of days over which the time-table is required.

The terms day and week need not be related in any way to the periods of time normally described by these words.

To describe an activity we only need to consider two factors:-

i) the length of time it lasts

and ii) the set of people and things which must be available for it to take place.

A particular activity is therefore completely described by:-

$$M_r, (EIN_{ri}, i = 1,n_r)$$

where $M_r$ periods is the length of the activity and the $EIN_{ri}$ are item names for the $n_r$ units involved. The inclusion of a given item name t times implies that t units of that item are required to be available.

2.1.12 An m-period activity

We define an m-period activity to be an activity which

lasts for m consecutive periods.

2.1.13 The longest activity parameter

We define the longest activity parameter to be the length in periods of the longest internal activity.

2.1.14 Equal activities

We define two activities to be equal if they last for the same length of time and require the same set of units to be available.

2.1.15 The space parameter

We define the space parameter to be the minimum number of periods occupied by other activities which must occur between any two equal activities on the same day.

2.1.16 A normal distribution

Let n and d be integers, m be the integral part of $\frac{n}{d}$ and $k = n - md$ then we define a normal distribution of n activities over d days to be a distribution which has m activities on any d - k days and m + 1 activities on the remaining k days subject to a given space parameter.

Example

A normal distribution of 4 activities over 5 days gives m = 0 and k = 4 i.e. one activity on any 4 of the 5 days.

Example

A normal distribution of 5 activities over 5 days gives m = 1 and k = 0 i.e. exactly one activity on each day.

Example

A normal distribution of 8 activities over 5 days gives m = 1 and k = 3 i.e. one activity on any 2 of the days and 2 activities on each of the remaining 3 days. If the space parameter is 2 any day which has 2 of the activities must have at least 2 periods of another activity between the end of the first activity and the start of the second.

Apart from the total breaks in the activity of the organisation between days, there may be other breaks within a day across which some activities may be allowed to continue while others

are not.

2.1.17 An m-period pattern

We define an m-period pattern to be a set of digits, equal in number to the number of periods on a given day, such that the $p^{th}$ digit is 1 if an m-period activity may start at the $p^{th}$ period of the day and is 0 otherwise.

2.1.18 The Time Structure List

We define the Time Structure List to be the following set of data which completely describes the time structure of the organisation:-

| | |
|---|---|
| ND | - the number of days per week; |
| $(EDN_d, DLENGTH_d, d = 1,ND)$ | - the name and the number of periods for each day; |
| $(EPN_p, p = 1,NP)$ | - the name of each period where $NP = \sum_{d=1}^{ND} DLENGTH_d$; |
| LAP | - the longest activity parameter; |
| SP | - the space parameter; |
| $((PP_{x,p}, p = 1,NP), x = 1,LAP)$ | - the period patterns. |

For obvious reasons no 2 days may have the same name nor may any two periods have the same name.

In order to write down as concisely as possible all the activities which are to appear on the time-table we may group sets of equal activities together.

2.1.19 A requirement

We define a requirement to be a statement which describes one group of equal activities which are to appear in the time-table with a normal distribution.

2.1.20 A tied set of requirements

We define a tied set of requirements to be a set of requirements which describe a group of activities (not all equal) which are to appear in the time-table with a normal distribution.

One method of recording a requirement is:-

$$ERN_r,\ TIE_r,\ M_r,\ N_r,\ (EIN_{ri},\ i = 1,n_r);$$

where $ERN_r$ is a name to identify the requirement,

$TIE_r$ is the tie name (see below),

$M_r$ is the length in periods of the activity involved,

$N_r$ is the number of times per week that the activity is required to occur

and $EIN_{ri}$, $i = 1,n_r$ are the names of the $n_r$ units involved.

For a single requirement the tie name is set equal to zero (or equal to the requirement name).

To tie a set of requirements together the tie name of each requirement is put equal to the name of the next requirement of the set where the requirement next after the last one is defined to be the first one.

Example

SC1, 0, 2, 3, CLASS1, SMITH, SCLAB;

This means "the requirement named SC1, which is not tied to any other requirement, is that one unit from each of the items named CLASS1, SMITH and SCLAB is required to be available for a 2-period activity 3 times per week".

Example

ENG6, 0, 1, 5, 6A, 6B, 6C, LIB, LIB, LIB;

This means "the requirement named ENG6, which is not tied to any other requirement, is that one unit from each of the items named 6A, 6B and 6C and 3 units from the item named LIB are required to be available for a 1-period activity 5 times per week".

Example

GER30, FR30, 1, 3, 3L, HANS;

FR30, FR35, 2, 1, 3L, PIERRE;

FR35, GER30,1, 1, 3L, PIERRE;

This means "the 3 requirements named GER30 (which requires one unit from 3L and HANS for 3 1-period activities), FR30 (which requires one unit from 3L and PIERRE for 1 2-period activity) and FR35 (which requires 3L and PIERRE for a 1-period activity) form a tied set i.e. the total of 5 activities involved are to be distributed as if they all belonged to the same requirement".

The pair of tied requirements

R1, R2, 1, 1, X, Y;

R2, R1, 1, 1, X, Y;

would be perfectly legal but are exactly equivalent to

R, 0, 1, 2, X, Y;

i.e. the activities involved in a set of tied requirements do not have to be unequal but if some of them are equal then the number of requirements in the set can be reduced.

2.1.21 The Requirement List

We define the Requirement List to be the following set of data which completely describes the activities which are to appear on the time-table:-

NR - the number of requirements;

$(ERN_r, TIE_r, M_r, N_r, (EIN_{ri}, i = 1,n_r), r = 1,NR)$; - the requirements.

For obvious reasons, no 2 requirements may have the same name.

2.1.22 An assigned activity

We define an activity to have been assigned to a given period if a decision has been taken which means that part or all of the activity will occur at that period in the time-table.

2.1.23 An employed unit

We define a unit to be employed at a given period if a decision has been taken which determines the internal or external activity in which the unit will take part at that period.

2.1.24 A free unit

We define a unit to be free at a given period if it is employed on an external activity at that period.

Due to the presence of part-time workers or the sharing of equipment with other organisations there may be periods when certain units cannot be available. This information is known before the time-table is made.

2.1.25 A fixed free period request

We define a fixed free period request to be a statement

which lists those periods when one or more units of an item are pre-assigned to take part in external activities and are therefore not available for internal activities.

One method of recording a fixed free period request is:-

$EIN_i$, ($EPN_{ip}$, p = 1,$m_i$);

where $EIN_i$ is the item name and the $EPN_{ip}$ are period names. The inclusion of a given period name t times implies that t units of the item are required to be free.

Example

CHLAB, M6, M7, M8, F2, F2;

means "at periods M6, M7 and M8 one unit from item CHLAB is not available and at period F2 two units from the same item are not available".

2.1.26 The Fixed Free Period List

We define the Fixed Free Period List to be the complete set of fixed free period requests.

Due to the nature of some of the activities it may be necessary to restrict the periods at which they can take place.

2.1.27 A forbidden assignment request

We define a forbidden assignment request to be a statement which lists those periods when a particular activity may not take place.

One method of recording a forbidden assignment request is:-

$ERN_r$, ($EPN_{rp}$, p = 1,$p_r$);

where $ERN_r$ is the requirement name and $EPN_{rp}$ are period names.

Example

SCI4, M1, M2, T6, T7, F1, F8;

where the activities in requirement SCI4 are 3-period activities forbids any one of the three periods of any activity from this requirement to occur at periods M1, M2, T6, T7, F1 or F8.

2.1.28 The Forbidden Assignment List

We define the Forbidden Assignment List to be the complete set of forbidden assignment requests.

There may also be some activities which must take place at known times.

2.1.29 A pre-assignment request

We define a pre-assignment request to be a statement which gives the period when a particular activity must commence.

A pre-assignment request may be recorded by:-

$ERN_r$, $EPN_p$;

where $ERN_r$ is the name of the requirement to which the activity belongs and $EPN_p$ is a period name.

Example

MAT1, W6;

where MAT1 is a requirement for 4 2-period activities, requests that one of these activities must start at period W6.

2.1.30 The Pre-assignment List

We define the Pre-assignment List to be the complete set of pre-assignment requests.

2.1.31 A Standard Data Deck

We define a Standard Data Deck to be the following set of documents which completely describe an organisation and its activities:-

| | |
|---|---|
| The Item List | - (2.1. 6) |
| The Time Structure List | - (2.1.18) |
| The Requirement List | - (2.1.21) |
| The Fixed Free Period List | - (2.1.26) |
| The Forbidden Assignment List | - (2.1.28) |
| The Pre-assignment List | - (2.1.30) |

## 2.2 The Example Problem

Two versions of the Standard Data Deck for a small organisation are shown in figures 2.1 and 2.2. The problem defined by these sets of data will be called the Example Problem and will be used in other chapters to provide illustrations. The data set in figure 2.1 uses alphanumeric names for items, days, periods and requirements while the set in figure 2.2 uses only numeric names. Both sets are in a form which, if punched on a suitable input medium, could be accepted by a computer.

In the alphanumeric set the character ',' is used as a separator and where it is necessary to show that a record, e.g. a requirement, is complete the ',' is replaced by the terminator

Figure 2.1

```
13,
CLASS1, 1, 1,
CLASS2, 1, 1,
CLASS3, 1, 1,
CLASS4, 1, 1,
SMITH,  1, 2,
JONES,  1, 2,
DAVIS,  1, 2,
BEGGS,  1, 2,
WISE,   1, 2,
SCILAB, 2, 3,
LANGLB, 1, 3,
GYM,    1, 3,
TAPERC, 1, 3,
END;

5,
MON,    5,
TUE,    5,
WED,    5,
THU,    5,
FRI,    5,
M1,  M2,  M3,  M4,  M5,
TU1, TU2, TU3, TU4, TU5,
W1,  W2,  W3,  W4,  W5,
TH1, TH2, TH3, TH4, TH5,
F1,  F2,  F3,  F4,  F5,
2,    0,
1, 1, 1, 1, 1,
1, 1, 1, 1, 1,
1, 1, 1, 1, 1,
1, 1, 1, 1, 1,
1, 1, 1, 1, 1,
1, 1, 0, 1, 0,
1, 1, 0, 1, 0,
1, 1, 0, 1, 0,
1, 1, 0, 1, 0,
1, 1, 0, 1, 0,
END;
```

Figure 2.2

```
13
10   1   1
20   1   1
30   1   1
40   1   1
50   1   2
60   1   2
70   1   2
80   1   2
90   1   2
100  2   3
110  1   3
120  1   3
130  1   3
9999

5
1   5
2   5
3   5
4   5
5   5
11  12  13  14  15
21  22  23  24  25
31  32  33  34  35
41  42  43  44  45
51  52  53  54  55
2   0
1   1   1   1   1
1   1   1   1   1
1   1   1   1   1
1   1   1   1   1
1   1   1   1   1
1   1   0   1   0
1   1   0   1   0
1   1   0   1   0
1   1   0   1   0
1   1   0   1   0
9999
```

Figure 2.1 continued

```
30,
MAT1,     0,    1,    5,   CLASS1,   CLASS2,   SMITH,    JONES;
ENG1,     0,    1,    5,   CLASS1,   BEGGS;
FR1,      0,    1,    5,   CLASS1,   WISE,     LANGLB;
SC11,   SC12,   2,    2,   CLASS1,   CLASS2,   SMITH,    DAVIS,
                           SCILAB,   SCILAB;
SC12,   SC11,   1,    2,   CLASS1,   CLASS2,   SMITH,    DAVIS,
                           SCILAB,   SCILAB;
ART1,     0,    2,    1,   CLASS1,   DAVIS;
PE1,    MUS1,   1,    1,   CLASS1,   WISE,     GYM;
MUS1,    PE1,   1,    1,   CLASS1,   DAVIS,    TAPERC;
ENG2,     0,    1,    5,   CLASS2,   BEGGS;
FR2,      0,    1,    5,   CLASS2,   WISE,     LANGLB;
ART2,     0,    2,    1,   CLASS2,   DAVIS;
PE2,    MUS2,   1,    1,   CLASS2,   DAVIS,    GYM;
MUS2,    PE2,   1,    1,   CLASS2,   DAVIS,    TAPERC;
MAT3,     0,    1,    5,   CLASS3,   SMITH;
ENG3,     0,    1,    5,   CLASS3,   BEGGS;
FG3,      0,    1,    3,   CLASS3,   JONES,    WISE,     LANGLB;
PH31,   PH32,   2,    2,   CLASS3,   DAVIS,    SCILAB;
PH32,   PH31,   1,    1,   CLASS3,   DAVIS,    SCILAB;
CH31,   CH32,   2,    2,   CLASS3,   SMITH,    SCILAB;
CH32,   CH31,   1,    1,   CLASS3,   SMITH,    SCILAB;
PE3,      0,    1,    1,   CLASS3,   WISE,     GYM;
AM3,      0,    1,    1,   CLASS3,   CLASS4,   JONES,    DAVIS;
MAT4,     0,    1,    5,   CLASS4,   JONES;
ENG4,     0,    1,    5,   CLASS4,   BEGGS;
FR41,   FR42,   2,    1,   CLASS4,   WISE,     LANGLB;
FR42,   FR41,   1,    3,   CLASS4,   WISE,     TAPERC;
GE41,   GE42,   2,    1,   CLASS4,   JONES,    LANGLB;
GE42,   GE41,   1,    3,   CLASS4,   JONES,    TAPERC;
HIS4,     0,    1,    3,   CLASS4,   BEGGS;
PE4,      0,    1,    1,   CLASS4,   DAVIS,    GYM;
END;
```

Figure 2.2 continued

```
 30
 10    0   1   5   10   20   50   60    0
 20    0   1   5   10   80    0
 30    0   1   5   10   90  110    0
 40   45   2   2   10   20   50   70  100  100    0
 45   40   1   2   10   20   50   70  100  100    0
 50    0   2   1   10   70    0
 60   70   1   1   10   90  120    0
 70   60   1   1   10   70  130    0
 80    0   1   5   20   80    0
 90    0   1   5   20   90  110    0
100    0   2   1   20   70    0
110  120   1   1   20   70  120    0
120  110   1   1   20   70  130    0
130    0   1   5   30   50    0
140    0   1   5   30   80    0
150    0   1   3   30   60   90  110    0
160  165   2   2   30   70  100    0
165  160   1   1   30   70  100    0
170  175   2   2   30   50  100    0
175  170   1   1   30   50  100    0
180    0   1   1   30   90  120    0
190    0   1   1   30   40   60   70    0
200    0   1   5   40   60    0
210    0   1   5   40   80    0
220  225   2   1   40   90  110    0
225  220   1   3   40   90  130    0
230  235   2   1   40   60  110    0
235  230   1   3   40   60  130    0
240    0   1   3   40   80    0
250    0   1   1   40   70  120    0
9999
```

Figure 2.1 continued

```
SCILAB,   M1, TU5, TU5;
JONES,    W4, TH5;
END;

ART2,     TU1, TU2;
END;

FG3,      M5;
MAT4,     M4;
END;
```

Figure 2.2 continued

```
100   11   25   25   0
 60   34   45    0
9999

100   21   22    0
9999

150   15
200   14
9999
```

';'. Each document is terminated with 'END;'.

In the numeric set 2 or more spaces are used as a separator and when it is necessary to terminate a record a '0' is used. The document terminator is '9999'.

The programs used to develop the system have been written in ALGOL for the University of London ICT Atlas computer. In Atlas ALGOL data in the numeric form is the more convenient but using Library subroutines for input and output of alphanumeric characters the alphanumeric data can be made acceptable.

## 2.3 The finished time-table

The time-table for an organisation which has been described by a Standard Data Deck may be expressed most concisely as an irregular array which has one row for each period in the week. The elements in the row for a given period are the requirement names of those lessons which commence at that period.

A solution for the example problem is shown in figures 2.3 and 2.4.

The time-table expressed in this form, combined with the Item List, the Time Structure List and the Requirement List provide all the necessary information to present the time-table in any other form. One possible alternative form consists of a rectangular array with one column for each unit and one row for each period. Element (p,u) contains the requirement name for the lesson on which unit u is to be employed at period p or the word FREE. The solution given in figures 2.3 and 2.4 is expressed in this expanded form in figures 2.5 and 2.6.

Figure 2.3

| Period | Lessons which commence | | | |
|---|---|---|---|---|
| M1 | MAT1 | PH32 | HIS4 | |
| M2 | ENG1 | FR2 | MAT3 | PE4 |
| M3 | FR1 | ENG2 | AM3 | |
| M4 | SC11 | ENG3 | MAT4 | |
| M5 | FG3 | ENG4 | | |
| TU1 | ART1 | FR2 | CH31 | ENG4 |
| TU2 | ENG2 | FR41 | | |
| TU3 | MAT1 | ENG3 | | |
| TU4 | ENG1 | ART2 | PE3 | MAT4 |
| TU5 | FR1 | MAT3 | GE42 | |
| W1 | FR1 | ENG2 | MAT3 | MAT4 |
| W2 | ENG1 | FR2 | PH31 | GE42 |
| W3 | MAT1 | HIS4 | | |
| W4 | SC11 | ENG3 | FR42 | |
| W5 | FG3 | ENG4 | | |
| TH1 | PE1 | ENG2 | MAT3 | GE42 |
| TH2 | MAT1 | ENG3 | FR42 | |
| TH3 | SC12 | FG3 | HIS4 | |
| TH4 | ENG1 | FR2 | CH31 | MAT4 |
| TH5 | FR1 | MUS2 | ENG4 | |
| F1 | MAT1 | PH31 | FR42 | |
| F2 | FR1 | ENG2 | MAT4 | |
| F3 | MUS1 | FR2 | MAT3 | ENG4 |
| F4 | SC12 | ENG3 | GE41 | |
| F5 | ENG1 | PE2 | CH32 | |

Figure 2.4

| Period | Lessons which commence | | | |
|---|---|---|---|---|
| 11 | 10 | 165 | 240 | |
| 12 | 20 | 90 | 130 | 250 |
| 13 | 30 | 80 | 190 | |
| 14 | 40 | 140 | 200 | |
| 15 | 150 | 210 | | |
| 21 | 50 | 90 | 170 | 210 |
| 22 | 80 | 220 | | |
| 23 | 10 | 140 | | |
| 24 | 20 | 100 | 180 | 200 |
| 25 | 30 | 130 | 235 | |
| 31 | 30 | 80 | 130 | 200 |
| 32 | 20 | 90 | 160 | 235 |
| 33 | 10 | 240 | | |
| 34 | 40 | 140 | 225 | |
| 35 | 150 | 210 | | |
| 41 | 60 | 80 | 130 | 235 |
| 42 | 10 | 140 | 225 | |
| 43 | 45 | 150 | 240 | |
| 44 | 20 | 90 | 170 | 200 |
| 45 | 30 | 120 | 210 | |
| 51 | 10 | 160 | 225 | |
| 52 | 30 | 80 | 200 | |
| 53 | 70 | 90 | 130 | 210 |
| 54 | 45 | 140 | 230 | |
| 55 | 20 | 110 | 175 | |

Figure 2.5

| Period | CLASS1 | CLASS2 | CLASS3 | CLASS4 | SMITH | JONES | DAVIS |
|---|---|---|---|---|---|---|---|
| M1 | MAT1 | MAT1 | PH32 | HIS4 | MAT1 | MAT1 | PH32 |
| M2 | ENG1 | FR2 | MAT3 | PE4 | MAT3 | FREE | PE4 |
| M3 | FR1 | ENG2 | AM3 | AM3 | FREE | AM3 | AM3 |
| M4 | SC11 | SC11 | ENG3 | MAT4 | SC11 | MAT4 | SC11 |
| M5 | SC11 | SC11 | FG3 | ENG4 | SC11 | FG3 | SC11 |
| TU1 | ART1 | FR2 | CH31 | ENG4 | CH31 | FREE | ART1 |
| TU2 | ART1 | ENG2 | CH31 | FR41 | CH31 | FREE | ART1 |
| TU3 | MAT1 | MAT1 | ENG3 | FR41 | MAT1 | MAT1 | FREE |
| TU4 | ENG1 | ART2 | PE3 | MAT4 | FREE | MAT4 | ART2 |
| TU5 | FR1 | ART2 | MAT3 | GE42 | MAT3 | GE42 | ART2 |
| W1 | FR1 | ENG2 | MAT3 | MAT4 | MAT3 | MAT4 | FREE |
| W2 | ENG1 | FR2 | PH31 | GE42 | FREE | GE42 | PH31 |
| W3 | MAT1 | MAT1 | PH31 | HIS4 | MAT1 | MAT1 | PH31 |
| W4 | SC11 | SC11 | ENG3 | FR42 | SC11 | FREE | SC11 |
| W5 | SC11 | SC11 | FG3 | ENG4 | SC11 | FG3 | SC11 |
| TH1 | PE1 | ENG2 | MAT3 | GE42 | MAT3 | GE42 | FREE |
| TH2 | MAT1 | MAT1 | ENG3 | FR42 | MAT1 | MAT1 | FREE |
| TH3 | SC12 | SC12 | FG3 | HIS4 | SC12 | FG3 | SC12 |
| TH4 | ENG1 | FR2 | CH31 | MAT4 | CH31 | MAT4 | FREE |
| TH5 | FR1 | MUS2 | CH31 | ENG4 | CH31 | FREE | MUS2 |
| F1 | MAT1 | MAT1 | PH31 | FR42 | MAT1 | MAT1 | PH31 |
| F2 | FR1 | ENG2 | PH31 | MAT4 | FREE | MAT4 | PH31 |
| F3 | MUS1 | FR2 | MAT3 | ENG4 | MAT3 | FREE | MUS1 |
| F4 | SC12 | SC12 | ENG3 | GE41 | SC12 | GE41 | SC12 |
| F5 | ENG1 | PE2 | CH32 | GE41 | CH32 | GE41 | PE2 |

Figure 2.5 continued

| Period | BEGGS | WISE | SCILAB | SCILAB | LANGLB | GYM | TAPERC |
|---|---|---|---|---|---|---|---|
| M1 | HIS4 | FREE | PH32 | FREE | FREE | FREE | FREE |
| M2 | ENG1 | FR2 | FREE | FREE | FR2 | PE4 | FREE |
| M3 | ENG2 | FR1 | FREE | FREE | FR1 | FREE | FREE |
| M4 | ENG3 | FREE | SC11 | SC11 | FREE | FREE | FREE |
| M5 | ENG4 | FG3 | SC11 | SC11 | FG3 | FREE | FREE |
| TU1 | ENG4 | FR2 | CH31 | FREE | FR2 | FREE | FREE |
| TU2 | ENG2 | FR41 | CH31 | FREE | FR41 | FREE | FREE |
| TU3 | ENG3 | FR41 | FREE | FREE | FR41 | FREE | FREE |
| TU4 | ENG1 | PE3 | FREE | FREE | FREE | PE3 | FREE |
| TU5 | FREE | FR1 | FREE | FREE | FR1 | FREE | GE42 |
| W1 | ENG2 | FR1 | FREE | FREE | FR1 | FREE | FREE |
| W2 | ENG1 | FR2 | PH31 | FREE | FR2 | FREE | GE42 |
| W3 | HIS4 | FREE | PH31 | FREE | FREE | FREE | FREE |
| W4 | MAT3 | FR42 | SC11 | SC11 | FREE | FREE | FR42 |
| W5 | ENG4 | FG3 | SC11 | SC11 | FG3 | FREE | FREE |
| TH1 | ENG2 | PE1 | FREE | FREE | FREE | PE1 | GE42 |
| TH2 | ENG3 | FR42 | FREE | FREE | FREE | FREE | FR42 |
| TH3 | HIS4 | FG3 | SC12 | SC12 | FG3 | FREE | FREE |
| TH4 | ENG1 | FR2 | CH31 | FREE | FR2 | FREE | FREE |
| TH5 | ENG4 | FR1 | CH31 | FREE | FR1 | FREE | MUS2 |
| F1 | FREE | FR42 | PH31 | FREE | FREE | FREE | FR42 |
| F2 | ENG2 | FR1 | PH31 | FREE | FR1 | FREE | FREE |
| F3 | ENG4 | FR2 | FREE | FREE | FR2 | FREE | MUS1 |
| F4 | ENG3 | FREE | SC12 | SC12 | GE41 | FREE | FREE |
| F5 | ENG1 | FREE | CH32 | FREE | GE41 | PE2 | FREE |

Figure 2.6

| Period | 10 | 20 | 30 | 40 | 50 | 60 | 70 |
|---|---|---|---|---|---|---|---|
| 11 | 10 | 10 | 165 | 240 | 10 | 10 | 165 |
| 12 | 20 | 90 | 130 | 250 | 130 | FREE | 250 |
| 13 | 30 | 80 | 190 | 190 | FREE | 190 | 190 |
| 14 | 40 | 40 | 140 | 200 | 40 | 200 | 40 |
| 15 | 40 | 40 | 150 | 210 | 40 | 150 | 40 |
| 21 | 50 | 90 | 170 | 210 | 170 | FREE | 50 |
| 22 | 50 | 80 | 170 | 220 | 170 | FREE | 50 |
| 23 | 10 | 10 | 140 | 220 | 10 | 10 | FREE |
| 24 | 20 | 100 | 180 | 200 | FREE | 200 | 100 |
| 25 | 30 | 100 | 130 | 235 | 130 | 235 | 100 |
| 31 | 30 | 80 | 130 | 200 | 130 | 200 | FREE |
| 32 | 20 | 90 | 160 | 235 | FREE | 235 | 160 |
| 33 | 10 | 10 | 160 | 240 | 10 | 10 | 160 |
| 34 | 40 | 40 | 140 | 225 | 40 | FREE | 40 |
| 35 | 40 | 40 | 150 | 210 | 40 | 150 | 40 |
| 41 | 60 | 80 | 130 | 235 | 130 | 235 | FREE |
| 42 | 10 | 10 | 140 | 225 | 10 | 10 | FREE |
| 43 | 45 | 45 | 150 | 240 | 45 | 150 | 45 |
| 44 | 20 | 90 | 170 | 200 | 170 | 200 | FREE |
| 45 | 30 | 120 | 170 | 210 | 170 | FREE | 120 |
| 51 | 10 | 10 | 160 | 225 | 10 | 10 | 160 |
| 52 | 30 | 80 | 160 | 200 | FREE | 200 | 160 |
| 53 | 70 | 90 | 130 | 210 | 130 | FREE | 70 |
| 54 | 45 | 45 | 140 | 230 | 45 | 230 | 45 |
| 55 | 20 | 110 | 175 | 230 | 175 | 230 | 110 |

Figure 2.6 continued

| Period | 80 | 90 | 100 | 100 | 110 | 120 | 130 |
|---|---|---|---|---|---|---|---|
| 11 | 240 | FREE | 165 | FREE | FREE | FREE | FREE |
| 12 | 20 | 90 | FREE | FREE | 90 | 250 | FREE |
| 13 | 80 | 30 | FREE | FREE | 30 | FREE | FREE |
| 14 | 140 | FREE | 40 | 40 | FREE | FREE | FREE |
| 15 | 210 | 150 | 40 | 40 | 150 | FREE | FREE |
| 21 | 210 | 90 | 170 | FREE | 90 | FREE | FREE |
| 22 | 80 | 220 | 170 | FREE | 220 | FREE | FREE |
| 23 | 140 | 220 | FREE | FREE | 220 | FREE | FREE |
| 24 | 20 | 180 | FREE | FREE | FREE | 180 | FREE |
| 25 | FREE | 30 | FREE | FREE | 30 | FREE | 235 |
| 31 | 80 | 30 | FREE | FREE | 30 | FREE | FREE |
| 32 | 20 | 90 | 160 | FREE | 90 | FREE | 235 |
| 33 | 240 | FREE | 160 | FREE | FREE | FREE | FREE |
| 34 | 130 | 225 | 40 | 40 | FREE | FREE | 225 |
| 35 | 210 | 150 | 40 | 40 | 150 | FREE | FREE |
| 41 | 80 | 60 | FREE | FREE | FREE | 60 | 235 |
| 42 | 140 | 225 | FREE | FREE | FREE | FREE | 225 |
| 43 | 240 | 150 | 45 | 45 | 150 | FREE | FREE |
| 44 | 20 | 90 | 170 | FREE | 90 | FREE | FREE |
| 45 | 210 | 30 | 170 | FREE | 30 | FREE | 120 |
| 51 | FREE | 225 | 160 | FREE | FREE | FREE | 225 |
| 52 | 80 | 30 | 160 | FREE | 30 | FREE | FREE |
| 53 | 210 | 90 | FREE | FREE | 90 | FREE | 70 |
| 54 | 140 | FREE | 45 | 45 | 230 | FREE | FREE |
| 55 | 20 | FREE | 175 | FREE | 230 | 110 | FREE |

# CHAPTER THREE

## DESCRIPTION OF A SCHOOL

### 3.0 Introduction

Can the model defined in Chapter Two be used to describe the type of school whose time-table we wish to construct? To answer this question let us consider the various data documents in turn.

### 3.1 The Item List

Let us deal with the pupils first. One method of meeting the requirements of the model is to regard each pupil as a separate unit. Although perfectly sound in theory this approach would normally produce an organisation with a very large number of items. This would be a disadvantage because:-

i) it would make the list of units involved in each requirement very long hence increasing the data preparation costs;

ii) it would increase the computer storage space required and possibly exceed the limits available;

iii) it would increase the effort required to make each assignment hence increasing the total time required to make the time-table and the cost of the operation.

To avoid these disadvantages we must minimise the number of items as far as possible.

In most schools the group of pupils who are at the same stage in their education and of approximately the same age are called a form. A normal British Secondary School ranges from the 11-12 year olds in the First Form up to the 17-18 year olds in the Sixth Form. Since a form might have anything from 20 up to about 400 pupils it is usually sub-divided into smaller groups. A First Form of 150 pupils might be divided into six groups of 25 called 1A, 1B, 1C, 1D, 1E and 1F. We will refer to these smaller sets of pupils as basic groups. In general a basic group will have some name which is meaningful to members of the school staff. In one school $4T_1$ is used to denote the first of two groups of pupils in the Fourth Form who specialize in Technical subjects. It does not follow that the basic

group of pupils will always be together throughout the week but in many cases they are together for most of the week.

The simplest situation is where all members of a basic group follow exactly the same curriculum, i.e. we can imagine that the group can be assembled at the start of the week and never be separated. In this case the complete group is defined to be a unit. Since there can be no other unit equal to it the group is also an item with one life.

Consider now the situation when setting occurs. The idea of setting is that two or more basic groups of pupils have lessons in a particular subject at coincident times so that they may be regrouped according to their ability in that subject. To illustrate, suppose that there are six boys called A, B, C, D, E and F. For the majority of subjects A,B and C are in class 1 while D, E and F are in class 2. For Mathematics, however, this grouping is not satisfactory, thus the Mathematics lessons for classes 1 and 2 are planned to coincide and at these periods A, B and D form class 1M while C, E and F form class 2M. We say that 'Mathematics classes are setted across class 1 and class 2'. In this case A, B and C can be defined as one unit and D, E and F as a second unit. When a Mathematics lesson takes place both units are required to be present. It is not uncommon to find setting across five or six basic groups of pupils.

Setting can of course occur in several subjects, i.e. there are several regroupings in addition to the basic groups. The basic groups might be used for English, History, Geography, French, Religious Education, Music, Physical Education and Games with setting in Mathematics, Science and Craft subjects (Woodwork, Needlework etc.).

If, for a given number of basic groups, different sets are required for every subject then there is no point in defining each basic group as a unit. It is sufficient to define all the basic groups involved in the setting as a unit. This may or may not be equivalent to defining the whole form as a unit.

As a rough rule we may say that a first approximation to the best method of defining a set of units to describe the pupils in the school is to assign an item name to each of the basic groups used by the school staff. In some cases this will

produce more items than it is necessary to have but adjustments can easily be made later.

Dealing with the teachers is much more straight-forward. Normally no two teachers are exactly equal, in the sense of definition 2.1.2, thus each teacher is an item with one life i.e. we simply assign an item name to every teacher in the school. It can happen, however, that a given teacher always teaches in the same classroom, usually a special room of some sort. In this case the teacher and his special room may be defined as a single unit. In effect, this is leaving the room out of the problem. For example a school might have a swimming pool and a special swimming instructor who takes all the swimming classes and does not teach any other subject. The instructor and the pool may then be taken as a unit.

The majority of rooms in the school will be ordinary classrooms equipped simply with desks and a blackboard. In some schools the allocation of these rooms is a trivial task and if this is the case they may be left out of the problem completely. If their allocation is difficult then a single item with one life for each room may be defined since all these ordinary classrooms are equal. Specialized rooms e.g. science laboratories, gymnasia etc. provide problems in most school time-tables thus there must be one item with one or more lives for each different type of room. In deciding whether or not two rooms are of the same type the definition of equality (2.1.2) must be strictly applied. Difficulty can arise in the case of a room which has a dual purpose since dual roles are not catered for in the model.

Example

A science laboratory equipped for advanced Chemistry lessons would not be equal to a laboratory equipped for teaching General Science. The advanced laboratory, however, might be suitable for teaching General Science as well i.e. for part of the week it could be regarded as an extra life on the item which represents the ordinary laboratory. This is not possible within the model thus in listing the requirements it would be necessary to state which of the General Science lessons were to take place in the advanced laboratory.

Mobile equipment, which does not have a permanent place in one of the specialized rooms, can also be included in the list of items describing the school. For example, film projectors or tape recorders might be used regularly in lessons and have to be considered in making the time-table. Again there may be several pieces of identical equipment thus allowing an item with several lives to be defined.

The fact that our model allows items to have several lives rather than simply giving a different item name to each unit gives extra freedom in constructing the time-table for, if a school has three Physics laboratories, the request 'the Third Form Physics lesson must take place in a Physics laboratory' is less restrictive than 'the Third Form Physics lesson must take place in Physics laboratory number 1'.

In the complete system the Item List is referred to as 'Document Number One'.

## 3.2 The Time Structure List

Most schools have a basic time unit or period length of about 35 to 45 minutes and lessons usually last for 1, 2, 3 or 4 periods. In some instances there is slight irregularity in the period lengths e.g. the first period or the last period on one or more days might be five minutes shorter than the other periods. If we are to use the model we must assume that all periods are of the same length.

A day in the life of a school is usually about 7 - 9 periods and fits in with the definition in the model as normally no lesson will start on one day and be completed on the next.

The majority of schools work a five day week, Monday to Friday, but some also have three or four periods on Saturday morning and the time-table usually repeats each calendar week. However, a few schools work a five day week with a time-table cycle of six or seven days. Clearly the model can cater for any of these systems.

### 3.2.1 The space parameter and normal distribution

A manual time-tabler usually attempts to spread lessons in a given subject for a particular class evenly over the week. Figure 3.1 illustrates the sort of distributions which would be considered acceptable and those which he would normally try to

Figure 3.1

| Number of lessons | Acceptable distributions | Non-acceptable distributions |
|---|---|---|
| 1 | any day | - |
| 2 | Mon.,Wed. | Mon.,Mon. |
| | Mon.,Thur. | Mon.,Tues. |
| 3 | Mon.,Tues.,Thur. | Mon.,Mon.,Mon. |
| | Mon.,Wed.,Fri. | Mon.,Mon.,Thur. |
| | Mon.,Thur.,Fri. | Mon.,Tues.,Wed. |
| 4 | any four days | Mon.,Mon.,Tues.,Fri. |
| 5 | one on each day | Mon.,Mon.,Tues.,Thur.,Fri. |

avoid.

Although these ideas of distribution may be his aim he will often be prepared to accept a less than perfect distribution in order to obtain a complete time-table. The distribution Monday, Tuesday, Tuesday, Tuesday, Thursday, Friday for six single periods of English occurred in the time-table used by one large school in 1963-1964.

In the model the use of the space parameter equal to 1 or 2 and the built in constraint that the activities in each requirement or tied set of requirements must have a normal distribution ensures that a perfect distribution is obtained in the majority of cases.

The form of the period patterns allows the daily pattern of a school to be described easily.

Example

Consider a school day of eight periods broken by a short morning break of 10 to 15 minutes after the third period and a longer lunch break of 30 to 40 minutes after the fifth period. Suppose that double period lessons are not allowed to run over either of these breaks but a triple period lesson may run across the short break, then the set of period patterns for that day would be:-

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | - | single |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | - | double |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | - | triple |

The situation which is not catered for by the model is where an m-period lesson in one subject might be allowed to run across a minor break while a lesson of the same length in another subject would not. The m-period pattern applies to all m-period lessons.

In the complete system the Time Structure List is referred to as 'Document Number Two'.

3.3 The Requirement List

Most of the activities which take place in a school are lessons but there is no reason why other activities should not be included in the time-table as well. For example it might be required to arrange a period when certain senior members of the teaching staff could all be free together for consultations. Since in the model the description of an activity requires only a statement of the set of units which have to be available and the duration of the activity, the nature of the activity is not a part of the data. In particular, the subject which will be taught when a certain group of pupils and a certain teacher meet is irrelevant as far as making the time-table is concerned. The job of the time-tabler is to ensure that there is a time or set of times during the week when they can meet. The responsibility for making sure that they spend their time wisely rests elsewhere.

The simplest type of situation is where a single basic group of pupils is to be taught for a given number of equal length lessons by the same teacher in the same type of room. Each lesson is an activity but since they are all of the same length and involve the same set of units they are equal activities and so may be described by a single requirement.

Example

If the pupils are item 6, the teacher is item 74, the room is item 132 and 5 single period lessons are required then the requirement might be:-

26, 0, 1, 5, 6, 74, 132;

The order of the items is arbitrary thus

26, 0, 1, 5, 74, 132, 6;

would mean exactly the same thing.

If one or more of a group of related activities are unequal to the others due to being different in length or involving dif-

ferent sets of items then more than one requirement will be needed and the tie facility may be used to indicate the relation between the various requirements when distribution of the lessons is considered.

Example

For the lessons in the previous example, if four are held in a room of one type (item 132) but the fifth is held in a special room (item133) then the fifth lesson is not equal to the other four and two requirements are needed. They might be

26, 27, 1, 4, 6, 74, 132;

27, 26, 1, 1, 6, 74, 133;

Assuming a five day week, the fact that the requirements are tied together ensures that one lesson will occur on each day.

Example

If the pupils and teacher in the previous two examples have three one-period lessons and one two-period lesson all in the same room, then again, we must use two requirements. They might be

26, 27, 1, 3, 6, 74, 132;

27, 26, 2, 1, 6, 74, 132;

Use of the tie facility ensures that the two-period lesson will not occur on the same day as any of the one period lessons provided the week has at least four days.

When 'setting' occurs then the lesson or group of lessons is described as if all the classes, all the teachers and all the rooms involved are taking part in the same activity.

Example

Suppose three basic groups of pupils (items 3, 4 and 5) are to be united and redivided into three different groups which are to be taught for five single periods per week by three teachers (items 71, 72 and 73). Each lesson takes place in an ordinary classroom (item 120). One form of the requirement (number 104) which describes this set-up is

104, 0, 1, 5, 3, 4, 5, 71, 72, 73, 120, 120, 120;

An alternative form might be

104, 0, 1, 5, 3, 71, 120, 4, 72, 120, 5, 73, 120;

Of course the original three basic groups might be redivided into

four new groups needing four teachers and four rooms, in which case the requirement might become

104, 0, 1, 5, 3, 4, 5, 71, 72, 73, 76, 120, 120, 120, 120;

Requirements for activities other then lessons are straight forward.

Example

If the senior members of staff are items 50, 62, 71 and 86 then the meeting mentioned above might be described by

100, 0, 1, 1, 50, 62, 71, 86;

In this no classes or rooms are involved.

Example

If a certain class (item 8) spends a period in the school library (item 151) without the supervision of a teacher then the requirement might be

101, 0, 1, 1, 8, 151;

The completion of the Requirement List involves work by a large number of people in the school and in the complete system various other documents have to be prepared before it can be produced. Details of a suggested procedure for producing it are given in Chapter 4. The Requirement List is referred to as 'Document Number Six'.

## 3.4 The Fixed Free Period List

Most of the pupils in a school are involved in lessons throughout the week, thus a unit which represents a group of pupils will not normally have free periods. The exceptions are often the higher forms of the school where pupils have a few periods for private study. If these involve use of a special study room they may have to be treated as internal activities and appear on the Requirement List, otherwise the pupils may simply be regarded as free. Unless requests to the contrary are included on the Fixed Free Period List these private study periods will be inserted wherever convenient on the time-table. They will not necessarily occur with a normal distribution.

There are also cases where pupils in the lower form or forms of a school work for a few periods less than the rest of the school, e.g. they might finish one period early on one or more days of the week. Again the item or items representing the

pupils may be considered to be free but in this case requests on the Fixed Free Period List must be included to ensure that the free periods occur at the correct times.

Full-time teaching staff will probably be free from teaching duties for at least 12% of the total time with extra free time for staff with special administrative responsibilities, e.g. Heads of subject departments, Form Masters and House Masters. Again, unless requests to the contrary are included on the Fixed Free Period List, these periods will be inserted wherever convenient on the time-table.

Part-time staff will normally only be available at certain pre-determined times of the week. They must therefore be assigned fixed free periods at those periods when they are not available. They may also have a few other 'floating' free periods like full-time staff.

Rooms and equipment in most cases will be available throughout the week except where special facilities are shared with another school, e.g. a swimming pool or sports field. In this case the units involved may be treated as 'part-time' and assigned fixed free periods when they are not available.

In the complete system the Fixed Free Period List is known as 'Document Number Seven'.

3.5 The Forbidden Assignment List

This facility in the model allows all sorts of strange requirements to be catered for.

Example

Mr. McWhirter does not like to have the Upper Sixth Biology class on a Monday morning because many of the lessons require preparation of experiments on the preceding day.

Example

Fourth Form Domestic Science classes must take place in the three periods immediately before the lunch break as the pupils frequently prepare their own lunch. These lessons are therefore forbidden at all other periods.

Example

Swimming lessons must not be given immediately after the lunch break.

Example

Third Form pupils must not have Physical Education on Tuesday as they have Games (pre-assigned) on Tuesday afternoon.

In the complete system the Forbidden Assignment List is known as 'Document Number Eight'.

3.6 The Pre-assignment List

This is also a useful facility for dealing with special requirements.

Example

The Second Form French lessons must coincide with the broadcast French course at 10.45 a.m. on Mondays, Wednesdays and Fridays.

Example

The Fourth Form double period of Art must coincide with the first two periods of the Third Form triple period of Art. This must be pre-assigned since different length lessons are involved.

Example

The following is a situation from an actual school timetable.

At eight periods of the week the whole of the Third Form (150 pupils) which consists of six basic groups takes part in a set of options for which they are redivided into seven groups. The options are:-

| | |
|---|---|
| Group 1<br>Group 2<br>Group 3 | 4 single periods of German,<br>2 single periods of Physical Education,<br>2 single periods of Religious Education; |
| Group 4<br>Group 5<br>Group 6 | 1 double period of Art,<br>2 single periods of Music,<br>2 single periods of Physical Education,<br>2 single periods of Religious Education; |
| Group 7 | 4 single periods of Greek,<br>1 double period of Greek,<br>2 free periods. |

At any period there are the following restrictions due to the number of teachers available:-

not more than 2 classes may be taught Physical Education,
not more than 2 classes may be taught Religious Education,
not more than 1 class may be taught Art,
not more than 1 class may be taught Music.

The problem in describing these requirements within the model is caused by the double period of Art required by groups 4, 5 and 6. Since only one class at a time may be taught Art this requires that the eight periods shall include at least three pairs of consecutive periods. At each period of a pair the other four groups must have two different subjects except for group 7 which has one double period of Greek.

The solution used by the school was as shown in figure 3.2. The only method of dealing with this example within the model is to pre-assign the complete set of lessons.

Figure 3.2

| | Groups | | | | | | |
|---|---|---|---|---|---|---|---|
| Period | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | German | German | German | P.E. | P.E. | Music | Greek |
| 12 | German | German | German | R.E. | Art | R.E. | Greek |
| 13 | R.E. | R.E. | P.E. | Music | Art | P.E. | Greek |
| 22 | German | German | German | Art | Music | P.E. | Greek |
| 23 | P.E. | P.E. | R.E. | Art | R.E. | Music | Free |
| 26 | R.E. | R.E. | P.E. | P.E. | Music | Art | Greek |
| 27 | P.E. | P.E. | R.E. | Music | R.E. | Art | Free |
| 33 | German | German | German | R.E. | P.E. | R.E. | Greek |

In the complete system the Pre-Assignment List is known as 'Document Number Nine'.

## 3.7 Factors which the model excludes

The model assumes that the units of the organisation are in the same location and physical state at the start and finish of each activity. This does not rule out movement or physical change but it does mean that if an activity requires changes in location and/or state then it must also include restoration to

the original location and/or state before the end of the activity.

This assumption means that the model cannot be used to describe a school which is spread over a large area for in such a school an important factor in making the time-table may be the time taken by pupils or teachers to travel from one section of the school to another. This problem has increased in recent years with the merging of schools, often a mile or more apart, to form new comprehensive units. In the majority of schools, however, the time taken to travel from any part of the school to any other is ignored by the time-tabler, thus this weakness in the model is but a minor disadvantage.

Physical changes in the staff and pupils are also of minor importance. A physical education teacher may have preferences as to the number of classes he has to take in a day but apart from that there seems little need to take account of the physical state of the personnel. In the case of laboratories, however, it may be necessary in some cases to allow time before a lesson for preparation. Often the teacher will be expected to do this work outside school hours or during meal breaks but in special cases he may be allowed a period for preparation before a certain lesson. This means that the preparation period must come immediately before the lesson. The requirements concerned might be:-

106, 0, 1, 1, 95, 192;

107, 0, 3, 1, 16, 95, 192;

where 16 is the class, 95 is the teacher and 192 is the laboratory. The only way to ensure that they occur in the time-table as required is to pre-assign both. Similar situations may arise with tidying up after certain lessons.

## 3.8 Conclusion

Thus the model defined in Chapter Two is capable of describing sufficiently accurately those aspects of a school which are relevant to the construction of its time-table.

# CHAPTER FOUR

## PREPARATION OF THE DATA DOCUMENTS

### 4.0 Introduction

In Chapter One the process of producing a time-table was said to have three phases:-

Phase 1 - Production of the Curriculum;

Phase 2 - Assigning a teacher and a room to each lesson;

Phase 3 - Fixing when each lesson is to take place.

Phases 1 and 2 may now be combined into the single operation of producing the data documents defined in Chapter Two. If, however, the staff of a school is asked simply to produce these documents without guidance on how to go about it the results are liable to be unsatisfactory due to many errors. The average school teacher is not used to working to the standards of accuracy required by a computer. In this chapter we describe a recommended procedure to be followed by the school staff with several intermediate documents which are designed to avoid errors in the final data documents.

The complete system consists of nine documents:-

Document Number One - The Item List
Two - The Time Structure List
Three - The Curriculum
Four - The Teacher Duty Chart
Five - The Room and Equipment Usage Chart
Six - The Requirement List
Seven - The Fixed Free Period List
Eight - The Forbidden Assignment List
Nine - The Pre-assignment List.

In describing the preparation of these documents we will assume that advice will be available to the members of the school staff from someone experienced in the use of the system. We will refer to this person as the 'analyst'.

### 4.1 Document Number One - The Item List

Construction of the Item List should be done by a senior member of the school staff at the same time as he is constructing

the Curriculum. Normally he would require some guidance from the analyst especially if a large amount of setting is required or the Sixth Form structure is complex.

In listing the items it has become our practice to list classes first, followed by teachers, rooms and other equipment in that order although this sequence is not demanded by the model. If, however, changes to the list have to be made it is best to delete incorrect entries completely and to add the corrections to the end of the list.

The Item List for the example problem is shown, on a suitable form, in figure 4.1.

## 4.2 Document Number Two - The Time Structure List

This is a relatively simple document to prepare although advice from the analyst might be required when working out the period patterns.

The Time Structure List for the example problem is shown, on a suitable two page form, in figures 4.2 and 4.3.

## 4.3 Document Number Three - The Curriculum

The information which forms the Curriculum is eventually incorporated in the Requirement List but the construction of the Curriculum is a necessary preliminary step. Its purpose is to record, in an unambiguous form, the course to be provided for each class within the school. The Curriculum for the example problem is shown in figure 4.5.

Each set of lessons is represented by an area enclosed by a solid line. Within the area is recorded:-

i) the subject involved;
ii) how the set of periods is to be broken down into lessons;
iii) a reference name or number for each length of lesson involved.

Figure 4.4 shows one method of recording this information and several examples.

If numeric names are used the areas on the Curriculum can be numbered in steps of 10 using the intermediate numbers only when the area requires more than one number. If alphanumeric names are used great care must be exercised in their selection as no two names may be the same.

**********
* SCATTAR *
**********

DOCUMENT NUMBER ONE
**********
THE ITEM LIST
**********

Name of School THE EXAMPLE PROBLEM

Page 1 of 1 pages

Notes: 1. Remember that no two items may have the same code name or number.

2. In the column marked 'L' fill in the total number of units of the item in the school.

3. In the column marked 'T' fill in the type code for the item. The type codes are:-
Class = 1, Teacher = 2, Room or Equipment = 3.

4. The details in the last column are for use by your school staff and NOT by the computer.

| NORMAL NAME OR DESCRIPTION OF ITEM | CODE NAME OR NUMBER | L | T | OTHER DETAILS |
|---|---|---|---|---|
| FIRST FORM | 10 | 1 | 1 | |
| SECOND FORM | 20 | 1 | 1 | |
| THIRD FORM (SCI.) | 30 | 1 | 1 | |
| THIRD FORM (LIT.) | 40 | 1 | 1 | |
| MR. J.P. SMITH | 50 | 1 | 2 | MATHS, SCI. |
| MISS P.L. JONES | 60 | 1 | 2 | MATHS., LANG., MUSIC |
| MR. B.D. DAVIS | 70 | 1 | 2 | SCI., P.E., ART, MUSIC |
| MR. E.A. BEGGS | 80 | 1 | 2 | ENGL., HIST. |
| MR. O.L. WISE | 90 | 1 | 2 | LANG., P.E. |
| SCIENCE LAB. | 100 | 2 | 3 | |
| LANGUAGE LAB. | 110 | 1 | 3 | HOLDS 16 PUPILS |
| GYM | 120 | 1 | 3 | |
| TAPE RECORDER | 130 | 1 | 3 | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

TOTAL NUMBER OF ITEMS 13

(to be filled in on the last page only)

Figure 4.2

SCATTAR

DOCUMENT NUMBER TWO

THE TIME STRUCTURE LIST

Name of School THE EXAMPLE PROBLEM



The number of days per week is 5 .

The details of the days are:-

| Name | Number of periods |
|---|---|
| 1 | 5 |
| 2 | 5 |
| 3 | 5 |
| 4 | 5 |
| 5 | 5 |
| | |
| | |

The names of the periods are:-

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 11 | 12 | 13 | 14 | 15 | | | | |
| 21 | 22 | 23 | 24 | 25 | | | | |
| 31 | 32 | 33 | 34 | 35 | | | | |
| 41 | 42 | 43 | 44 | 45 | | | | |
| 51 | 52 | 53 | 54 | 55 | | | | |
| | | | | | | | | |
| | | | | | | | | |

The longest lesson lasts for 2 periods.

If two lessons in the same subject take place on the same day there should be 0 periods of other subjects between them

Figure 4.3

```
**********
* SCATTAR *
**********
```

DOCUMENT NUMBER TWO
**********
THE TIME STRUCTURE LIST
**********

Name of School THE EXAMPLE PROBLEM

Page 2

The period patterns are:-

Single periods

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | | | | |
| 1 | 1 | 1 | 1 | 1 | | | | |
| 1 | 1 | 1 | 1 | 1 | | | | |
| 1 | 1 | 1 | 1 | 1 | | | | |
| 1 | 1 | 1 | 1 | 1 | | | | |
| | | | | | | | | |
| | | | | | | | | |

Double periods

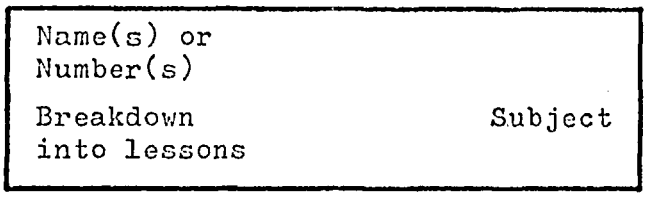
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | | | | |
| 1 | 1 | 0 | 1 | 0 | | | | |
| 1 | 1 | 0 | 1 | 0 | | | | |
| 1 | 1 | 0 | 1 | 0 | | | | |
| 1 | 1 | 0 | 1 | 0 | | | | |
| | | | | | | | | |
| | | | | | | | | |

Triple periods

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |

Figure 4.4

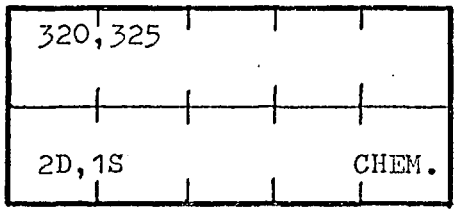
Format for each area:

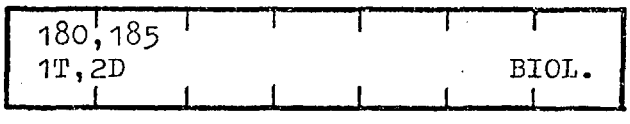



Code: nS = n single period lessons;
nD = n double period lessons;
nT = n triple period lessons;
nQ = n quadruple period lessons.

Examples :




The Curriculum would normally be prepared by a senior member of the school staff in consultation with some of his colleagues but some aid from the analyst would usually be required to express it in the standard form.

It may not be convenient for the senior member of staff to divide each subject area into individual lessons as he may not know the preferences of the teachers concerned. It is sufficient if he leaves the area undivided and gives it a single name or number. Additional names or numbers can easily be inserted while the Requirement List is being prepared.

When the Curriculum has been prepared copies of it are reproduced for distribution to the persons who will decide which teacher is to take each class.

## 4.4 Document Number Four - The Teacher Duty Chart
## Document Number Five - The Room and Equipment Usage Chart
## Document Number Six - The Requirement List

Once the Curriculum has been prepared the next stage in the process is to prepare the Requirement List i.e. to decide which teachers and which rooms will be required for each lesson on the Curriculum. In most schools this job is done by the Heads of Departments.

The suggested procedure is that each Head of Department should be given a copy of the Curriculum and be asked to prepare the requirements for all lessons involving the subjects for which he is responsible. Each requirement is recorded on an individual Requirement Card examples of which are shown in figure 4.8. Special arrangements have to be made for the preparation of requirements which represent options and involve subjects from different departments.

In performing this task the Head of Department has to make sure that the decisions he makes are consistent with the number of periods which each member of his staff should teach and the number of periods for which each room can be used. It is not unheard of for a Head of Department to put even himself down for teaching duties far in excess of the total number of periods in the week. To avoid this possibility, and to provide a check on the data, each Head of Department is required to record the duties he has allotted to each teacher and the use he has made of each room or piece of special equipment. The documents provided for this purpose are the Teacher Duty Chart and the Room and Equipment Usage Chart. These documents for the example problem are shown in figures 4.6 and 4.7.

The row of elements for a given teacher on the Teacher Duty Chart records the number of periods and the requirement name of each lesson to which the teacher has been assigned. It may also record any other duties which he has to perform. The last two columns record the total number of periods at which he is involved in internal activities and external activities respectively.

The Room and Equipment Usage Chart performs exactly the same role for rooms and equipment.

Figure 4.5 The Curriculum

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | FIRST FORM | 10 | 20 5S ENG. | 30 5S FR. | 40, 45 | 50 1D A. | 60 P.E. | 70 M. | 25 | 0 |
| 20 | SECOND FORM | 5S MA. | 80 5S ENG. | 90 5S FR. | 2D, 2S SCI. | 100 1D A. | 110 P.E. | 120 M. | 25 | 0 |
| 30 | THIRD FORM(SCI) | 130 5S MA. | 140 5S ENG. | 150 3S FR. or GE. | 160, 165 2D, 1S PHY. | 170, 175 2D, 1S CH. | 180 P.E | 190 | 25 | 0 |
| 40 | THIRD FORM(LIT) | 200 5S MA. | 210 5S ENG. | 220, 225 1D, 3S FR. | 230, 235 1D, 3S GE. | 240 3S H. | 250 P.E | A. or M. | 25 | 0 |
| | | | | | | | | | * | ** |

Figure 4.6 The Teacher Duty Chart

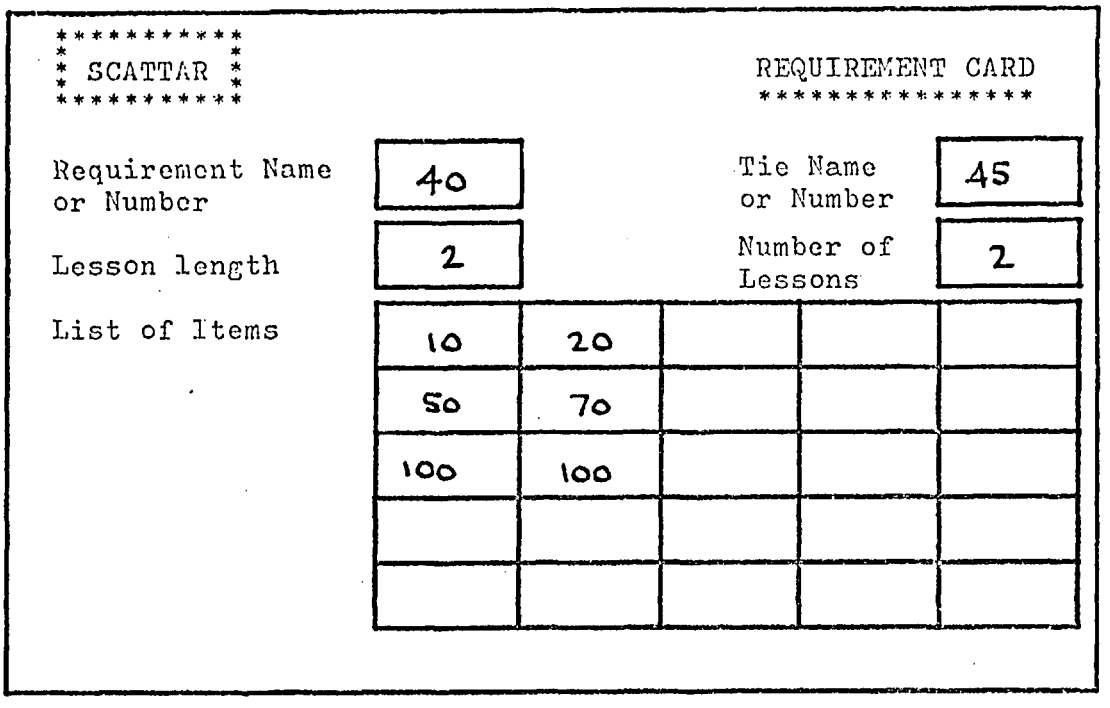
| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 50 | SMITH | 10 | 40 | 45 | 130 | 170 | 175 | FREE | 21 | 4 |
| 60 | JONES | 10 | 150 | 190 | 200 | 230 | 235 | ADMIN. | FREE | 19 | 6 |
| 70 | DAVIS | 40 | 45 | 50 | 70 | 190 | 110 | 120 | 160 | 165 | 190 | 250 | CAREERS | FREE | 20 | 5 |
| 80 | BEGGS | 20 | 80 | 140 | 210 | 240 | FREE | 23 | 2 |
| 90 | WISE | 30 | 60 | 90 | 150 | 180 | 220 | 225 | ADMIN. | FREE | 20 | 5 |
| | | | | | | | | | | | | | * | ** |

Figure 4.7 The Room and Equipment Usage Chart

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 100 | SCIENCE LAB. | 40 | 45 | 160 | 165 | 170 | 175 | 22 | 28 |
| 110 | LANG. LAB. | 30 | 90 | 150 | 220 | 230 | | 17 | 8 |
| 120 | GYM | 60 | 110 | 180 | 250 | | | 4 | 21 |
| 130 | TAPE REC. | 70 | 120 | 225 | 235 | | | 8 | 17 |
| | | | | | | | | * | ** |

* Total Internal Activities ** Total External Activities

Figure 4.8




While filling in his requirement cards a Head of Department may find that he wishes to change the pattern of lessons recorded on the Curriculum within a subject area. Provided he understands that any new requirement names he introduces must be different from those already on the Curriculum there is no reason why he should not be permitted to make whatever changes he pleases. He must never, of course, alter any subject area which is the responsibility of another Head of Department without discussing it with him.

Another circumstance in which it is necessary to introduce extra requirement names is where one teacher is allotted to one or two of a set of lessons while another teacher gets the remainder. Due to the restrictions of the model this requires that the two sets of lessons be described in different requirements. Usually sets of requirements introduced in this way have to be tied together to obtain a proper distribution thus the Heads of Department must fully understand the ideas behind tied sets. In practice it has been found that they picked up these tricks of the trade very quickly.

Figure 4.9

```
**********
* SCATTAR *
**********
```

Name of School THE EXAMPLE PROBLEM

Page 1 of 1 pages

General Note: The details in the last column of each section of this form are for use by your school staff and NOT by the computer.

DOCUMENT NUMBER SEVEN - THE FIXED FREE PERIOD LIST

Notes: 1. In the column marked 'I' fill in the name or number of the unit which is to be free.
2. In the area marked 'P' fill in the name(s) or number(s) of the period(s) at which the unit is to be free.

| I | P | | | | | OTHER DETAILS |
|---|---|---|---|---|---|---|
| 100 | 11 | 25 | 25 | | | |
| 60 | 34 | 45 | | | | On career advice duties |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

DOCUMENT NUMBER EIGHT - THE FORBIDDEN ASSIGNMENT LIST

Notes: 1. In the column marked 'R' fill in the name or number of the requirement which is forbidden.
2. In the area marked 'P' fill in the name(s) or numbers(s) of the periods at which the requirement is forbidden.

| R | P | | | | | OTHER DETAILS |
|---|---|---|---|---|---|---|
| 100 | 21 | 22 | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

continued

4.5 Document Number Seven - The Fixed Free Period List
Document Number Eight - The Forbidden Assignment List
Document Number Nine - The Pre-assignment List

In every school there are special needs or complications which can only be allowed for by using the facilities provided in the model for fixing free periods, for forbidding lessons from taking place at certain times and for making sure that some lessons do take place at certain times.

Each Head of Department is, therefore, also supplied with a form on which to record his section of the Fixed Free Period List, the Forbidden Assignment List and the Pre-assignment List. The sorts of situation which he can allow for on these documents were discussed in 3.4, 3.5 and 3.6. In some cases where particularly complicated situations have to be described he may require advice from the analyst. These documents for the example problem are shown in figure 4.9.

4.6 Final assembly

When each person has completed his section of the task all the various documents are collected together and they are then ready for the next stage of the process which is to check that the data is correct.

Figure 4.9 continued

DOCUMENT NUMBER NINE - THE PRE-ASSIGNMENT LIST
**********

Notes: 1. In the column marked 'R' fill in the name or number of the requirement in which a lesson is to be pre-assigned.
2. In the column marked 'P' fill in the name or number of the period at which the pre-assigned lesson is to start.

| R | P | OTHER DETAILS |
|---|---|---|
| 150 | 15 | |
| 200 | 14 | |
| | | |
| | | |
| | | |
| | | |

| R | P | OTHER DETAILS |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

# CHAPTER FIVE

## CHECKING THE DATA

### 5.0 Introduction

When the set of hand-written data documents reaches the computing centre the appropriate parts of Documents 1, 2, 6, 7, 8 and 9 are transferred to a suitable medium for computer input. During development punched paper tape and punched cards have been used. Punched cards have proved the more convenient as they enable corrections and changes to the data to be made easily. We will, therefore, assume that the data has been punched on cards and that the correct document terminator constants have been inserted. As far as the Atlas ALGOL system is concerned the data can be punched in an unformatted form but from the user's point of view it is convenient to set out the data more or less as shown in figures 2.1 or 2.2 taking a new card for each line.

It must be assumed that the data deck so produced contains errors which were either present in the hand-written documents or were introduced during the punching or deck assembly stages. All these errors must be corrected before any attempt to construct the time-table is made. Unfortunately the computer cannot be programmed to detect every type of mistake but it can help considerably by:-

i) checking the data documents as far as it can

and ii) listing all nine data documents for reference purposes and so that additional manual checks can be carried out.

The program written for this purpose is called the Data Checking Routine.

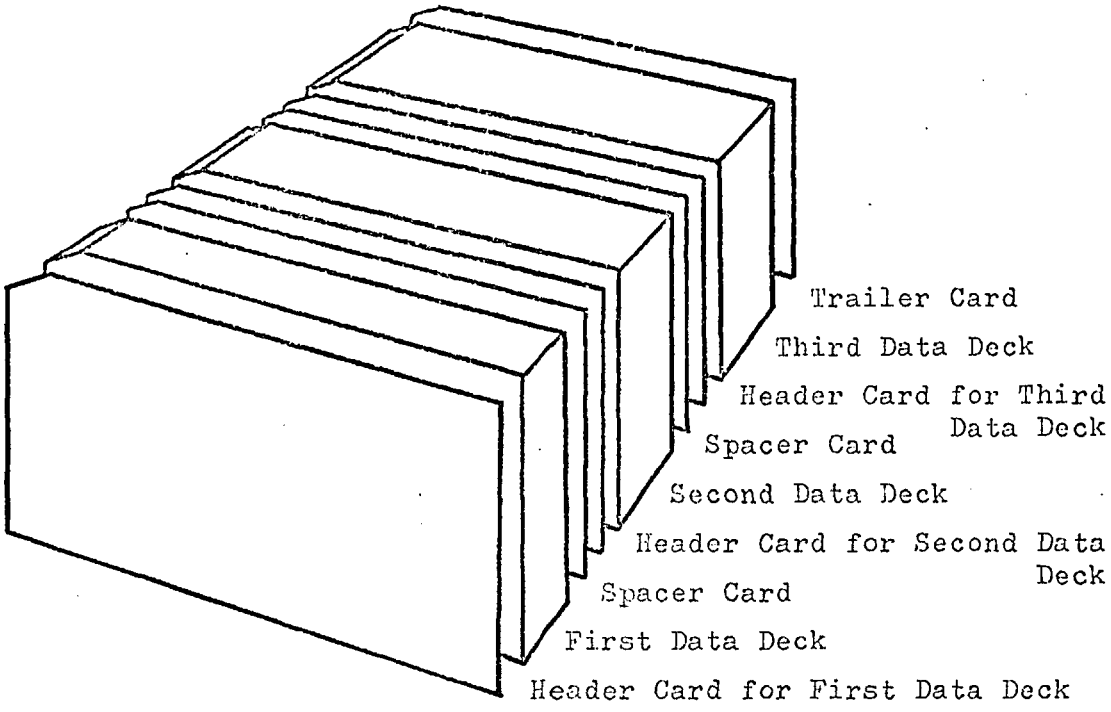
### 5.1 The Data Checking Routine

Slightly different versions of this program are required for the alternative forms of data presentation. We will discuss only the program for checking numeric data.

#### 5.1.1 Input data

The Data Checking Routine accepts several sets of data. The complete data deck is constructed as shown in figure 5.1.

Figure 5.1




A Header Card contains any positive integer called the Case Number which is used to identify the problem.
A Spacer Card contains the integer 10000.
A Trailer Card contains the integer 10001.

Due to restrictions imposed by the Main Program and as an aid to the data checking process limits have been imposed on the size of various quantities. These limits are shown in figure 5.2.

5.1.2 Output

If the data presented is correct as far as the program can see then it simply lists the nine data documents as shown in figure 5.3. If, however, there are detectable errors in the data then the program inserts diagnostic messages in the listing and in certain cases terminates it. A complete list of the errors detected, with the messages printed out and any associated operations, is given figure 5.4.

As in any operation, where a computer is checking a document which should be in a particular form but is not, one error can cause many others. This fact must be allowed for when interpreting the output.

Figure 5.2

| Quantity | Limit(s) |
|---|---|
| Total number of items | ≤ 300 |
| Total number of units | ≤ 400 |
| Total number of lives for any item | ≤ 100 |
| Total number of days | ≤ 10 |
| Total number of periods | ≤ 50 |
| Longest activity parameter | ≤ 5 |
| Total number of requirements | ≤ 600 |
| Numeric name for any item, day, period or requirement | > 0 and < 9999 |
| *Total number of lessons in any requirement or tied set of requirements | ≤ the number of days per week |

* One version of the Main Program allows single period lessons in an untied requirement up to twice the number of days in the week.

5.2 Manual checks

In addition to correcting any errors detected by the computer the user must check the listing produced by the Data Checking Routine for any other errors which the computer is not capable of finding. The most important step is to check Documents 3, 4 and 5 as they appear in the listing against the hand-written version produced by the school staff. This and the other checks which should be performed are listed in figure 5.5.

It is suggested that the computer centre staff or the analyst should check the output for any errors which have been caused by incorrect punching. When these have been removed and a new listing obtained a member, or members, of the school staff should perform the manual checks.

Only when a 'clean' listing has been obtained should the data be passed as ready for the Main Program. This does not mean that it will accept the data at once as there are other inconsistencies which can exist in the data but which cannot be detected until an attempt to make the time-table is made. For example, the pre-assignment of a lesson to a period at which it cannot occur will not be noticed until an attempt to perform the pre-assignment is made. Details of the problems detected by the Main Program are given in Chapter 10.

Figure 5.3

```
***********
* SCATTAR *
***********

                    ******************************
                    DATA CHECKING ROUTINE LISTING
                    ******************************

                          CASE NUMBER    1
                          ***************

DOCUMENT NUMBER ONE                                          THE ITEM LIST
*******************                                          *************

  NO NAME LIVES TYPE     NO NAME LIVES TYPE     NO NAME LIVES TYPE     NO NAME LIVES TYPE

   1   10    1    1       5   50    1    2       8   80    1    2      11  110    1    3
   2   20    1    1       6   60    1    2       9   90    1    2      12  120    1    3
   3   30    1    1       7   70    1    2      10  100    2    3      13  130    1    3
   4   40    1    1
```

continued

Figure 5.3 continued

DOCUMENT NUMBER TWO
**********

THE TIME STRUCTURE LIST
**********

THE NUMBER OF DAYS IS 5

THE DETAILS OF THE DAYS ARE

| NO | NAME | PERIODS |
|---|---|---|
| 1 | 1 | 5 |
| 2 | 2 | 5 |
| 3 | 3 | 5 |
| 4 | 4 | 5 |
| 5 | 5 | 5 |

THE PERIOD NAMES ARE

| | | | | |
|---|---|---|---|---|
| 11 | 12 | 13 | 14 | 15 |
| 21 | 22 | 23 | 24 | 25 |
| 31 | 32 | 33 | 34 | 35 |
| 41 | 42 | 43 | 44 | 45 |
| 51 | 52 | 53 | 54 | 55 |

THE LONGEST ACTIVITY PARAMETER IS 2

THE SPACE PARAMETER IS 0

THE PERIOD PATTERNS ARE

| | | | | |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |

continued

Figure 5.3 continued

DOCUMENT NUMBER SIX
**********

THE REQUIREMENT LIST
**********

| IRN | ERN | TIES | LENGTH | LIST OF ITEMS | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 0 | 1 | 5 | 10 | 20 | 50 | 60 | | |
| 2 | 20 | 0 | 1 | 5 | 10 | 80 | | | | |
| 3 | 30 | 0 | 1 | 5 | 10 | 90 | 110 | | | |
| 4 | 40 | 45 | 2 | 2 | 10 | 20 | 50 | 70 | 100 | 100 |
| 5 | 45 | 40 | 1 | 2 | 10 | 20 | 50 | 70 | 100 | 100 |
| 6 | 50 | 0 | 2 | 1 | 10 | 70 | | | | |
| 7 | 60 | 70 | 1 | 1 | 10 | 90 | 120 | | | |
| 8 | 70 | 60 | 1 | 1 | 10 | 70 | 130 | | | |
| 9 | 80 | 0 | 1 | 5 | 20 | 80 | | | | |
| 10 | 90 | 0 | 1 | 5 | 20 | 90 | 110 | | | |
| 11 | 100 | 0 | 2 | 1 | 20 | 70 | | | | |
| 12 | 110 | 120 | 1 | 1 | 20 | 70 | 120 | | | |
| 13 | 120 | 110 | 1 | 1 | 20 | 70 | 130 | | | |
| 14 | 130 | 0 | 1 | 5 | 30 | 50 | | | | |
| 15 | 140 | 0 | 1 | 5 | 30 | 80 | | | | |
| 16 | 150 | 0 | 1 | 3 | 30 | 60 | 90 | 110 | | |
| 17 | 160 | 165 | 2 | 2 | 30 | 70 | 100 | | | |
| 18 | 165 | 160 | 1 | 1 | 30 | 70 | 100 | | | |
| 19 | 170 | 175 | 2 | 2 | 30 | 50 | 100 | | | |
| 20 | 175 | 170 | 1 | 1 | 30 | 50 | 100 | | | |
| 21 | 180 | 0 | 1 | 1 | 30 | 90 | 120 | | | |
| 22 | 190 | 0 | 1 | 1 | 30 | 40 | 60 | 70 | | |
| 23 | 200 | 0 | 1 | 5 | 40 | 60 | | | | |
| 24 | 210 | 0 | 1 | 5 | 40 | 80 | | | | |
| 25 | 220 | 225 | 2 | 1 | 40 | 90 | 110 | | | |
| 26 | 225 | 220 | 1 | 3 | 40 | 90 | 130 | | | |
| 27 | 230 | 235 | 2 | 1 | 40 | 60 | 110 | | | |
| 28 | 235 | 230 | 1 | 3 | 40 | 60 | 130 | | | |
| 29 | 240 | 0 | 1 | 3 | 40 | 80 | | | | |
| 30 | 250 | 0 | 1 | 1 | 40 | 70 | 120 | | | |

continued

DOCUMENT NUMBERS THREE FOUR AND FIVE
**********

THE CURRICULUM
**********
THE TEACHER DUTY CHART
**********
THE ROOM AND EQUIPMENT USAGE CHART
**********

ITEM NAME 10

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 10 | 5 | 20 | 5 | 30 | 5 | 40 | 4 | 45 | 2 |
| 50 | 2 | 60 | 1 | 70 | 1 | | | | |

TOTAL INTERNAL ACTIVITIES = 25
TOTAL EXTERNAL ACTIVITIES = 0

ITEM NAME 20

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 10 | 5 | 40 | 4 | 45 | 2 | 80 | 5 | 90 | 5 |
| 100 | 2 | 110 | 1 | 120 | 1 | | | | |

TOTAL INTERNAL ACTIVITIES = 25
TOTAL EXTERNAL ACTIVITIES = 0

ITEM NAME 30

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 130 | 5 | 140 | 5 | 150 | 3 | 160 | 4 | 165 | 1 |
| 170 | 4 | 175 | 1 | 180 | 1 | 190 | 1 | | |

TOTAL INTERNAL ACTIVITIES = 25
TOTAL EXTERNAL ACTIVITIES = 0

ITEM NAME 40

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 190 | 1 | 200 | 5 | 210 | 5 | 220 | 2 | 225 | 3 |
| 230 | 2 | 235 | 3 | 240 | 3 | 250 | 1 | | |

TOTAL INTERNAL ACTIVITIES = 25
TOTAL EXTERNAL ACTIVITIES = 0

Figure 5.3 continued

Figure 5.2 continued

```
ITEM NAME   50
              10   5      40   4      45   2     130   5     170   4
             175   1

                                                  TOTAL INTERNAL ACTIVITIES =  21
                                                  TOTAL EXTERNAL ACTIVITIES =   4

ITEM NAME   60
              10   5     150   3     190   1     200   5     230   2
             235   3

                                                  TOTAL INTERNAL ACTIVITIES =  19
                                                  TOTAL EXTERNAL ACTIVITIES =   6

ITEM NAME   70
              40   4      45   2      50   2      70   1     100   2
             110   1     120   1     160   4     165   1     190   1
             250   1

                                                  TOTAL INTERNAL ACTIVITIES =  20
                                                  TOTAL EXTERNAL ACTIVITIES =   5

ITEM NAME   80
              20   5      80   5     140   5     210   5     240   3
                                                  TOTAL INTERNAL ACTIVITIES =  23
                                                  TOTAL EXTERNAL ACTIVITIES =   2

ITEM NAME   90
              30   5      60   1      90   5     150   3     180   1
             220   2     225   3

                                                  TOTAL INTERNAL ACTIVITIES =  20
                                                  TOTAL EXTERNAL ACTIVITIES =   5
```

continued

Figure 5.3 continued

```
ITEM NAME  100
             40   4     40   4     45   2     45   2    160   4
            165   1    170   4    175   1

                                                TOTAL INTERNAL ACTIVITIES =  22
                                                TOTAL EXTERNAL ACTIVITIES =  28

ITEM NAME  110
             30   5     90   5    150   3    220   2    230   2
                                                TOTAL INTERNAL ACTIVITIES =  17
                                                TOTAL EXTERNAL ACTIVITIES =   8

ITEM NAME  120
             60   1    110   1    180   1    250   1
                                                TOTAL INTERNAL ACTIVITIES =   4
                                                TOTAL EXTERNAL ACTIVITIES =  21

ITEM NAME  130
             70   1    120   1    225   3    235   3
                                                TOTAL INTERNAL ACTIVITIES =   8
                                                TOTAL EXTERNAL ACTIVITIES =  17

DOCUMENT NUMBER SEVEN                      THE FIXED FREE PERIOD LIST
**********************                     **************************

ITEM NAME     PERIOD NAMES

   100        11    25    25
    60        34    45
```

continued

DOCUMENT NUMBER EIGHT
**********

THE FORBIDDEN ASSIGNMENT LIST
**********

| REQ.NAME | PERIOD NAMES |
|---|---|
| 100 | 21 22 |

DOCUMENT NUMBER NINE
**********

THE PRE-ASSIGNMENT LIST
**********

| REQ.NAME | PERIOD NAME |
|---|---|
| 150 | 15 |
| 200 | 14 |

THIS LISTING IS COMPLETE

Figure 5.3 continued

Figure 5.4

| SECTION OF DATA AND ERROR | MESSAGE OUTPUT | ACTION CODE (see below) |
|---|---|---|
| ANYWHERE | | |
| An integer < 0 or > 10001 | ILLEGAL NUMBER x | 1 |
| THE ITEM LIST | | |
| The same item name used two or more times | REPEATED ITEM NAME x | 2 |
| Number of lives for an item > 100 | NUMBER OF LIVES TOO LARGE THE UPPER LIMIT IS 100 | 2 |
| Type code other than 1, 2 or 3 | ILLEGAL TYPE CODE x | 2 |
| Number of items > 300 | THIS PROBLEM INVOLVES TOO MANY ITEMS. THE UPPER LIMIT IS 300 | 3 |
| Number of units > 400 | THIS PROBLEM INVOLVES TOO MANY UNITS. THE UPPER LIMIT IS 400 | 3 |
| Number of items not as stated | SOME ITEMS ARE MISSING FROM THE ITEM LIST | 4 |
| | or | |
| | THERE ARE SOME EXTRA ITEMS ON THE ITEM LIST followed by THE STATED NUMBER OF ITEMS IS x THE ACTUAL NUMBER OF ITEMS IS y | 4 |
| Terminator in the wrong place | THE ITEM LIST STARTS WITH A TERMINATOR | 3 |
| | or | |
| | THE ITEM LIST IS TERMINATED IN THE MIDDLE OF A RECORD | 5 |
| THE TIME STRUCTURE LIST | | |
| Number of days > 10 | THIS PROBLEM INVOLVES TOO MANY DAYS. THE UPPER LIMIT IS 10 | 3 |
| The same day name used two or more times | REPEATED DAY NAME x | 2 |
| Number of periods > 50 | THIS PROBLEM INVOLVES TOO MANY PERIODS. THE UPPER LIMIT IS 50 | 3 |
| The same period name used two or more times | REPEATED PERIOD NAME x | 2 |

continued

Figure 5.4 continued

| SECTION OF DATA AND ERROR | MESSAGE OUTPUT | ACTION CODE (see below) |
|---|---|---|
| THE TIME STRUCTURE LIST | | |
| Longest activity parameter > 5 | THE LONGEST ACTIVITY PARAMETER x IS TOO HIGH. THE UPPER LIMIT IS 5 | 3 |
| Number in period pattern other than 0 or 1 | ILLEGAL ELEMENT IN PERIOD PATTERN x | 2 |
| Extra data before the terminator | List of extra data | 6 |
| Terminator in the wrong place | THE TIME STRUCTURE LIST STARTS WITH A TERMINATOR<br>or<br>THE TIME STRUCTURE LIST IS NOT COMPLETE | 3<br><br>5 |
| THE REQUIREMENT LIST | | |
| The same requirement name used two or more times | REPEATED REQUIREMENT NAME x | 2 |
| Lesson length too long | LESSON LENGTH GREATER THAN THE STATED MAXIMUM | 2 |
| Too many lessons in a single requirement | TOO MANY LESSONS IN A REQUIREMENT x | 2 |
| Undefined item | THE ITEM NAMED x DOES NOT APPEAR ON THE ITEM LIST | |
| Number of requirements > 600 | THIS PROBLEM INVOLVES TOO MANY REQUIREMENTS. THE UPPER LIMIT IS 600 | 3 |
| Number of requirements not as stated | SOME REQUIREMENTS ARE MISSING FROM THE REQUIREMENT LIST<br>or<br>THERE ARE SOME EXTRA REQUIREMENTS ON THE REQUIREMENT LIST<br>followed by<br>THE STATED NUMBER OF REQUIREMENTS IS x<br>THE ACTUAL NUMBER OF REQUIREMENTS IS y | 4<br><br>4 |
| Terminator in the wrong place | THE REQUIREMENT LIST STARTS WITH A TERMINATOR<br>or<br>THE REQUIREMENT LIST IS TERMINATED IN THE MIDDLE OF A RECORD | 3<br><br>5 |

continued

Figure 5.4 continued

| SECTION OF DATA AND ERROR | MESSAGE OUTPUT | ACTION CODE (see below) |
|---|---|---|
| THE REQUIREMENT LIST | | |
| Incorrect tied requirements | TIE ERROR. TIE NAME IS x | 7 |
| Too many lessons in a tied set of requirements | THE TOTAL NUMBER OF LESSONS INVOLVED IN THE TIED SET WHICH STARTS WITH REQUIREMENT x IS TOO LARGE | 7 |
| THE FIXED FREE PERIOD LIST | | |
| Undefined item name | THE ITEM NAMED x DOES NOT APPEAR ON THE ITEM LIST | 2 |
| Undefined period name | THE PERIOD NAMED x DOES NOT APPEAR ON THE TIME STRUCTURE LIST | 2 |
| Too many fixed free periods for an item | THE ABOVE ITEM HAS TOO MANY FIXED FREE PERIODS | 7 |
| Terminator in the wrong place | THE FIXED FREE PERIOD LIST IS TERMINATED IN THE MIDDLE OF A RECORD | 7 |
| THE FORBIDDEN ASSIGNMENT LIST | | |
| Undefined requirement name | THE REQUIREMENT NAMED x DOES NOT APPEAR ON THE REQUIREMENT LIST | 2 |
| Undefined period name | THE PERIOD NAMED x DOES NOT APPEAR ON THE TIME STRUCTURE LIST | 2 |
| Terminator in the wrong place | THE FORBIDDEN ASSIGNMENT LIST IS TERMINATED IN THE MIDDLE OF A RECORD | 7 |
| THE PRE-ASSIGNMENT LIST | | |
| Undefined requirement name | THE REQUIREMENT NAMED x DOES NOT APPEAR ON THE REQUIREMENT LIST | 8 |
| Undefined period name | THE PERIOD NAMED x DOES NOT APPEAR ON THE TIME STRUCTURE LIST | 2 |
| Terminator in the wrong place | THE PRE-ASSIGNMENT LIST IS TERMINATED IN THE MIDDLE OF A RECORD | 7 |

continued

Figure 5.4 continued

ACTION CODES

1 - Continues as if the illegal number were not there.
2 - Continues with the incorrect quantity as zero.
3 - Skips to the next problem.
4 - Corrects the incorrect number for the remainder of the check.
5 - Continues with the information which it has found.
6 - Continues as if the extra data were not there.
7 - Continues.
8 - Omits the next number in the data and continues.

Figure 5.5

| DOCUMENT NAME AND SUGGESTED MANUAL CHECKS | COMMENTS |
|---|---|
| THE ITEM LIST | |
| Has each item been given the correct number of units? | The effort involved in these two checks is considerably reduced if all items of type 1 are listed first followed by all items of type 2 and finally all items of type 3. Any necessary re-arrangement can be done when the data cards have been punched. |
| Has each item been given the correct type code? | |
| THE TIME STRUCTURE LIST | |
| Have the period patterns been punched correctly? | The computer has only checked that each element is either 0 or 1 |
| THE CURRICULUM, THE TEACHER DUTY CHART AND THE ROOM AND EQUIPMENT USAGE CHART | |
| Do the computer versions of these documents agree with the hand written versions? | It is sufficient to check the 'TOTAL INTERNAL ACTIVITIES' or the 'TOTAL EXTERNAL ACTIVITIES' on the right hand side of the computer listing against the corresponding total on the hand written forms. Any error will cause at least two totals to be wrong. |
| THE REQUIREMENT LIST | |
| Are the period length and the number of lessons in each requirement the right way round? | Since it is the product of the period length and the number of lessons which appears on documents 3,4 and 5 it will not show up if they have been transposed. |
| THE FIXED FREE PERIOD LIST<br>THE FORBIDDEN ASSIGNMENT LIST<br>THE PRE-ASSIGNMENT LIST | |
| Does all the information on the hand written documents appear on the computer listing? | Since the computer has not been told how many records are on each of these documents it will not notice if some have been omitted. |

# CHAPTER SIX

## STORAGE AND DISPLAY OF INFORMATION

### 6.0 Introduction

As an aid to our later discussions we here record a possible method of storing within the computer and of displaying on the computer output the information relevant to an attempt to construct a time-table for a school which has been described by a Standard Data Deck. It is not implied that all the information is indispensable nor that the programs written by the Author store the information in exactly the way described.

In defining arrays we will use the ALGOL method of defining dimensions. Thus:-

i) ARRAY[$r_1$:$r_2$,$c_1$:$c_2$] defines a rectangular array with $r_2-r_1+1$ rows numbered from $r_1$ to $r_2$ and $c_2-c_1+1$ columns numbered from $c_1$ to $c_2$;

ii) ARRAY[$e_1$:$e_2$] defines vector with $e_2-e_1+1$ elements numbered from $e_1$ to $e_2$.

### 6.1 Single variables

The following variables represent the quantities indicated:

NI - the total number of items;

ND - the total number of days;

NP - the total number of periods;

LAP - the longest activity parameter;

SP - the space parameter;

NR - the total number of requirements;

NTS - NR plus the total number of tied sets;

TI - the total number of item names which appear on the Requirement List.

### 6.2 Arrays associated with days and periods

EDN[1:ND] - Element EDN[d] contains the external day name for the $d^{th}$ day i.e. the name defined on the Time Structure List. In what follows and within the computer each day is referenced by an internal day number which is simply d for the $d^{th}$ day.

EPN[1:NP] - Element EPN[p] contains the external period name for the $p^{th}$ period i.e. the name defined on the Time Structure List. In what follows and in the computer each period is referenced by an internal period number which is simply p for the $p^{th}$ period.

DAY[1:NP] - Element DAY[p] = d if period p falls on day d.

LPD[0:ND] - Element LPD[0] = 0 and LPD[d] contains the internal period number for the last period on day d.

6.3 Arrays associated with items

EIN[1:NI] - Element EIN[i] contains the external item name for the $i^{th}$ item i.e. the name defined on the Item List. In what follows and within the computer each item is referenced by an internal item number which is simply i for the $i^{th}$ item.

NOLIVES[1:NI] - Element NOLIVES[i] contains the total number of lives possessed by item i.

TYPE[1:NI] - Element TYPE[i] contains the type code for item i.

TEA[1:NI] - Element TEA[i] contains the total over the week of free periods which the units of item i may still have. Initially TEA[i] contains the quantity called 'Total External Activities' on the computer listing of documents 3, 4 and 5. This array is updated each time a free period is allocated.

PI[1:NP,1:NI] - Element PI[p,i] contains the number of units of item i which remain unemployed at period p. Initially PI[p,i] = NOLIVES[i]. This array is updated each time a unit is employed.

NUF[1:NP,1:NI] - Element NUF[p,i] contains the number of the units of item i which have been allocated free periods at period p. This array is updated each time a unit is allocated a free period.

IFM[1:NP,1:NI] - Element IFM[p,i] relates to period p and item i and at any stage of constructing the time-table it is in one of two possible states:-

available: IFM[p,i] is in this state if it is still possible for one or more further units from item i to be allocated

a free period at period p as part of at least one complete time-table. An available element contains a 1.

cancelled: IFM[p,i] is in this state if it is no longer possible for any unit from item i to be allocated a free period at period p as part of at least one complete time-table. A cancelled element contains a 0.

TPA[1:NP,1:NI] - Element TPA[p,i] contains the total number of possible internal activities in which the units of item i can still be employed at period p. The method of determining this number is described in 6.7. This array is updated each time a possible activity becomes impossible.

6.4 Arrays associated with requirements

ERN[1:NR] - Element ERN[r] contains the external requirement name of the $r^{th}$ requirement i.e. the name defined on the Requirement List. In what follows and within the computer each requirement is referenced by an internal requirement number which is simply r for the $r^{th}$ requirement.

TIE[1:NR] - Element TIE[r] = r if requirement r is not tied to any other requirement and TIE[r] = s if requirement r belongs to a tied set the next member of which has internal requirement number s.

SET[1:NR] - Element SET[r] = r if requirement r is not tied to any other requirement and SET[r] = NR + s if requirement r belongs to the $s^{th}$ tied set of requirements.

M[1:NR] - Element M[r] contains the length of the lessons involved in requirement r.

N[1:NR] - Element N[r] contains the number of lessons which remain to be assigned in requirement r. Initially N[r] contains the number defined on the Requirement List. This array is updated each time a lesson is assigned.

P[1:NR] - Element P[r] contains the priority parameter for requirement r. Methods of calculating this parameter are discussed in 10.3.

PR[1:NP,1:NR] - Element PR[p,r] is related to period p and requirement r and at any stage in the process of constructing

the time-table it is in one of two possible states:-

available: PR[p,r] is in this state if it is still possible for part of a lesson from requirement r to take place at period p and form part of at least one complete time-table.

cancelled: PR[p,r] is in this state if it is no longer possible for any part of a lesson from requirement r to take place at period p and form part of a complete time-table.

An available element contains a negative integer the value of which depends on the possible positions of lessons which could include period p. The value of the integer is defined by

$$PR[p,r] = -\sum_{s=1}^{M[r]} a_s . 2^{s-1}$$

where $a_s = 1$ if period p may be the $s^{th}$ period of a lesson and $a_s = 0$ otherwise.

Example

$PR[p,r] = -1 = -1_2$

means that period p may only be used as the first period of a lesson.

Example

$PR[p,r] = -2 = -10_2$

means that period p may only be used as the second period of a lesson.

Example

$PR[p,r] = -3 = -11_2$

means that period p may be used as the first or second period of a lesson.

Example

$PR[p,r] = -13 = -1101_2$

means that period p may be used as the first, third or fourth period of a lesson.

Initially every element in PR is available and the values are determined by the period patterns.

Example

If the period pattern for triple periods over a 9 period day is 1, 0, 0, 0, 0, 1, 1, 0, 0 then the column in array PR

for any requirement involving triple periods will initially appear as:-

| decimal | binary |
|---|---|
| -1 | -001 |
| -2 | -010 |
| -4 | -100 |
| 0 | 000 |
| 0 | 000 |
| -1 | -001 |
| -3 | -011 |
| -6 | -110 |
| -4 | -100 |

A cancelled element contains a positive integer the value of which depends on the stage in the time-table construction process at which the element became cancelled. If PR[p,r] = n it indicates that the cancellation took place at the stage in the process when n lessons had been assigned.

DCOUNT[1:ND,1:NTS] - If $r \leq NR$ then element DCOUNT[d,r] contains the number of possible positions for a lesson to be assigned on day d in requirement r. Its value, at any stage in the process, is the number of elements which contain a negative odd number in that part of column r of array PR which represents day d.

If r > NR then element DCOUNT[d,r] contains the total number of possible positions for a lesson to be assigned on day d in the requirements of tied set s, where s = r - NR. Its value at any stage in the process is given by

$$DCOUNT[d,r] = \sum_{t} DCOUNT[d,t]$$

where the summation is taken over all requirements in tied set s.

This array is updated each time an element in array PR is cancelled.

DMARK[1:ND,1:NTS] - If $r \leq NR$ then element DMARK[d,r] = 1 if at least one further lesson must be assigned on day d in requirement r and DMARK[d,r] = 0 otherwise.

If r > NR then element DMARK[d,r] = 1 if at least one

further lesson must be assigned on day d in a requirement of tied set s where s = r - NR and DMARK[d,r] = 0 otherwise.

This array is updated at any stage in the process when, due to cancellations in array PR, a lesson in any requirement or tied set of requirements becomes essential on any day.

6.5 Arrays associated with items and requirements

IR[1:TI], IRF[0:NR], RI[1:TI], RIF[0:NI].

It is necessary to record the items involved in each requirement and the requirements in which each item is involved. For the example problem the arrays which contain this information are shown in figures 6.1 and 6.2. Since the number of elements in a row of one of these arrays is variable it would be inefficient to store them in rectangular arrays. Each is therefore stored as a vector with a second vector containing the address of the final element in each row of the irregular array.

Thus IR[k] contains the internal item number corresponding to the $k^{th}$ external item name on the Requirement List.

IRF[0] = 0 and IRF[r] contains the number of the element in array IR which records the final unit involved in requirement r.

RI[k] contains the internal requirement number corresponding to the $k^{th}$ external requirement name on the re-arranged array.

RIF[0] = 0 and RIF[i] contains the number of the element in array RI which records the final requirement in which unit i is involved.

UU[1:NI,1:NR] - Element UU[i,r] contains the number of units of item i used in each lesson of requirement r. This array is not stored explicitly. The value of UU[i,r] can be found easily from either array IR or array RI.

6.6 Item bands

It is convenient to have a special name for the set of columns from array PR which involve units of a particular item, item i say. We therefore refer to such a set of columns as the Item Band for item i. The relevant columns are those listed in elements RIF[i-1] + 1 to RIF[i] of array RI. Since

Figure 6.1

| Req. No. (internal) | List of (internal) item numbers which are the elements of array IR | Elements of IRF |
|---|---|---|
| 0 | | 0 |
| 1 | 1 2 5 6 | 4 |
| 2 | 1 8 | 6 |
| 3 | 1 9 11 | 9 |
| 4 | 1 2 5 7 10 10 | 15 |
| 5 | 1 2 5 7 10 10 | 21 |
| 6 | 1 7 | 23 |
| 7 | 1 9 12 | 26 |
| 8 | 1 7 13 | 29 |
| 9 | 2 8 | 31 |
| 10 | 2 9 11 | 34 |
| 11 | 2 7 | 36 |
| 12 | 2 7 12 | 39 |
| 13 | 2 7 13 | 42 |
| 14 | 3 5 | 44 |
| 15 | 3 8 | 46 |
| 16 | 3 6 9 11 | 50 |
| 17 | 3 7 10 | 53 |
| 18 | 3 7 10 | 56 |
| 19 | 3 5 10 | 59 |
| 20 | 3 5 10 | 62 |
| 21 | 3 9 12 | 65 |
| 22 | 3 4 6 7 | 69 |
| 23 | 4 6 | 71 |
| 24 | 4 8 | 73 |
| 25 | 4 9 11 | 76 |
| 26 | 4 9 13 | 79 |
| 27 | 4 6 11 | 82 |
| 28 | 4 6 13 | 85 |
| 29 | 4 8 | 87 |
| 30 | 4 7 12 | 90 |

Figure 6.2

| Item No. (internal) | List of (internal) requirement numbers which are the elements of array RI | Elements of RIF |
|---|---|---|
| 0 | | 0 |
| 1 | 1 2 3 4 5 6 7 8 | 8 |
| 2 | 1 4 5 9 10 11 12 13 | 16 |
| 3 | 14 15 16 17 18 19 20 21 22 | 25 |
| 4 | 22 23 24 25 26 27 28 29 30 | 34 |
| 5 | 1 4 5 14 19 20 | 40 |
| 6 | 1 16 22 23 27 28 | 46 |
| 7 | 4 5 6 8 11 12 13 17 18 22 30 | 57 |
| 8 | 2 9 15 24 29 | 62 |
| 9 | 3 7 10 16 21 25 26 | 69 |
| 10 | 4 4 5 5 17 18 19 20 | 77 |
| 11 | 3 10 16 25 27 | 82 |
| 12 | 7 12 21 30 | 86 |
| 13 | 8 13 26 28 | 90 |

each requirement in general involves several different items each column from array PR occurs in several different item bands.

Example

From figure 6.2 we see that in the example problem the item band for item 1 consists of columns 1, 2, 3, 4, 5, 6, 7 and 8 while that for item 6 consists of columns 1, 16, 22, 23, 27 and 28. We note that column 1 occurs in both these item bands. It also occurs in the item bands for items 2 and 5 since items 1, 2, 5 and 6 are all involved in requirement 1.

## 6.7 Elements in TPA

As stated in 6.3, the value of element TPA[p,i] at any stage is the total number of possible internal activities in which the units of item i can still be employed at period p. This number is determined by examining the available elements at period p in the item band for item i in array PR. Account must be taken of complications due to tied sets of requirements. The relevant rules are:-

### 6.7.1 Rule A1

Let requirement r be an untied requirement in the item band

for item i.

Let k = UU[i,r] if PR[p,r] < 0 and k = 0 otherwise.

Then requirement r counts as k possible activities in the total TPA[p,i].

6.7.2 Rule A2

Let requirements $r_1$, $r_2$, .... $r_n$ be a set of tied requirements in the item band for item i where there may or may not be additional members of the set outside the item band.

Let k = $\max_j$ UU[i,$r_j$] where the evaluation of the maximum is taken over those members of the set $r_1$, $r_2$, ... $r_n$ for which PR[p,$r_j$] < 0.

Then the requirements $r_1$, $r_2$, .... $r_n$ count as k possible activities in the total TPA[p,i].

Example

Figure 6.3 shows day 1 of the item band for item 10 in the example problem with relevant parts of other arrays.

Figure 6.3

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 4 | 5 | 17 | 18 | 19 | 20 | IIN | 10 |
| | | ERN | 40 | 45 | 160 | 165 | 170 | 175 | EIN | 100 |
| | | TIE | 5 | 4 | 18 | 17 | 20 | 19 | | |
| | | SET | 31 | 31 | 34 | 34 | 35 | 35 | | |
| | | M | 2 | 1 | 2 | 1 | 2 | 1 | | |
| | | N | 2 | 2 | 2 | 1 | 2 | 1 | | |
| IIN | 10 | UU | 2 | 2 | 1 | 1 | 1 | 1 | | |
| IPN | 1 | PR | -1 | -1 | 3 | 3 | 3 | 3 | TPA | 2 |
| | 2 | | -3 | -1 | -1 | -1 | 4 | 4 | | 3 |
| | 3 | | -2 | -1 | -2 | -1 | 4 | 4 | | 3 |
| | 4 | | 6 | -1 | -1 | 6 | -1 | -1 | | 4 |
| | 5 | | 6 | -1 | -2 | 6 | -2 | -1 | | 4 |

At period 1 there are available elements in columns 4 and 5 of array PR each of which involves two units of item 10 but, since these columns form a tied set, the value of TPA[1,10] is 2 and not 4. Similarly, at periods 2 and 3, columns 4 and 5 contribute 2 possible activities and columns 17 and 18 contribute

1 giving the values of TPA[2,10] and TPA[3,10] as 3. Finally, at periods 5 and 6 column 5 contributes 2, column 17 contributes 1 and columns 19 and 20 combined contribute 1 making TPA[4,10] and TPA[5,10] each 4.

# CHAPTER SEVEN

## CANCELLATION RULES FOR ARRAY PR

### 7.0 Introduction

As a further preliminary to discussion of the main program we now look at the problems associated with keeping the array PR correctly updated.

### 7.1 The operation cancel PR(p,r)

As an aid to this description it is convenient to define an operation which changes an element, and in some cases associated elements, in array PR from the available state to the cancelled state. We define this operation, cancel PR(p,r), by the following ALGOL procedure:-

```
procedure cancel PR(p,r); value p,r; integer p,r;
begin integer u,v,w;
comment the arrays are assumed to have been declared in an
enclosing block;
L1:for u := p step -1 until p-M[r]+1 do
if (PR[u,r]+2)*2 ≠ PR[u,r] then goto L2;
L2:v := 1;
for w := u step 1 until u+M[r]-1 do
begin
  PR[w,r] := PR[w,r] + v;
  if PR[w,r] = 0 then PR[w,r] := n;
  comment where n is the total number of lessons so far
  assigned in the time-table;
  v := v + v
end;
if PR[p,r] < 0 then goto L1
end of procedure cancel;
```

Example

If M[r] = 4 and n = 6 then the result of cancel PR(2,r) on

| period | decimal | binary | | period | decimal | binary |
|---|---|---|---|---|---|---|
| 1 | -1 | -0001 | | 1 | 6 | - |
| 2 | -3 | -0011 | | 2 | 6 | - |
| 3 | -7 | -0111 | is | 3 | -1 | -0001 |
| 4 | -14 | -1110 | | 4 | -2 | -0010 |
| 5 | -12 | -1100 | | 5 | -4 | -0100 |
| 6 | -8 | -1000 | | 6 | -8 | -1000 |

Example

If M[r] = 4 and n = 6 then the result of cancel PR(3,r) on

| period | decimal | binary | | period | decimal | binary |
|---|---|---|---|---|---|---|
| 1 | -1 | -0001 | | 1 | 6 | - |
| 2 | -3 | -0011 | | 2 | 6 | - |
| 3 | -7 | -0111 | is | 3 | 6 | - |
| 4 | -14 | -1110 | | 4 | 6 | - |
| 5 | -12 | -1100 | | 5 | 6 | - |
| 6 | -8 | -1000 | | 6 | 6 | - |

7.2 To cancel or not to cancel

By definition, an element PR[p,r] should be in the cancelled state if it is no longer possible for any part of a lesson from requirement r to take place at period p and form part of a complete time-table. In practice, however, it is not always easy to see whether or not the occurrence of a given lesson or part of it at a given period can in fact form part of a complete solution. We are therefore forced to make the assumption that an element is available unless it can be seen that it should be cancelled.

Since, at any stage in the time-table construction process, the next lesson to be assigned will always be chosen from those elements which are in the available state it is most important that no element which should have been cancelled is in fact still available. If such an element is chosen for the next assignment then sooner or later a stage must be reached from which it is impossible to proceed to a complete solution. If, on the other hand, the element chosen for the next assignment is always a genuine available element then it will always be possible to

proceed to a complete solution. We therefore require a rigid set of rules which, at any given stage in the construction process, dictate which elements in array PR should be cancelled. These rules arise from the constraints which are built into the basic model e.g. the constraint that the lessons in any requirement or tied set of requirements must occur in the time-table with a normal distribution.

7.3 Some fairly obvious rules

Our first set of rules, Rules B1 to B6, results directly from consideration of operations on a single requirement or a tied set of requirements.

7.3.1 Rule B1

If element PR[p,r] is named on the Forbidden Assignment List and PR[p,r] < 0 then cancel PR(p,r) must be performed.

7.3.2 Rule B2

If a lesson from requirement r is assigned to commence at period p then cancel PR(p,r) must be performed.

7.3.3 Rule B3

If, by allocation of a free period or by assignment of a lesson involving it, a unit of item i is employed at period p leaving x units of that item unemployed, then cancel PR(p,r) must be performed for all values of r such that:-

i) requirement r involves more than x units of item i;
and ii) PR[p,r] < 0.

7.3.4 Rule B4

If a lesson from requirement r is assigned to commence at period p then cancel PR(s,t) must be performed for all values of s and t such that:-

i) t = r or SET[t] = SET[r]
and ii) PR[s,t] < 0
and iii) assignment of a lesson from requirement t to commence at period s would cause violation of the normal distribution constraint in requirement r or in any requirement tied to requirement r.

7.3.5 Rule B5

If a lesson from requirement r is assigned to commence at

Figure 7.1

| | | | | | |
|---|---|---|---|---|---|
| | | IRN | 16 | 17 | 18 |
| | | ERN | 160 | 170 | 180 |
| | | ITIE | 17 | 18 | 16 |
| | | SET | 64 | 64 | 64 |
| | | M | 2 | 1 | 1 |
| | | N | 1 | 2 | 1 |
| IPN | 1 | PR | -1 | 6 | -1 |
| | 2 | | -2 | 6 | -1 |
| | 3 | | -1 | 6 | -1 |
| | 4 | | -2 | 6 | -1 |
| | 5 | | -1 | 7 | -1 |
| | 6 | | -2 | 7 | -1 |
| | 7 | | -1 | 8 | -1 |
| | 8 | | -2 | -1 | -1 |
| | 9 | | -1 | -1 | -1 |
| | 10 | | -2 | -1 | -1 |
| | 11 | | -1 | -1 | -1 |
| | 12 | | -2 | -1 | -1 |
| | 13 | | -1 | -1 | -1 |
| | 14 | | -3 | -1 | -1 |
| | 15 | | -2 | -1 | -1 |
| DAY | 1 | DCOUNT | 2 | 0 | 4 |
| | 2 | | 2 | 1 | 4 |
| | 3 | | 2 | 4 | 4 |
| | 4 | | 2 | 3 | 3 |
| DAY | 1 | DMARK | 0 | 0 | 0 |
| | 2 | | 0 | 0 | 0 |
| | 3 | | 0 | 0 | 0 |
| | 4 | | 0 | 0 | 0 |

Figure 7.2

| | | | | |
|---|---|---|---|---|
| 16 | 17 | 18 | | |
| 160 | 170 | 180 | | |
| 17 | 18 | 16 | | |
| 64 | 64 | 64 | | |
| 2 | 1 | 1 | | |
| 1 | 2 | 1 | | |
| -1 | 6 | -1 | | |
| -2 | 6 | -1 | | |
| -1 | 6 | -1 | | |
| -2 | 6 | -1 | | |
| -1 | 7 | -1 | | |
| -2 | 7 | -1 | | |
| -1 | 8 | -1 | | |
| -2 | 9 | -1 | | |
| 9 | -1 | 9 | | |
| 9 | -1 | 9 | | |
| 9 | -1 | 9 | | |
| 9 | -1 | 9 | | |
| 9 | -1 | 9 | | |
| 9 | -1 | 9 | | |
| 9 | -1 | 9 | | |
| 2 | 0 | 4 | | |
| 2 | 0 | 4 | | |
| 0 | 4 | 0 | | |
| 0 | 3 | 0 | | |
| 0 | 0 | 0 | | |
| 0 | 0 | 0 | | |
| 0 | 1 | 0 | | |
| 0 | 1 | 0 | | |

period p and all other lessons in this requirement have previously been assigned then cancel PR(s,r) must be performed for all values of s for which PR[s,r] < 0.

7.3.6 Rule B6

If SET[r] ≠ r and cancel PR(p,r) is performed, causing DMARK[d,r] to change from 0 to 1 then cancel(s,t) must be performed for all s, t such that:-

i) SET[t] = SET[r] but t ≠ r

and ii) DAY[s] = d

and iii) PR[s,t] < 0.

Example

Figures 7.1 and 7.2 show the $4^{th}$ tied set of requirements from a problem (not the example problem) with a total of 60 requirements. In figure 7.1 PR[8,17] is available. If it is assumed that no lessons have been assigned in requirement 17 on day 1 or day 2 then figure 7.2 shows the set of elements which would be cancelled by application of Rule B6 following the operation cancel PR(8,17). This set of cancellations is necessary because, since the total number of lessons in the tied set is 4, the normal distribution constraint requires that one of them shall be assigned on each day. Thus as soon as it becomes essential for a lesson from requirement 17 to occur on each of days 3 and 4 it becomes impossible for a lesson from requirement 16 or requirement 18 to occur on these days.

7.4 Some less obvious rules

7.4.1 Rule B7

On day d of the item band for an item with one life let R be a set each member of which is:-

either i) a requirement in which at least one lesson must still be assigned on day d;

or ii) a tied set of requirements in which at least one lesson must still be assigned on day d in a requirement which is in the item band.

Let $R_j$ be a subset containing any or all of the members of R.

Corresponding to each $R_j$ let $P_j$ be the set of periods such that:-

i) each member is a period on day d;

Figure 7.3

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| | | ERN | 10 | 20 | 30 | 40 | 45 | 50 | 60 | 70 |
| | | TIE | 1 | 2 | 3 | 5 | 4 | 6 | 8 | 7 |
| | | SET | 1 | 2 | 3 | 31 | 31 | 6 | 32 | 32 |
| | | M | 1 | 1 | 1 | 2 | 1 | 2 | 1 | 1 |
| | | N | 5 | 5 | 5 | 2 | 2 | 1 | 1 | 1 |
| IPN | 1 | PR | -1 | -1 | 3 | 2 | 2 | 2 | 3 | 2 |
| | 2 | | -1 | -1 | -1 | 2 | 4 | 5 | 6 | 4 |
| | 3 | | 7 | -1 | -1 | 2 | 7 | 5 | 6 | -1 |
| | 4 | | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| | 5 | | -1 | -1 | -1 | -2 | -1 | -2 | -1 | -1 |
| DAY | 1 | DCOUNT | 4 | 5 | 4 | 1 | 2 | 1 | 2 | 3 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Figure 7.4

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | -1 | -1 | 3 | 2 | 2 | 2 | 3 | 2 |
| | 2 | | -1 | -1 | -1 | 2 | 4 | 5 | 6 | 4 |
| | 3 | | 7 | -1 | -1 | 2 | 7 | 5 | 6 | 8 |
| | 4 | | 8 | 8 | 8 | -1 | -1 | -1 | -1 | -1 |
| | 5 | | 8 | 8 | 8 | -2 | -1 | -2 | -1 | -1 |
| DAY | 1 | DCOUNT | 2 | 3 | 2 | 1 | 2 | 1 | 2 | 2 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Figure 7.5

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | -1 | -1 | 3 | 2 | 2 | 2 | 3 | 2 |
| | 2 | | -1 | -1 | 9 | 2 | 4 | 5 | 6 | 4 |
| | 3 | | 7 | 9 | -1 | 2 | 7 | 5 | 6 | 8 |
| | 4 | | 8 | 8 | 8 | -1 | -1 | -1 | -1 | -1 |
| | 5 | | 8 | 8 | 8 | -2 | -1 | -2 | -1 | -1 |
| DAY | 1 | DCOUNT | 2 | 2 | 1 | 1 | 2 | 1 | 2 | 2 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Note: In these and in later diagrams IPN stands for Internal Period Number and IRN for Internal Requirement Number. The other arrays are as defined in Chapter 6.

and ii) for each member the only available elements in the item band occur in the members of $R_j$.

Let $x_j$ be the total length of the lessons which must be assigned on day d in the members of $R_j$.

Let $y_j$ be the total number of periods in $P_j$.

Then, if $x_j = y_j$ for any $R_j, P_j$, each available element which is in a member of $R_j$ but not at a period in $P_j$ must be cancelled unless more than one lesson may be assigned in that member on day d.

Example

If, in figure 7.3, the operation cancel PR(3,8) is performed then the situation which remains gives rise to further cancellations under Rule B7. In this case the set R contains requirements 1, 2 and 3 and taking a subset $R_1$ to be the same 3 requirements then the corresponding set $P_1$ contains periods 1, 2 and 3. Thus $x_1 = 3$ and $y_1 = 3$ which means that the operations cancel PR(4,1), cancel PR(4,2), cancel PR(4,3), cancel PR(5,1), cancel PR(5,2) and cancel PR(5,3) must be performed. The item band after these operations is shown in figure 7.4.

Now if, as a result of operations elsewhere in the array, the operation cancel PR(2,3) is performed then again Rule B7 causes a further associated cancellation. As before the set R contains columns 1, 2 and 3 but if we now take the subset $R_1$ to be columns 1 and 2 then the corresponding set $P_1$ contains rows 1 and 2. Thus $x_1 = 2$ and $y_1 = 2$ which means that cancel PR(3,2) must be performed. The item band after these operations is shown in figure 7.5.

7.4.2 Rule B8

On day d of the item band for an item with one life let R be a set as defined in Rule B7.

For a given integer n let $R_n$ be the subset containing those members of R for which the number of available elements on day d is $\leq n$ where, if the member is a tied set of requirements, at most one element at each period may be counted.

Corresponding to each $R_n$ let $P_n$ be the set of periods such that

i) each member is a period on day d;

and ii) for each member at least one member of $R_n$ has an avail-

able element.

Let $x_n$ be the total length of the lessons which must be assigned on day d in the members of $R_n$.

Let $y_n$ be the total number of periods in $P_n$.

Let $z_n = y_n - x_n$.

Then each available element in the item band, which

i) occurs on day d in a requirement or tied set of requirements which is not a member of R.

and ii) if a lesson were assigned to commence at it, would employ the item for more than $z_n$ of the periods which are members of $P_n$,

must be cancelled.

Example

If, in figure 7.6, the operation cancel PR(4,2) is performed then the situation which remains gives rise to further cancellations under Rule B8. In this case R contains requirements 1, 2 and 3 and $R_5$ contains the same 3 requirements. The corresponding set $P_5$ contains periods 1, 2, 3 and 5. Thus $x_5 = 3$, $y_5 = 4$ and $z_5 = 1$ which means that cancel PR(1,4), cancel PR(2,4), cancel PR(1,6) and cancel PR(2,6) must be performed. The item band after these operations is shown in figure 7.7.

Now, if as a result of operations elsewhere in the array the operations cancel PR(5,2) and cancel PR(5,3) were performed then further cancellations would be dictated by Rule B8. Again R consists of requirements 1, 2 and 3 and $R_5$ contains the same 3 requirements. The corresponding set $P_5$ contains periods 1, 2 and 3. Thus $x_5 = 3$, $y_5 = 3$ and $z_5 = 0$ which means that cancel PR(1,5), cancel PR(2,5), cancel PR(3,5), cancel PR(1,7), cancel PR(2,7), cancel PR(3,7), cancel PR(1,8), cancel PR(2,8), and cancel PR(3,8) must be performed. The item band after these operations is shown in figure 7.8.

A further application of Rule B8 causes still further cancellations. Since requirements 7 and 8 form a tied set at most one lesson can be assigned in them on day 1 thus the set R now contains requirements 1, 2, 3 and the tied sets (4,5) and (7,8). Taking $R_5$, $R_4$, $R_3$ or $R_2$ each leads to a value of $z > 0$ but the set $R_1$ contains only the tied set (4,5) thus $P_1$ contains only period 4 and $x_1 = 1$, $y_1 = 1$, $z_1 = 0$ which means that cancel

Figure 7.6

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| | | ERN | 10 | 20 | 30 | 40 | 45 | 50 | 60 | 70 |
| | | TIE | 1 | 2 | 3 | 5 | 4 | 6 | 8 | 7 |
| | | SET | 1 | 2 | 3 | 31 | 31 | 6 | 32 | 32 |
| | | M | 1 | 1 | 1 | 2 | 1 | 2 | 1 | 1 |
| | | N | 5 | 5 | 5 | 2 | 2 | 1 | 1 | 1 |
| IPN | 1 | PR | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| | 2 | | -1 | -1 | -1 | -3 | -1 | -3 | -1 | -1 |
| | 3 | | -1 | -1 | -1 | -2 | -1 | -2 | -1 | -1 |
| | 4 | | 2 | -1 | 2 | 4 | -1 | 2 | -1 | -1 |
| | 5 | | 4 | -1 | -1 | 4 | 4 | 2 | -1 | -1 |
| DAY | 1 | DCOUNT | 3 | 5 | 4 | 2 | 4 | 2 | 5 | 5 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Figure 7.7

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | -1 | -1 | -1 | 5 | -1 | 5 | -1 | -1 |
| | 2 | | -1 | -1 | -1 | 5 | -1 | 5 | -1 | -1 |
| | 3 | | -1 | -1 | -1 | 5 | -1 | 5 | -1 | -1 |
| | 4 | | 2 | 5 | 2 | 4 | -1 | 2 | -1 | -1 |
| | 5 | | 4 | -1 | -1 | 4 | 4 | 2 | -1 | -1 |
| DAY | 1 | DCOUNT | 3 | 4 | 4 | 0 | 4 | 0 | 5 | 5 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Figure 7.8

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | -1 | -1 | -1 | 5 | 6 | 5 | 6 | 6 |
| | 2 | | -1 | -1 | -1 | 5 | 6 | 5 | 6 | 6 |
| | 3 | | -1 | -1 | -1 | 5 | 6 | 5 | 6 | 6 |
| | 4 | | 2 | 5 | 2 | 4 | -1 | 2 | -1 | -1 |
| | 5 | | 4 | 6 | 6 | 4 | 4 | 2 | -1 | -1 |
| DAY | 1 | DCOUNT | 3 | 3 | 3 | 0 | 1 | 0 | 2 | 2 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

Figure 7.9

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | -1 | -1 | -1 | 5 | 6 | 5 | 6 | 6 |
| | 2 | | -1 | -1 | -1 | 5 | 6 | 5 | 6 | 6 |
| | 3 | | -1 | -1 | -1 | 5 | 6 | 5 | 6 | 6 |
| | 4 | | 2 | 5 | 2 | 4 | -1 | 2 | 6 | 6 |
| | 5 | | 4 | 6 | 6 | 4 | 4 | 2 | -1 | -1 |
| DAY | 1 | DCOUNT | 3 | 3 | 3 | 0 | 1 | 0 | 1 | 1 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

PR(4,7) and cancel PR(4,8) must be performed. The item band after these operations is shown in figure 7.9.

7.4.3 Rule B9

Let R be a set containing 2 requirements $r_1$ and $r_2$ in the item band for a unit with one life.

Let D be the set containing those days d for which

i) DCOUNT[d,$r_1$] = DCOUNT[d,$r_2$] = 1

and ii) an available element in $r_1$ occurs at the same period as an available element in $r_2$.

Let x be the total number of days in D.

Let y be the total number of non-zero elements in the columns of DCOUNT corresponding to $r_1$ and $r_2$.

Let z = y - x.

Then if z = N[$r_1$] + N[$r_2$] the operation cancel PR(p,r) must be performed for all values of p, r such that

i) DAY[p] is a member of D,

ii) PR[p,$r_1$] < 0 and PR[p,$r_2$] < 0,

iii) r is in the same item band as $r_1$ and $r_2$ but $r \neq r_1$ and $r \neq r_2$

and iv) PR[p,r] < 0.

Example

If, in figure 7.10, after the cancellation of PR[19,16] and PR[19,22] Rule B9 is applied taking the set R as requirements 16 and 28 then D contains days 2 and 3. Thus x = 2, y = 8 and z = 6 which causes the operations cancel PR(8,22), cancel PR(8,23) and cancel PR(15,22). The item band after these operations is shown in figure 7.11.

Figure 7.10

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 16 | 22 | 23 | 27 | 28 | IIN | 6 |
| | | ERN | 10 | 150 | 190 | 200 | 230 | 235 | EIN | 60 |
| | | M | 1 | 1 | 1 | 1 | 2 | 1 | TYPE | 2 |
| | | N | 2 | 3 | 1 | 2 | 0 | 3 | NOLIVES | 1 |
| IPN | 1 | PR | -1 | -1 | -1 | -1 | 24 | -1 | PI | 1 |
| | 2 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 3 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 4 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 5 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 6 | | 7 | 7 | 7 | 7 | 7 | 7 | | 0 |
| | 7 | | 7 | 9 | 6 | 9 | 8 | 8 | | 0 |
| | 8 | | 7 | -1 | -1 | -1 | 8 | -1 | | 1 |
| | 9 | | 7 | 4 | -1 | -1 | 24 | 11 | | 1 |
| | 10 | | 7 | 4 | -1 | -1 | 24 | 10 | | 1 |
| | 11 | | 18 | 16 | -1 | 21 | 18 | 19 | | 1 |
| | 12 | | 18 | 18 | 18 | 18 | 18 | 18 | | 0 |
| | 13 | | 18 | 21 | 21 | 21 | 18 | 21 | | 0 |
| | 14 | | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |
| | 15 | | 18 | -1 | -1 | 21 | 0 | -1 | | 1 |
| | 16 | | 24 | 24 | 24 | 24 | 24 | 24 | | 0 |
| | 17 | | 24 | 24 | 24 | 24 | 24 | 24 | | 0 |
| | 18 | | 25 | 25 | 25 | 25 | 24 | 24 | | 0 |
| | 19 | | 25 | 26 | 26 | 26 | 0 | 24 | | 0 |
| | 20 | | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |
| | 21 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 22 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 23 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 24 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 25 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| DAY | 1 | DCOUNT | 5 | 5 | 5 | 5 | 0 | 5 | | |
| | 2 | | 0 | 1 | 3 | 3 | 0 | 1 | | |
| | 3 | | 0 | 1 | 2 | 0 | 0 | 1 | | |
| | 4 | | 0 | 0 | 0 | 0 | 0 | 0 | | |
| | 5 | | 5 | 5 | 5 | 5 | 0 | 5 | | |

Note: underlined elements denote assignments.

Figure 7.11

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 16 | 22 | 23 | 27 | 28 | IIN | 6 |
| | | ERN | 10 | 150 | 190 | 200 | 230 | 235 | EIN | 60 |
| | | M | 1 | 1 | 1 | 1 | 2 | 1 | TYPE | 2 |
| | | N | 2 | 3 | 1 | 2 | 0 | 3 | NCLIVES | 1 |
| IPN | 1 | PR | -1 | -1 | -1 | -1 | 24 | -1 | PI | 1 |
| | 2 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 3 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 4 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 5 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 6 | | 7 | 7 | 7 | 7 | 7 | 7 | | 0 |
| | 7 | | 7 | 9 | 6 | 9 | 8 | 8 | | 0 |
| | 8 | | 7 | -1 | 26 | 26 | 8 | -1 | | 1 |
| | 9 | | 7 | 4 | -1 | -1 | 24 | 11 | | 1 |
| | 10 | | 7 | 4 | -1 | -1 | 24 | 10 | | 1 |
| | 11 | | 18 | 16 | -1 | 21 | 18 | 19 | | 1 |
| | 12 | | 18 | 18 | 18 | 18 | 18 | 18 | | 0 |
| | 13 | | 18 | 21 | 21 | 21 | 18 | 21 | | 0 |
| | 14 | | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |
| | 15 | | 18 | -1 | 26 | 21 | 0 | -1 | | 1 |
| | 16 | | 24 | 24 | 24 | 24 | 24 | 24 | | 0 |
| | 17 | | 24 | 24 | 24 | 24 | 24 | 24 | | 0 |
| | 18 | | 25 | 25 | 25 | 25 | 24 | 24 | | 0 |
| | 19 | | 25 | 26 | 26 | 26 | 0 | 24 | | 0 |
| | 20 | | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |
| | 21 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 22 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 23 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 24 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| | 25 | | -1 | -1 | -1 | -1 | 24 | -1 | | 1 |
| DAY | 1 | DCOUNT | 5 | 5 | 5 | 5 | 0 | 5 | | |
| | 2 | | 0 | 1 | 2 | 2 | 0 | 1 | | |
| | 3 | | 0 | 1 | 1 | 0 | 0 | 1 | | |
| | 4 | | 0 | 0 | 0 | 0 | 0 | 0 | | |
| | 5 | | 5 | 5 | 5 | 5 | 0 | 5 | | |

Note: underlined elements denote assignments.

7.4.4 Rule B10

The operation cancel PR[p,r] must be performed for all periods p and all requirements r such that

i) assignment of a lesson or part of a lesson from requirement r to occur at period p would cause the number of units which may be given free periods at period p to become negative;

and ii) PR[p,r] < 0.

The statement of this rule assumes that it is possible to calculate the number of units which may be given free periods at any period p. In fact this information is not normally available.

To reveal the difficulties associated with the determination of the number of units which may be free at a given period consider a school which consists of nine items of which four are classes and five are teachers.

If all lessons involve one class and one teacher then, at a period when all four classes are being taught, there will be one teacher free while at a period when only three classes are being taught there will be one class and two teachers free.

If some lessons involve one class with two teachers then, at a period when all four classes are being taught and one such lesson occurs, there will be no teacher free.

If some lessons involve two classes with one teacher then, at a period when all four classes are being taught and one such lesson occurs, there will be two teachers free, unless one of the other classes uses two teachers.

The number of teachers free at period p therefore depends on

i) the total number of teachers less the total number of classes;

ii) the number of classes which are free at period p;

iii) the number of teachers involved with each class in the lessons which occur at period p.

This means that the total number of teachers who may be free at a given period is not known until any classes which are to have free periods have been given them and the lessons at that period have been assigned.

An exactly similar argument applies to the relationship

between class free periods and room free periods thus in our generalised model we cannot, at a stage when only some of the lessons at a given period have been assigned, calculate the exact number of units which can still be allocated free periods at that period. Application of Rule B10 is therefore difficult. The method used by the Main Program to resolve this difficulty is described in 10.7.3.

## 7.5 Generalisation of Rules B7, B8 and B9

Rules B7, B8 and B9 only deal with the item band for an item with one life. Clearly there should be generalisations of these rules for dealing with an item with several lives. To illustrate the complications which arise in such an item band consider figure 7.12. This figure shows one day from the item band for an item with 3 lives (not from the example problem) before any assignments have been made. The cancellations (indicated by zeroes) are necessary due to the possible positions of lessons in requirements 6, 7 and 9. Any rule which indicates that these cancellations should be made must take into account

i) the number of units still unemployed at each period;
ii) the lessons which must be assigned;
iii) the number of units of the item concerned involved in each requirement;
iv) the possible positions for the double and triple period lessons.

Clearly such a rule would be very difficult to write down and probably even more difficult to implement efficiently on a computer.

## 7.6 The perfect set of rules

No doubt additional rules could be devised to detect situations not dealt with by Rules B1 to B10 but the Author suggests that the production of a perfect set of rules is an impossible task. If such a set of rules exists then it would have the power to detect at once whether or not solution of a given problem was possible for if no solution exists then, by the definition of cancellation (6.4), every element in array PR would be cancelled. Time-table construction would then be considerably simplified for the algorithm defined by the flow

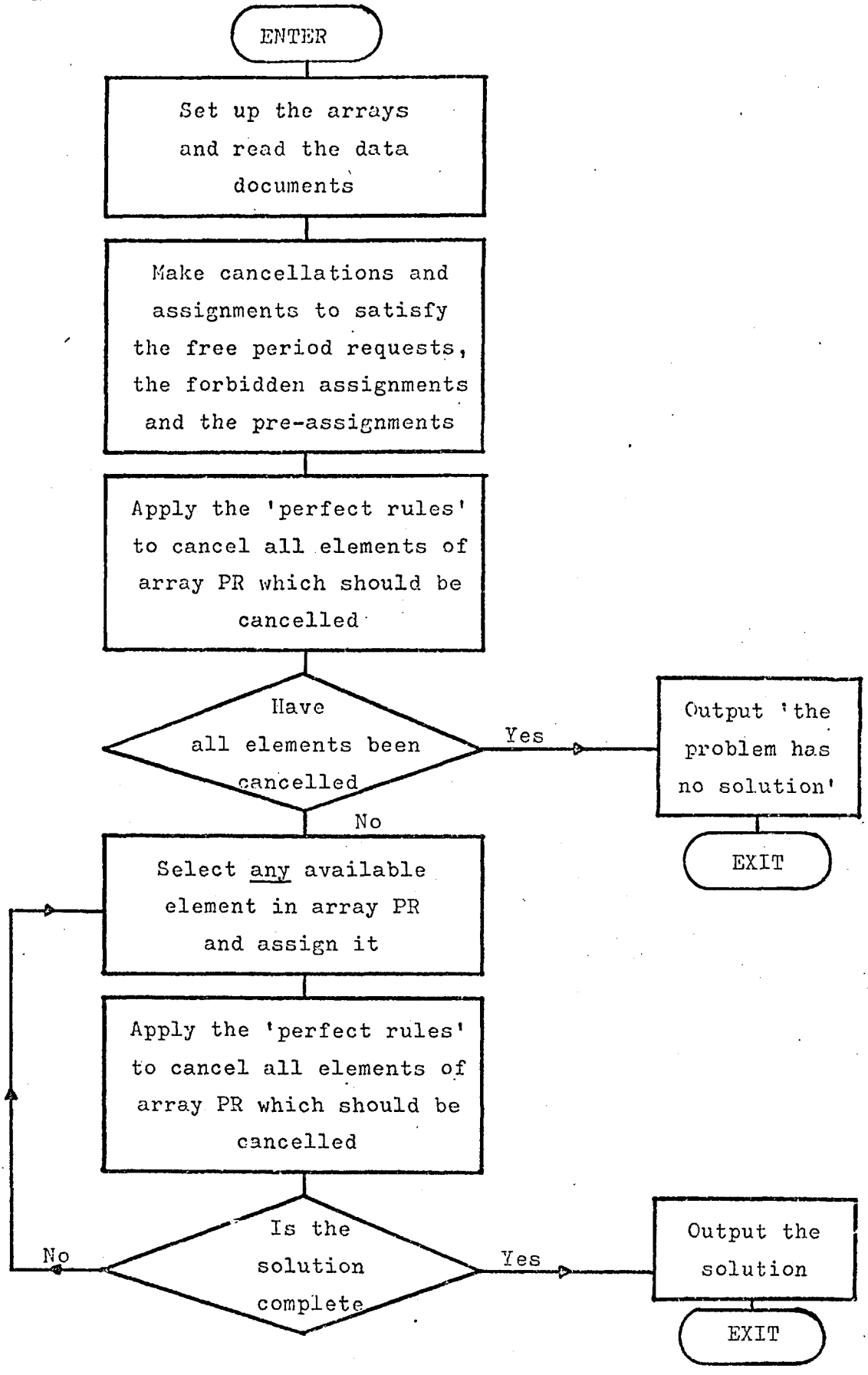
Figure 7.12

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | | M | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 3 |
| | | UU | 2 | 1 | 1 | 2 | 2 | 3 | 1 | 1 | 1 |
| IPN | 1 | PR | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| | 2 | | 0 | -1 | -1 | 0 | 0 | -3 | -3 | -3 | -2 |
| | 3 | | -1 | -1 | -1 | -1 | -1 | -2 | -2 | -2 | -4 |
| | 4 | | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| | 5 | | 0 | -1 | -1 | 0 | 0 | -3 | -3 | -3 | -2 |
| | 6 | | -1 | -1 | -1 | -1 | -1 | -2 | -2 | -2 | -4 |
| DAY | 1 | DMARK | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

diagram in figure 7.13 would either prove that no solution exists or would produce a solution in every case.

Because our system for detecting the elements which should be cancelled is certainly not perfect there may, at any stage, be elements in array PR which appear to be available but which in fact should be cancelled. This imperfection of the cancellation techniques means that any program using them must have some method of removing assignments which have led to an impossible situation. The Main Program described in Chapter 10 has such facilities.

Figure 7.13


ENTER
Set up the arrays and read the data documents
Make cancellations and assignments to satisfy the free period requests, the forbidden assignments and the pre-assignments
Apply the 'perfect rules' to cancel all elements of array PR which should be cancelled
Have all elements been cancelled
Yes
Output 'the problem has no solution'
EXIT
No
Select any available element in array PR and assign it
Apply the 'perfect rules' to cancel all elements of array PR which should be cancelled
Is the solution complete
No
Yes
Output the solution
EXIT

# CHAPTER EIGHT

## CANCELLATION RULES FOR ARRAY IFM

### 8.0 Introduction

As yet another preliminary to discussion of the main program we now examine the problems associated with keeping the array IFM correctly updated.

### 8.1 The operation cancel IFM(p,i)

Although the process involved is trivial it is convenient to define an operation which leaves an element in array IFM in the cancelled state. We define this operation, cancel IFM(p,i), by the following ALGOL procedure:-

```
procedure cancel IFM(p,i); value p,i; integer p,i;
begin comment the array is assumed to have been defined in an
              enclosing block;
  IFM[p,i] := 0
end
```

### 8.2 When to use cancel IFM(p,i)

By definition, an element IFM[p,i] should be in the cancelled state if it is no longer possible for any unit from item i to be allocated a free period at period p as part of a complete time-table. In practice, as with deciding whether an element in PR should be available or cancelled, it is not always easy to see whether or not the allocation of a free period to a given item at a given period can in fact form part of a complete solution thus, as with array PR, we must assume that a particular element is available unless it can be seen that it should be cancelled. We therefore require a set of rules which dictate when elements in IFM should be cancelled. Like Rules B1 to B10 these arise from the constraints which are built into the model.

#### 8.2.1 Rule C1

If TEA[i] becomes zero then cancel IFM(p,i) must be performed for all possible values of p.

In this and in the later rules we ignore the possibility that some of the elements to be cancelled may already be in the

cancelled state since computationally it is simpler to recancel an element than to determine its state and then cancel it only if it is in the available state.

Since many of the items can have no free periods TEA[i] will be zero for some values of i when it is first calculated and all elements in the corresponding columns of array IFM will be cancelled.

8.2.2 Rule C2

Let $X_{id} = \sum_{p} PI[p,i]$ where the summation is taken over periods on day d i.e. $X_{id}$ is the total number of units of item i which remain unemployed on day d.

Let $Y_{id} = \sum_{r} UU[i,r] * M[r]$ where the summation is taken over requirements in which lessons, not yet assigned, involving item i, must be assigned on day d. If n lessons must be assigned in a given requirement then that requirement is included in the summation n times. If a lesson must be assigned in a tied set of requirements then the requirement of the set which is included in the summation is that which has at least one available element on day d and uses the smallest number of units of item i.

Where one or more members of the tied set lie outside the item band for item i this may mean that no member of the set is included in the summation.

Then if $X_{id} = Y_{id}$, cancel IFM(p,i) must be performed for all values of p for which DAY[p] = d.

Example

Figure 8.1 shows day 1 of the item band for item 5 in the example problem and parts of other relevant arrays.

If cancel PR(3,14) were performed then item 5 would be forced to have a free period at period 3. For Rule C2 we then have $X_{51} = 2$ and $Y_{51} = 2$ thus cancel IFM(4,5) and cancel IFM(5,5) must be performed.

The resultant changes in the arrays, including cancellations caused by Rule B8, are shown in figure 8.2.

This rule applies to an item with any number of lives thus it is useful even though it overlaps with Rule C3 if the item has only one life.

Figure 8.1

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 4 | 5 | 14 | 19 | 20 | IIN | 5 | | |
| | | ERN | 10 | 40 | 45 | 130 | 170 | 175 | EIN | 50 | | |
| | | M | 1 | 2 | 1 | 1 | 2 | 1 | TYPE | 2 | | |
| | | N | 5 | 1 | 2 | 5 | 2 | 1 | NOLIVES | 1 | | |
| IPN | 1 | PR | 16 | 16 | 16 | 16 | 16 | 16 | PI | 0 | IFM | 0 |
| | 2 | | 16 | 16 | 16 | 16 | 16 | 16 | | 0 | | 0 |
| | 3 | | 17 | 16 | 16 | -1 | 16 | 17 | | 1 | | 1 |
| | 4 | | -1 | 16 | 16 | -1 | 16 | -1 | | 1 | | 1 |
| | 5 | | -1 | 16 | 16 | -1 | 16 | -1 | | 1 | | 1 |
| DAY | 1 | DCOUNT | 2 | 0 | 0 | 3 | 0 | 2 | | | | |
| DAY | 1 | DMARK | 1 | 0 | 0 | 1 | 0 | 0 | | | | |

Figure 8.2

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | 16 | 16 | 16 | 16 | 16 | 16 | PI | 0 | IFM | 0 |
| | 2 | | 16 | 16 | 16 | 16 | 16 | 16 | | 0 | | 0 |
| | 3 | | 17 | 16 | 16 | 18 | 16 | 17 | | 1 | | 0 |
| | 4 | | -1 | 16 | 16 | -1 | 16 | 18 | | 1 | | 0 |
| | 5 | | -1 | 16 | 16 | -1 | 16 | 18 | | 1 | | 0 |
| DAY | 1 | DCOUNT | 2 | 0 | 0 | 2 | 0 | 0 | | | | |
| DAY | 1 | DMARK | 1 | 0 | 0 | 1 | 0 | 0 | | | | |

Note: underlined elements denote assignments.

### 8.2.3 Rule C3

On day d of the item band for an item, item i, with one life let R, $R_n$, $P_n$, $x_n$, $y_n$ and $z_n$ be defined as in Rule B8. Then if $z_n = 0$ the operation cancel IFM(p,i) must be performed for all periods p which are members of $P_n$.

Example

Figure 8.3 shows the item band for item 8 in the example problem with parts of other relevant arrays.

If cancel PR(5,15) is performed then R contains requirements 2, 9, 15 and 24 and $R_4$ contains requirements 2, 9 and 15. The corresponding set $P_4$ contains periods 2, 3 and 4. Thus $x_4 = 3$, $y_4 = 3$ and $z_4 = 0$ which means that, in addition to cancellations

Figure 8.3

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 2 | 9 | 15 | 24 | 29 | IIN | 8 | |
| | | ERN | 20 | 80 | 140 | 210 | 240 | EIN | 80 | |
| | | M | 1 | 1 | 1 | 1 | 1 | TYPE | 2 | |
| | | N | 5 | 5 | 5 | 5 | 3 | NOLIVES | 1 | |
| IPN | 1 | PR | 6 | 7 | 8 | -1 | -1 | PI | 1 | IFM 1 |
| | 2 | | -1 | -1 | -1 | -1 | -1 | | 1 | 1 |
| | 3 | | -1 | -1 | -1 | -1 | -1 | | 1 | 1 |
| | 4 | | -1 | -1 | -1 | -1 | -1 | | 1 | 1 |
| | 5 | | 9 | 10 | -1 | -1 | -1 | | 1 | 1 |
| DAY | 1 | DCOUNT | 3 | 3 | 3 | 5 | 5 | | | |
| DAY | 1 | DMARK | 1 | 1 | 1 | 1 | 0 | | | |

Figure 8.4

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | 6 | 7 | 8 | -1 | -1 | PI | 1 | IFM 1 |
| | 2 | | -1 | -1 | -1 | 11 | 11 | | 1 | 0 |
| | 3 | | -1 | -1 | -1 | 11 | 11 | | 1 | 0 |
| | 4 | | -1 | -1 | -1 | 11 | 11 | | 1 | 0 |
| | 5 | | 9 | 10 | 11 | -1 | -1 | | 1 | 1 |
| DAY | 1 | DCOUNT | 3 | 3 | 3 | 2 | 2 | | | |
| DAY | 1 | DMARK | 1 | 1 | 1 | 1 | 0 | | | |

in PR caused by Rule B8, the operations cancel IFM(2,8), cancel IFM(3,8) and cancel IFM(3,8) must be performed. The resultant state of the arrays is shown in figure 8.4.

There should of course be a generalisation of this rule to deal with an item with several lives but the Author has been unable to devise a rule which takes into account all the relevant factors. Probably, as with additional rules for cancellations in PR, a general rule would be very complicated and either impossible or very inefficient to implement.

8.2.4 Rule C4

If at period p no further unit may be allocated a free period then cancel IFM(p,i) must be performed for all possible values of i.

As with Rule B10 (7.4.4) the statement of this rule assumes that it is possible to calculate the number of units which may be given free periods at any period p. The problems associated with calculating this quantity are discussed in 7.4.4 and the methods used by the Main Program are discussed in 10.7.3.

# CHAPTER NINE

## IMPOSSIBLE, FORCING AND TIGHT SITUATIONS

### 9.0 Introduction

In the time-table construction process applied by the Main Program methods of answering the following questions are needed:-

i) Has a stage been reached from which it is impossible to proceed to a complete solution?

ii) Are there any further assignments which must be made as a result of those which have already been made if a complete solution is to be obtained?

iii) What areas of the time-table have the most limited number of choices of assignment?

The purpose of this chapter is to provide a set of rules which enable these questions to be answered.

### 9.1 Definitions

#### 9.1.1 An impossible situation

We define an impossible situation to be a state in a partially constructed time-table from which it is impossible to proceed to a complete solution without violation of the constraints inherent in the model.

#### 9.1.2 A forcing situation

We define a forcing situation to be a state in a partially constructed time-table which uniquely determines at least one further assignment in the construction process.

#### 9.1.3 A tight situation

We define a tight situation to be a state in a partially constructed time-table which determines that all but one of a given set of assignments must be made.

### 9.2 Rules for locating special situations

#### 9.2.1 Rule D1

If IFM[p,i] = 0 then

i) if PI[p,i] > TPA[p,i] there is an impossible situation;

ii) if PI[p,i] = TPA[p,i] there is a forcing situation;
iii) if PI[p,i] = TPA[p,i] - 1 there is a tight situation.

Example

In figure 9.1 an impossible situation exists at period 2 since there is no way of employing item 1; a forcing situation exists at period 3 since item 1 must be employed on a lesson from requirement 3 and a tight situation exists at period 4 since item 1 must be employed on a lesson from either requirement 1 or requirement 2.

Figure 9.1

| | IRN | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ERN | 10 | 20 | 30 | 40 | 45 | 50 | 60 | 70 | | | | |
| | M | 1 | 1 | 1 | 2 | 1 | 2 | 1 | 1 | | | | |
| | N | 5 | 5 | 5 | 2 | 1 | 1 | 1 | 1 | | | | |
| | TIE | 1 | 2 | 3 | 5 | 4 | 6 | 8 | 7 | | | | |
| | SET | 1 | 2 | 3 | 31 | 31 | 6 | 32 | 32 | IIN 1 | | 1 | |
| IPN 1 | PR | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | IFM 0 | PI 0 | | TPA 0 |
| 2 | | 7 | 8 | 9 | 6 | 6 | 6 | 12 | 10 | 0 | | 1 | 0 |
| 3 | | 7 | 4 | -1 | 6 | 6 | 6 | 12 | 13 | 0 | | 1 | 1 |
| 4 | | -1 | -1 | 13 | 6 | 6 | 6 | 11 | 14 | 0 | | 1 | 2 |
| 5 | | -1 | -1 | 13 | 6 | 6 | 6 | -1 | -1 | 0 | | 1 | 3 |

9.2.2 Rule D2

If IFM[p,i] = 1 then

i) if PI[p,i] > TPA[p,i] there is a forcing situation;
ii) if PI[p,i] = TPA[p,i] there is a tight situation.

Example

In figure 9.2 a forcing situation exists at period 4 since the unit of item 5 must be assigned a free period and a tight situation exists at period 2 since either the unit of item 5 may be assigned a free period or it may be employed on a lesson from requirement 1.

Figure 9.2

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 4 | 5 | 14 | 19 | 20 | | | | | |
| | | ERN | 10 | 40 | 45 | 130 | 170 | 175 | | | | | |
| | | M | 1 | 2 | 1 | 1 | 2 | 1 | | | | | |
| | | N | 5 | 2 | 1 | 4 | 2 | 1 | | | | | |
| | | TIE | 1 | 5 | 4 | 14 | 20 | 19 | | | | | |
| | | SET | 1 | 31 | 31 | 14 | 35 | 35 | IIN | 5 | | 5 | 5 |
| IPN | 1 | PR | 6 | 6 | 6 | 6 | 6 | 6 | IFM | 0 | PI | 0 | TPA 0 |
| | 2 | | -1 | 6 | 6 | 15 | 10 | 17 | | 1 | | 1 | 1 |
| | 3 | | -1 | 6 | 6 | 15 | 10 | -1 | | 1 | | 1 | 2 |
| | 4 | | 8 | 6 | 6 | 15 | 15 | 12 | | 1 | | 1 | 0 |
| | 5 | | 15 | 6 | 6 | 15 | 15 | 15 | | 0 | | 0 | 0 |

Figure 9.3

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| | | ERN | 10 | 20 | 30 | 40 | 45 | 50 | 60 | 70 |
| | | M | 1 | 1 | 1 | 2 | 1 | 2 | 1 | 1 |
| | | N | 5 | 5 | 5 | 1 | 2 | 1 | 1 | 1 |
| IPN | 1 | PR | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| | 2 | | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| | 3 | | 5 | -1 | 6 | 4 | 4 | 4 | 4 | 4 |
| | 4 | | 5 | -1 | 6 | 4 | 4 | 4 | 4 | 4 |
| | 5 | | -1 | 5 | 5 | 4 | 4 | 4 | 4 | 4 |
| | | DCOUNT | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | DMARK | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

9.2.3 Rule D3

Let x, the total number of lessons which must be assigned on day d in requirement r, be > 0 then

i) if DCOUNT[d,r] < x there is an impossible situation;

ii) if DCOUNT[d,r] = x there is a forcing situation;

iii) if DCOUNT[d,r] = x + 1 there is a tight situation.

Example

In figure 9.3 an impossible situation exists in requirement 3

since there is no available period for the lesson which is essential; a forcing situation exists in requirement 1 since the lesson which is essential must be assigned at period 5 and a tight situation exists in requirement 2 since the lesson which is essential must be assigned either at period 3 or at period 4.

9.2.4 Rule D4

If DCOUNT[d,r] $\leq 1$ for all d and N[r] = $\sum_i$ DCOUNT[d,r]-1, where the summation is taken over all d, there is a tight situation.

Example

In figure 9.4 N[16] = 3 and $\sum$ DCOUNT[d,16] = 4 thus assignments in three of the fours available elements in column 16 of array PR must be made.

If NR < r $\leq$ NTS then Rule D4 applies to a tied set of requirements instead of to a single requirement.

9.2.5 Rule D5

Let $z_n$ be defined as in Rule B8 (7.4.2) then if $z_n < 0$ there is an impossible situation.

Example

If in figure 9.6 lessons in columns 4 and 20 become essential due to cancellations on other days of the item band then, when Rule B8 has been applied, the arrays are as shown in figure 9.7 and there is an impossible situation.

9.2.6 Rule D6

Let R, D and z be defined as in Rule B9 (7.4.3) then:

i) if $z < N[r_1] + N[r_2]$ there is an impossible situation;

ii) if $z = N[r_1] + N[r_2]$ there is a tight situation on each day which is a member of D;

iii) if $z = N[r_1] + N[r_2]$ there is a forcing situation on any day d for which either DCOUNT[d,$r_1$] = 1 and DCOUNT[d,$r_2$] = 0 or DCOUNT[d,$r_1$] = 0 and DCOUNT[d,$r_2$] = 1.

Example

If in figure 9.5 the operations cancel PR(15,16) and cancel PR(15,18) are performed then taking R to be the two requirements shown, which lie in the item band for item 6 in the example problem, then D contains days 2 and 3 and we get x = 1, y = 7

Figure 9.4

| | | |
|---|---|---|
| | IRN | 16 |
| | ERN | 150 |
| | M | 1 |
| | N | 3 |
| | TIE | 16 |
| | SET | 16 |
| IPN | 1 | 6 |
| | 2 | -1 |
| | 3 | 5 |
| | 4 | 3 |
| | 5 | 4 |
| | 6 | 12 |
| | 7 | 15 |
| | 8 | 13 |
| | 9 | 14 |
| | 10 | 14 |
| | 11 | 11 |
| | 12 | 11 |
| | 13 | -1 |
| | 14 | 17 |
| | 15 | 19 |
| | 16 | -1 |
| | 17 | 8 |
| | 18 | 8 |
| | 19 | 13 |
| | 20 | 14 |
| | 21 | 22 |
| | 22 | 23 |
| | 23 | 23 |
| | 24 | 15 |
| | 25 | -1 |

Figure 9.5

| | | | |
|---|---|---|---|
| | IRN | 16 | 28 |
| | ERN | 150 | 235 |
| | M | 1 | 1 |
| | N | 3 | 3 |
| | TIE | 16 | 27 |
| | SET | 16 | 37 |
| IPN | 1 PR | -1 | 13 |
| | 2 | -1 | 13 |
| | 3 | -1 | -1 |
| | 4 | 15 | 15 |
| | 5 | -1 | 16 |
| | 6 | 7 | 7 |
| | 7 | 9 | 8 |
| | 8 | -1 | -1 |
| | 9 | 4 | 11 |
| | 10 | 4 | 10 |
| | 11 | 16 | 19 |
| | 12 | 18 | 18 |
| | 13 | 20 | 21 |
| | 14 | 0 | 0 |
| | 15 | -1 | -1 |
| | 16 | 24 | 24 |
| | 17 | 24 | 24 |
| | 18 | 25 | 24 |
| | 19 | -1 | 24 |
| | 20 | 0 | 0 |
| | 21 | -1 | -1 |
| | 22 | -1 | -1 |
| | 23 | -1 | -1 |
| | 24 | -1 | -1 |
| | 25 | -1 | -1 |

Figure 9.6

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | IRN | 1 | 4 | 5 | 14 | 19 | 20 |
| | | ERN | 10 | 40 | 45 | 130 | 170 | 175 |
| | | M | 1 | 2 | 1 | 1 | 2 | 1 |
| | | N | 4 | 2 | 2 | 5 | 1 | 1 |
| | | TIE | 1 | 5 | 4 | 14 | 20 | 19 |
| | | SET | 1 | 31 | 31 | 14 | 35 | 35 |
| IPN | 1 | PR | 18 | 18 | 20 | 21 | 16 | 21 |
| | 2 | | 18 | 18 | 18 | 16 | 16 | 16 |
| | 3 | | 18 | 18 | -1 | -1 | 16 | -1 |
| | 4 | | 18 | -1 | -1 | -1 | -1 | -1 |
| | 5 | | 18 | -2 | -1 | -1 | -2 | -1 |
| DAY | 1 | DCOUNT | 0 | 1 | 3 | 3 | 1 | 3 |
| DAY | 1 | DMARK | 0 | 0 | 0 | 1 | 0 | 0 |

Figure 9.7

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| IPN | 1 | PR | 18 | 18 | 20 | 21 | 16 | 21 |
| | 2 | | 18 | 18 | 18 | 16 | 16 | 16 |
| | 3 | | 18 | 18 | 24 | -1 | 16 | -1 |
| | 4 | | 18 | -1 | 14 | -1 | 24 | -1 |
| | 5 | | 18 | -2 | 24 | -1 | 24 | -1 |
| DAY | 1 | DCOUNT | 0 | 1 | 0 | 3 | 0 | 3 |
| DAY | 1 | DMARK | 0 | 1 | 0 | 1 | 0 | 1 |

Note: underlined elements denote assignments.

and z = 6. Thus since N[16] + N[28] = 6 there is a tight situation on day 2 and an assignment must be made in PR[8,16] or in PR[8,28]. Also there is a forcing situation on day 4 since an assignment must be made in PR[19,16].

9.2.7 Rule D7

If the number of units which may be given free periods at period p becomes < 0 then there is an impossible situation.

The application of this rule suffers from the same difficulties as Rules B10 (7.4.4) and C4 (8.2.4).

Figure 9.8

| Classes | Possible teachers | | |
|---|---|---|---|
| C1 | T1 | T2 | T3 |
| C2 | T1 | T4 | T5 |
| C3 | T4 | T5 | |
| C4 | T4 | T5 | |

Example

Suppose that at a given period, (not in the example problem), all but 4 classes and 5 teachers have been employed and that the possible teachers for each class are as shown in figure 9.8 then if C1 is taught by T1, teachers T2 and T3 must have a free period thus an impossible situation would arise. In this example C2 must be taught by T1 if a solution is to be obtained.

# CHAPTER TEN

## THE MAIN PROGRAM

### 10.0 Introduction

We are now in a position to discuss the Main Program. To some extent it seeks to imitate the process carried out by a manual time-tabler.

The manual time-tabler starts off with a large blank time-table and sets about assigning lessons one after another until he has produced a solution or until he reaches a stage where he can see that it is not possible to produce a solution. In the latter case he changes some of the assignments already made or changes the problem by altering the requirements or the constraints to enable him to continue towards a solution. The method which the Main Program applies is similar to this except that we never allow the computer to alter the problem. We make a sequence of assignments and if a stage is reached where it can be seen that it is not possible to produce a solution we remove some of the assignments already made and make different ones.

There are two types of assignment. A forced assignment is one which is dictated by an entry on the Pre-assignment List or by the detection of a forcing situation (Chapter 9). Other assignments are chosen assignments.

In general, when a chosen assignment is made it will cause some forced assignments which in turn may cause other forced assignments hence producing a chain reaction. We call the chosen assignment and the set of forced assignments caused by it a chain. At the start of the process the pre-assignments and the assignments forced by them are called the initial chain.

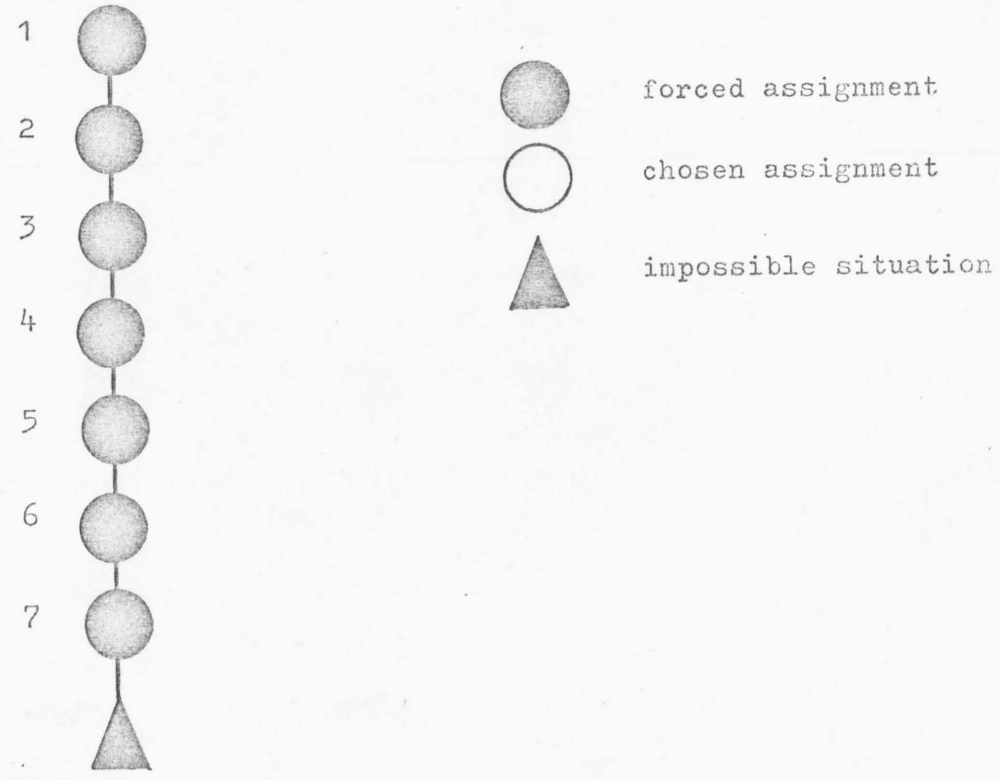
A chain may be terminated in one of two ways:-

i) by failure to detect any further forcing situations;

ii) by the detection of an impossible situation (Chapter 9).

In case i) we advance the solution by making a further chosen assignment. In case ii) we retreat by removing the current chain of assignments and then re-advance by extending the previous chain or making a new chosen assignment.

Figure 10.1




10.1 Diagrammatic representation of the problem

If we use a white circle to denote a chosen assignment, a black circle to denote a forced assignment and a black triangle to denote an impossible situation then the algorithm is represented diagrammatically by figures 10.1, 10.2 and 10.3.

In figure 10.1 the initial chain is terminated by the locating of an impossible situation. This means that the data is such that no solution exists. To produce a solution from such a set of data either the data must be changed or some of the constraints must be relaxed.

In figure 10.2 a complete solution is produced after 18 assignments but several retreats have been made. The initial chain (assignments 1-4) and chain 1 (assignments 5-6) are completed without trouble but the first retreat occurs after assignment 9 in chain 2. Assignments 7, 8 and 9 are therefore removed and a new chosen assignment made at assignment 7. Chain 2 can then be completed and chain 3 started but a retreat occurs after assignment 12. This time there is only one alternative to the original assignment 11 thus the new assignment 11 is a forced assignment and extends chain 2. This allows chain 2 to

Figure 10.2

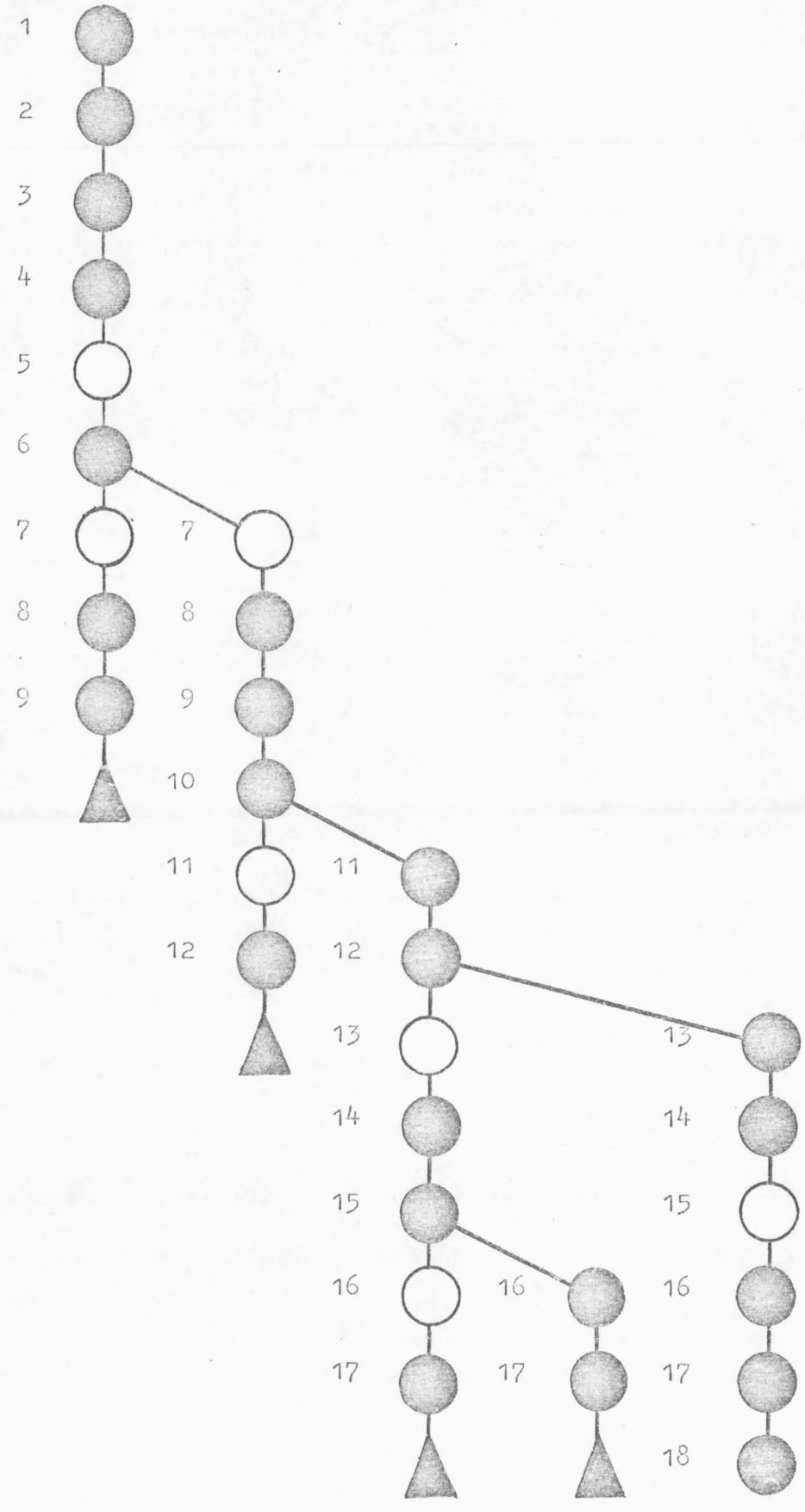
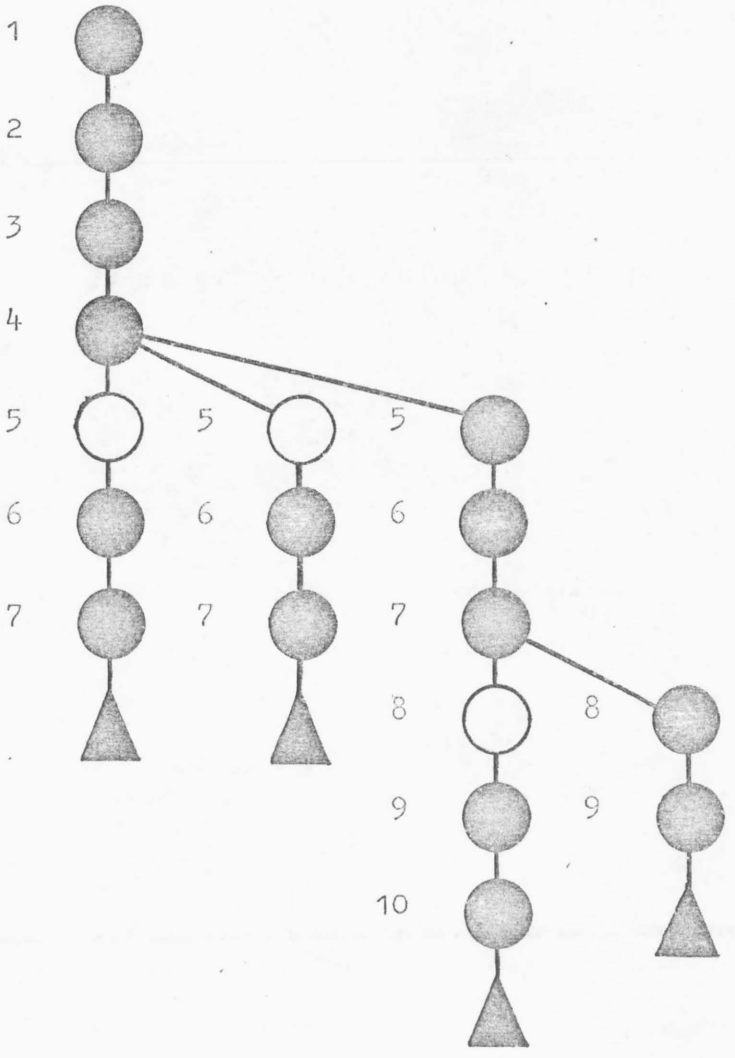


Figure 10.3




be completed at assignment 12 and chain 3 to be completed at assignment 15 but chain 4 runs into trouble at assignment 17. After a retreat to assignment 16 which results in an extension of chain 3 assignment 17 gives an impossible situation again. The whole of the extended chain 3 is therefore removed and assignment 13 made over again. This further extends chain 2 but allows the problem to be completed.

In figure 10.3 a similar advance and retreat situation is recorded but this time the process does not produce a complete solution. This is because the retreats have caused extensions to the initial chain and an impossible situation has been reached. As in the first example this shows that no solution exists.

If solutions exist for a given set of data then this pro-

cess must always lead to one of them providing an unlimited number of retreats is allowed. Similarly if no solution exists then the process must eventually lead to a failure in the initial chain.

The process will only be effective if the number of retreats which occur, either in producing a solution or in showing that no solution exists, is small compared with the total number of assignments made.

10.2 Chosen assignment strategy

In theory, we may at the end of any chain, select as the first assignment of the next chain any element PR[p,r] which contains a negative odd number since this indicates that a lesson from requirement r may still be assigned to start at period p. In practice, such a procedure would not be wise because, as we have discussed in 7.6, it is not possible to be absolutely certain that all elements which appear to be available are in fact available. If an element is chosen for assignment when it should have been cancelled during an earlier chain then sooner or later an impossible situation must be reached. If, between making the incorrect assignment and reaching this situation several complete chains have been assigned then it will, in general, be difficult to determine which of the assignments has caused the trouble and correction of the fault will only be achieved when the retreat-advance technique has removed all the intervening chains and caused the incorrect decision to be revised. Our aim is, therefore, to select assignments such that if one of them makes completion of a solution impossible then this fact can be detected quickly and if possible before any other chosen assignment is made. Thus, the chosen assignment strategy is to advance in that area of the time-table which is tightest i.e. where an impossible situation is most likely to arise.

10.3 The priority parameter

The main weapon in locating the tightest area in the time-table at a given stage in the problem is the detection of tight situations by Rules D1 to D6 (9.2) but an additional aid is required either when no such tight situation has been detected or when more than one has been detected and it is necessary to decide which of them is tightest. This aid is a priority

parameter associated with each requirement. The lessons in a requirement with a large parameter are assumed to be more difficult to assign than those of a requirement with a smaller parameter.

The priority parameter used in the Main Program is:-

$$P[r] = ((((t_1+t_2)*100+t_3)*100+t_4)*100+t_5)*100+t_6$$

where the terms $t_i$, i = 1(1)6 are defined in the following sections.

10.3.1 The teacher balance term $t_1$

Let $n_1$ and $n_2$ be the total number of units of type 1 and type 2 respectively involved in the lessons of requirement r then $t_1$ is defined by:-

$$t_1 = n_2 - n_1 \quad \text{if } n_1 < n_2$$
$$t_1 = 0 \quad \text{if } n_1 \geq n_2.$$

10.3.2 The room balance term $t_2$

Let $n_1$ and $n_3$ be the total number of units of type 1 and type 3 respectively involved in the lessons of requirement r then $t_2$ is defined by:-

$$t_2 = n_3 - n_1 \quad \text{if } n_1 < n_3$$
$$t_2 = 0 \quad \text{if } n_1 \geq n_3.$$

The inclusion of the terms $t_1$ and $t_2$ as the most significant terms in the priority parameter reflects the difficulties discussed in 7.4.4 in determining the number of units which may be assigned free periods at a given period. If a requirement involves more teachers than classes then assignment of a lesson in it at period p will reduce the total number of teachers who may have free periods at that period. It is important (see 10.7.3) that all such lessons should be assigned as early as possible in the construction process. This is achieved since for such a requirement $t_1$ is non-zero while for requirements involving the same number of classes as teachers or involving fewer teachers than classes it is zero. A similar argument applies to the balance between classes and rooms and the term $t_2$.

10.3.3 The 'busy' units term

We say that the units of item i are 'busy' if

(RIF[i] - RIF[i - 1] ÷ NOLIVES[i] > 3 * NP ÷ 5

i.e. if, allowing for the effects of integer division, the average

number of requirements in which a unit of the item is involved is greater than three-fifths of the number of periods in the week.

We then define $t_3$ to be the total number of 'busy' units involved in requirement r.

This term is included because experience has shown (11.3) that teachers and rooms which are involved with a large number of different classes during a week cause difficulties in a time-table. For example a physical training teacher who may meet 20 to 25 different classes and only take each for 1 or 2 periods will cause more problems than a Mathematics teacher who only meets 6 or 7 classes but takes each for 4 or 5 periods. The difficulties arise because the assignment of a lesson involving the physical training teacher will cause a large number of elements at the same period in array PR to be cancelled whereas the assignment of a lesson involving the Mathematics teacher causes relatively few cancellations.

10.3.4 The lesson length term $t_4$

We define $t_4 = M[r]$. This ensures that long lessons get priority over shorter ones.

10.3.5 The units involved term $t_5$

We define $t_5 = IRF[r] - IRF[r-1]$ i.e. the total number of units involved in requirement r. This term gives priority to setted lessons involving several classes, teachers and rooms over those which involve fewer units.

10.3.6 The tied requirement term $t_6$

We define $t_6 = 1$ if $r \neq TIE[r]$

$t_6 = 0$ if $r = TIE[r]$.

This gives priority to a tied requirement over one which is untied.

Example

Figure 10.4 shows the priority parameters for the example problem with the order of priority in which this places the requirements. A second order, based on the arrangement count, discussed in 13.3 is also shown. There is general agreement between the two orders for this problem.

Figure 10.4

| IRN | ERN | $t_1$ $t_2$ $t_3$ $t_4$ $t_5$ $t_6$ | A | B |
|---|---|---|---|---|
| 1 | 10 | 00 00 01 04 00 | 10 | 8 |
| 2 | 20 | 00 00 01 02 00 | 24 | 28 |
| 3 | 30 | 00 00 01 03 00 | 20 | 24 |
| 4 | 40 | 00 00 02 06 01 | 2 | 6 |
| 5 | 45 | 00 00 01 06 01 | 9 | 7 |
| 6 | 50 | 00 00 02 02 00 | 7 | 17 |
| 7 | 60 | 00 00 01 03 01 | 12 | 22 |
| 8 | 70 | 00 00 01 03 01 | 12 | 11 |
| 9 | 80 | 00 00 01 02 00 | 24 | 29 |
| 10 | 90 | 00 00 01 03 00 | 20 | 20 |
| 11 | 100 | 00 00 02 02 00 | 7 | 23 |
| 12 | 110 | 00 00 01 03 01 | 12 | 16 |
| 13 | 120 | 00 00 01 03 01 | 12 | 18 |
| 14 | 130 | 00 00 01 02 00 | 24 | 25 |
| 15 | 140 | 00 00 01 02 00 | 24 | 13 |
| 16 | 150 | 01 00 01 04 00 | 1 | 2 |
| 17 | 160 | 00 00 02 03 01 | 3 | 4 |
| 18 | 165 | 00 00 01 03 01 | 12 | 5 |
| 19 | 170 | 00 00 02 03 01 | 3 | 26 |
| 20 | 175 | 00 00 01 03 01 | 12 | 27 |
| 21 | 180 | 00 00 01 03 00 | 20 | 12 |
| 22 | 190 | 00 00 01 04 00 | 10 | 1 |
| 23 | 200 | 00 00 01 02 00 | 24 | 30 |
| 24 | 210 | 00 00 01 02 00 | 24 | 14 |
| 25 | 220 | 00 00 02 03 01 | 3 | 10 |
| 26 | 225 | 00 00 01 03 01 | 12 | 9 |
| 27 | 230 | 00 00 02 03 01 | 3 | 19 |
| 28 | 235 | 00 00 01 03 01 | 12 | 21 |
| 29 | 240 | 00 00 01 02 00 | 24 | 15 |
| 30 | 250 | 00 00 01 03 00 | 20 | 3 |

A = order based on the priority parameter.

B = order based on the arrangement count (13.3).

10.3.7 Comment on the priority parameter

The formula used for the priority parameter may be more complex than it need be or there may be factors, for example N[r], which should be included. Only carefully controlled experiments on a wide range of problems with different forms of parameter could determine a 'best' parameter. Such experiments have not been carried out since the faults which the program exhibits (see Chapters 11 and 12) are not likely to be greatly affected by further changes in the priority parameter.

10.4 The tightest element

Let $A_0(n)$ be the set of all available elements PR[p,r] in array PR at the stage in the construction process when n assignments have been made.

Let $A_1(n)$ be a subset of $A_0(n)$ each member of which is involved in at least one tight situation as detected by Rules D1 to D7 but which is not a member of $A_3(n)$ (see below).

Let $A_2(n)$ be a subset of $A_0(n)$ each member of which belongs to a requirement r for which P[r] > 20,000.

Let $A_3(n)$ be a subset of $A_0(n)$ each member of which is involved in at least one tight situation as detected by Rules D1 to D7 and satisfies at least one of the following conditions:-

i) is involved in a tight situation detected by Rule D3 with x = 1 where a similar tight situation exists in another requirement in the same item band and at the same two periods;
ii) is involved in a tight situation detected by Rule D4;
iii) is involved in a tight situation detected by Rule D6;
iv) has P[r] > 20,000.

Let $A_4(n)$ be a subset of $A_0(n)$ each member of which is involved in at least one forcing situation detected by Rule D1 D2, D3 or D6.

We are now in a position to state rules which define the tightest element in array PR at any stage in the construction process.

10.4.1 Rule E1

If $A_4(n)$ is not empty then the tightest element is taken to be the latest element in $A_4(n)$.

10.4.2 Rule E2

If $A_4(n)$ is empty and $A_3(n)$ is not empty then the tightest element is taken to be the latest element in $A_3(n)$.

10.4.3 Rule E3

If $A_4(n)$ and $A_3(n)$ are empty but $A_2(n)$ is not empty then the tightest is chosen as defined by Rule E5.

10.4.4 Rule E4

If $A_4(n)$, $A_3(n)$ and $A_2(n)$ are empty but $A_1(n)$ is not empty then the tightest element is taken to be that member PR[p,r] of $A_1(n)$ for which P[r] is greatest. In cases where a subset of $A_1(n)$ all have the same greatest P[r] the latest member of this subset is taken.

10.4.5 Rule E5

If $A_4(n)$ and $A_3(n)$ are empty and either $A_2(n)$ is not empty or $A_2(n)$ and $A_1(n)$ are both empty then the tightest element is taken to be a member PR[p,r] of $A_0(n)$ where period p occurs on day d and:-

i) if R is the set of requirements for which N[r] > 0 then r is that member of R for which P[r] is greatest; in cases of equality the member with smallest internal requirement number is chosen.

ii) If D is the set of days for which DCOUNT[d,r] > 0 then d is that member of D for which DCOUNT[d,r] is smallest; in cases of equality the member with smallest internal day number is taken.

iii) If an odd number of chosen assignments have been made in the time-table then p is the first available element on day d of requirement r.

and iv) If an even number of chosen assignments have been made in the time-table then p is the last available element on day d of requirement r.

10.4.6 Comments

The form of the above rules ensures that the following principles are observed:-

i) If a forcing situation has been found then the next assignment is a forced assignment. In this case Rule E1 applies.

ii) If no forcing situation has been found then the next assign-

Figure 10.5

| | | IRN | 4 | 6 | selected element | adjusted element |
|---|---|---|---|---|---|---|
| | | ERN | 40 | 50 | | |
| IPN | 1 | PR | -1 | -1 | PR[1,4] | PR[1,4] |
| | 2 | | -3 | -2 | PR[2,4] | PR[2,4] |
| | 3 | | -2 | 6 | PR[3,4] | PR[2,4] |
| | 4 | | -1 | -1 | PR[2,6] | PR[1,6] |
| | 5 | | -2 | -2 | | |

ment is a chosen assignment where priority is given first to special types of tight situation (those in $A_3(n)$), then to lessons with high priority parameters (those for which $t_1$, $t_2$ or $t_3 > 0$ or $t_4 > 1$) and finally to all other tight situations.

The alternative methods in Rule E5 of selecting the period, having determined the requirement and day, ensure that in the early stages of the construction process all assignments will not be made in the early periods of each day.

10.4.7 Adjustment

If the element selected as the tightest element by one of the above rules is PR[p,r] and M[r] > 1 then the position of the lesson to be assigned has not in general been uniquely determined. To make the determination unique an adjustment is carried out by applying the operation defined by the following ALGOL segment to select the period p1 at which the lesson will commence:-

```
for u := p step -1 until p - M[r] + 1 do
if (PR[u,r] ÷ 2) * 2 ≠ PR[u,r] then goto L1;
L1: p1 := u;
```

Example

Figure 10.5 shows a section of array PR for the example problem with a table of selected and adjusted elements.

10.5 The lists

As an aid to the implementation of the advance-retreat technique with the assignment strategy discussed in 10.2 a set of five lists are used. When any assignment is made these are used as described in the following sections.

10.5.1 The assignment list - List 0

As each assignment is made the following information is recorded, in packed form, on List 0:-

i) the selection code;

ii) the internal period number;

and iii) the internal requirement number;

where the selection codes are:-

0 - pre-assignment

1 - forced assignment (Rule E1)

2 - chosen assignment (Rule E2)

3 - chosen assignment (Rule E4)

4 - chosen assignment (Rule E5).

The length of the list is determined by the number of assignments required to complete the time-table i.e. $\sum_r N[r]$ where the summation is taken over all requirements.

One of the main uses of this list is the location of those assignments which must be removed when a retreat has to be made. This is possible since the chosen assignment which started the current chain of assignments at any stage is the latest entry on List 0 which has selection code greater than 1.

10.5.2 The forcing and tight situation lists - Lists 1,2 and 3

As the arrays are being updated following any assignment Rules D1 to D7 (9.2) are applied to locate the forcing and tight situations. When an element is found which is involved in one of these situations then the following information is recorded:-

i) the total number of assignments made;

ii) the internal period number;

and iii) the internal requirement number.

The number of assignments is included so that when a retreat is performed each list can be restored to almost the state it was in at the start of the chain which is being removed.

The lists are used as follows:-

List 1 - for all elements involved in forcing situations, i.e. the members of $A_4(n)$;

List 2 - for those elements involved in tight situations which belong to the subset $A_3(n)$;

List 3 - for those elements involved in tight situations which belong to the subset $A_2(n)$.

10.5.3 The day-band-check list - List 4

Because application of Rules B8 (7.4.2) and C3 (8.2.4) requires a relatively large amount of computation the number of times they are applied is reduced as far as possible. In theory they should be applied to day d of the item band for item i when any of the following changes occurs:-

i) item i is employed at a period on day d i.e is assigned to take part in a lesson or is given a free period;

ii) for a requirement r within the item band DMARK[d,r] changes from 0 to 1;

iii) an element PR[p,r], within the item band and for which DMARK[DAY[p],r] = 1, is cancelled.

Instead of applying the rules at once when one of these changes occurs an entry is made on List 4. The information stored is:-

i) the day on which the change has occurred;

and ii) the item band in which the change has occurred.

The method in which the entries on this list are used is shown in the flow diagram in figure 10.10.

10.5.4 List maintenance operations

The lengths of Lists 1, 2, 3 and 4 are chosen so that they will normally have room for all the necessary records but provision for dealing with over-flow is made. As safeguards against unnecessary over-flow the following steps are taken each time a new entry is made on a list:-

i) the new entry is checked against all previous entries and if it is found to be present already then it is not entered;

ii) in the cases of lists 1, 2 and 3 all previous entries are checked to make sure that the elements in array PR which they represent are still in the available state. Elements not in the available state are deleted from the list.

Some economy in execution time could probably be achieved by applying these rules less frequently, for example only when each chain has been completed.

10.6 Additional arrays

Because of the information needed for the currect method of controlling the allocation of free periods the Main Program requires three additional arrays. These were not discussed

in Chapter 6 as they are not yet regarded as permanent features of the program. As is discussed in 11.3 the control of free period allocation is not completely satisfactory and further improvements may be possible.

10.6.1 The balance count array

BC[2:3,1:NR] : Element BC[n,r] contains the number of units of type 1 less the number of units of type n involved in requirement r.

10.6.2 The free period count array

FPC[2:3,1:NP] : Element FPC[n,p] contains the current estimate of the total number of units of type n which may still be given a free period at period p. This element is incremented by 1 when any element of type 1 is given a free period at period p and is incremented by BC[n,r] when a lesson or part of a lesson from requirement r is assigned to period p.

10.6.3 The free period marker array

FPM[2:3,1:NP] : Element FPM[n,p] = 1 if it is still possible for either i) a further unit of type 1 to be given a free period at period p;

or ii) an assignment to be made at period p in a requirement r for which BC[n,r] > 0;

and FPM[n,p] = 0 otherwise.

10.7 Implementation of the various sets of rules

We have already defined the following sets of rules:-

Rules A1 - 2 (6.7) - for calculating elements in TPA;
Rules B1 - 10 (7.3 and 7.4) - for cancelling elements in PR;
Rules C1 - 4 (8.3) - for cancelling elements in IFM;
Rules D1 - 7 (9.2) - for locating impossible, forcing and tight situations;
Rules E1 - 5 (10.4) - for selecting the next assignment.

All these rules are implemented in the Main Program but some are subject to the limitations discussed in the following sections.

10.7.1 Restrictions involving distribution

It has been found necessary to restrict the total number of lessons which may be requested in a single requirement or tied set of requirements as follows:-

i) in a single requirement, $N[r] \leq ND$;

ii) in a tied set of requirements, $\sum N[r] \leq ND$ where the summation is taken over all members of the set.

A version of the program has also been written which allows i) to be replaced by:-

iii) in a single requirement, $N[r] \leq 2*ND$ if $M[r] = 1$

$N[r] \leq ND$ if $M[r] > 1$.

Even this modest easing of the restrictions considerably increases the difficulty in implementing several of the rules. If the full generality of the distribution rules implied in the definition of a normal distribution (2.1) were allowed the procedures used to apply some of the rules would become extremely complex with a corresponding increase in program size and decrease in execution efficiency. Since very few lessons occur more than once per day the restrictions imposed by i) and ii) do not prevent the program from dealing with data from real schools. Where say 6 single period lessons are required in a five day week, we simple put 5 of them in one requirement and the odd one in a second requirement with a related requirement name or number. The only disadvantage is that the odd lesson may be assigned to a period just before or just after a lesson from the first requirement hence producing an unwanted double period. This is a small price to pay for the simplifications made possible in the program.

10.7.2 Restrictions in Rule B7

A thorough application of Rule B7 (7.4.1) would require all possible sub-sets $R_j$ of the main set R to be considered. For example if R contains 5 members then there are:-

i) 1 set containing 5 members of R;

ii) 5 sets containing 4 members of R;

iii) 10 sets containing 3 members of R;

iv) 10 sets containing 2 members of R;

v) 5 sets containing 1 member of R;

giving a total of 31 sets to be checked. Much of the time spent on such a complete check would be wasted thus the implementation of Rule B7 only deals with sets containing 2 members of R.

10.7.3 Problems associated with Rules B10, C4 and D7

We have already indicated (7.4.4) why the application of these rules is difficult. The Main Program seeks to overcome these difficulties using the arrays BC, FPC and FPM.

To see how this part of the program operates we first note the following facts:-

i) The number of units of type2/type 3 free at period p depends on:-
   a) the total number of units of type 2/type 3 less the total number of classes;
   b) the number of classes free at period p;
   c) the number of units of type 2/type 3 involved with each class in the lessons which occur at period p (7.4.4).

ii) FPC[n,p] where n = 2 or 3 contains the current estimate of the total number of units of type n which may still be given free periods at period p (10.6.2) and must become zero when all lessons at period p have been assigned.

iii) If FPM[n,p] = 0 it indicates that FPC[n,p] cannot again increase, except possibly after a retreat. This means that FPC[n,p] can only decrease either due to the allocation of a free period at period p to a unit of type n or due to the assignment at period p of a lesson for which BC[n,r] < 0 (10.6.3).

Now to implement Rule B10 we want to answer the question: "If we assign a lesson from requirement r at period p will the number of units which may be given free periods at period p become negative?" If the answer to this question is "yes" then the operation cancel PR(p,r) will be performed provided PR[p,r] < 0. In order that no element will be cancelled when in fact it should be available we must therefore always answer "no" unless we can definitely see that the answer is "yes". The method used by the program to make this decision is represented in figure 10.6.

To implement Rule C4 we want to answer the question: "May any further unit be allocated a free period at period p?" The answer to this question is not always a simple "yes" or "no" but will in general be "yes, provided the unit to be given the free period is of a suitable type". To clarify the situation we

Figure 10.6

Diagram to answer the question: "If we assign a lesson from requirement r at period p will the number of units which may be given free periods at period p become negative?".

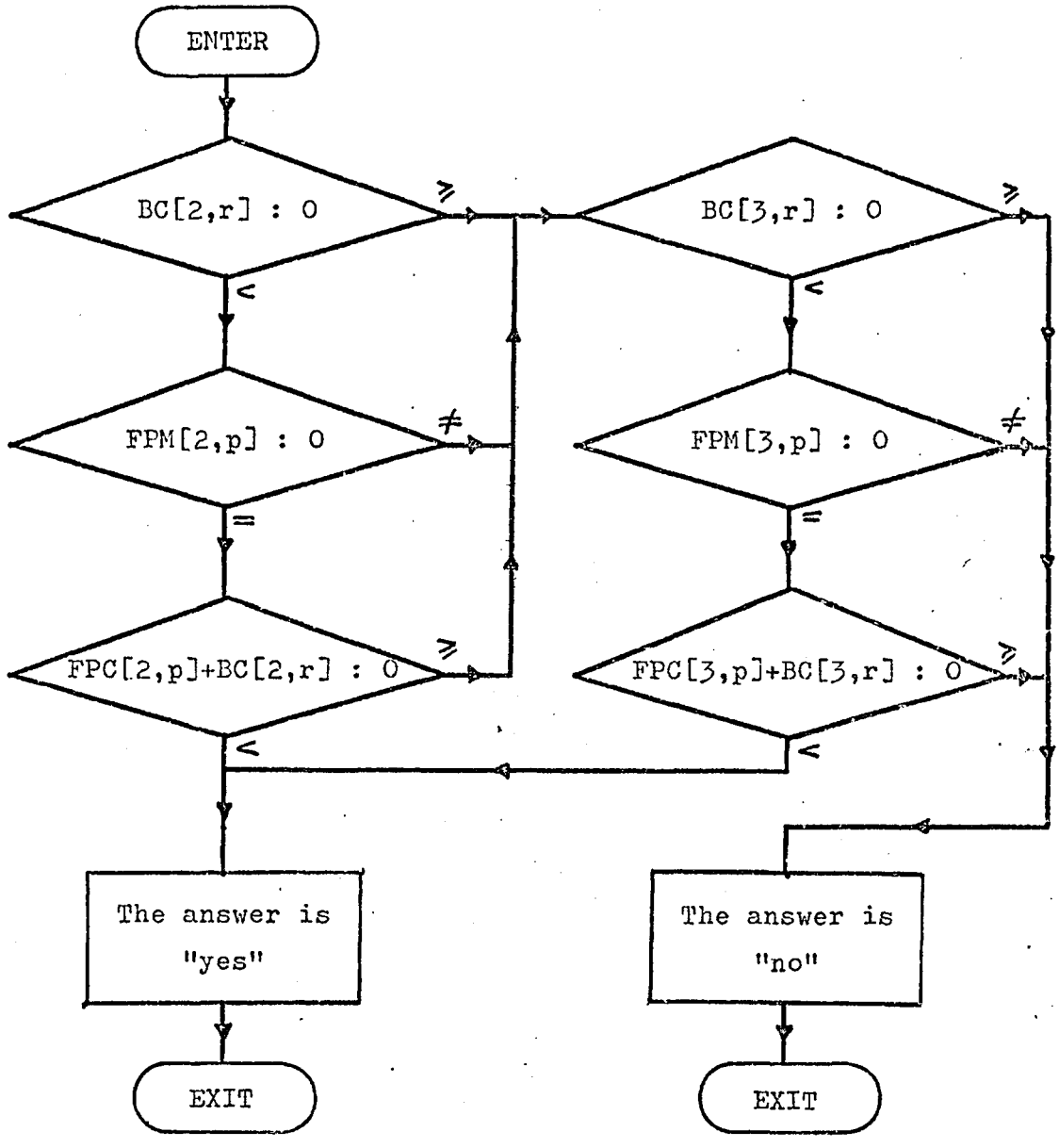
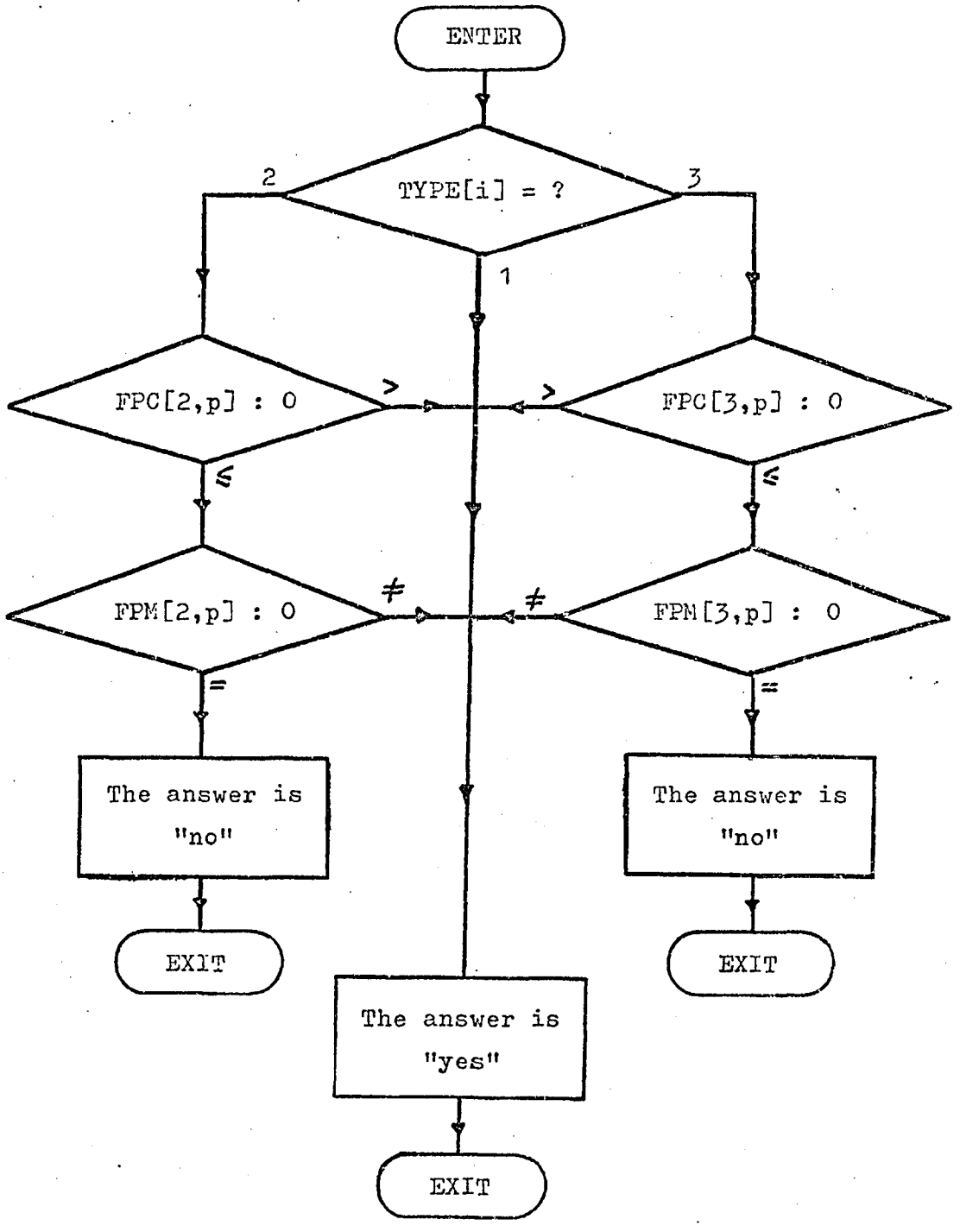


Figure 10.7

Diagram to answer the question: "As far as considerations at period p are concerned, may a unit of item i be allocated a free period at that period?".



rephrase the question to read: "As far as considerations at period p are concerned may a unit of item i be allocated a free period at that period?" If the answer to this question is "no" then the operation cancel IFM(p,i) will be performed. In order that no element will be cancelled when in fact it should be available we must always answer "yes" unless we can definitely see that the answer is "no". The method used by the program to make this decision is represented in figure 10.7.

The implementation of Rule D7 consists simply of examining FPC[2,p] when FPM[2,p] becomes zero and FPC[3,p] when FPM[3,p] becomes zero. If a negative value is ever found then an impossible situation exists.

These methods of implementing Rules B10, C4 and D7 are not perfect because at times they allow the assignment of lessons and allocation of free periods which cause elements in FPC to become negative without ensuring that these elements will later increase to zero again. If an element in FPC which is negative fails to increase to zero at period p then an impossible situation must eventually be located at that period. The presence in the priority parameter (10.3) of the term $t_1$ helps to avoid this problem by getting all the lessons for which BC[2,r] and/or BC[3,r] are/is negative assigned early in the construction process. A modification to the implementation of the rules, to make sure that no element of FPC would ever be allowed to become negative unless it was certain to increase to zero again, would be much more satisfactory but the Author has been unable to devise such a modification. It was thought about these problems which led to the investigation discussed in Chapter 13.

## 10.8 Retreating

### 10.8.1 The retreat marker

Although the necessity for a retreat may be detected during the updating process following a given assignment it is essential, due to the way that the updating is performed, that it should be completed before the retreat operation is commenced. When an impossible situation is located it is monitored (10.11) but instead of initiating the retreat at once a retreat marker, normally set equal to zero, is reset equal to unity.

If the process is in the initial chain then the retreat marker is not consulted until the initial chain has been completed. Its value then determines whether the run should be continued or terminated. This means that a series of impossible situations can all be detected at once.

If the process has passed the initial chain then the retreat marker is consulted at the end of each assignment and its value determines whether the process should advance or retreat.

### 10.8.2 The retreat count

Since, as discussed in 7.6, the methods of determining those elements in array PR which should be cancelled are imperfect the program from time to time assigns a lesson which should have been cancelled. This always causes an impossible situation sooner or later, If many chains have been inserted before the impossible situation is discovered the amount of computation required to work back to the fault and correct it may be excessive. It is therefore better to terminate the run and allow the analyst to locate and correct the difficulty. To this end a count of the total number of retreats made is kept and if this exceeds a pre-determined constant the run is terminated. Values of the constant between 5 and 20 have been used in development.

### 10.8.3 The retreat operation

When a retreat is required, and the retreat count has not been exceeded, List 0 is consulted to determine the last chosen assignment. If no entry with a selection code greater than 1 can be found then the program has proved that no solution exists. In this case, or when the retreat count has been exceeded, the current state of the time-table is printed out and the attempt to obtain a solution is terminated. Otherwise, let assignment n be the first assignment of the latest chain then all assignments which are members of this chain are removed from the time-table and all the arrays are restored to a state as near as possible to the state they were in when the chosen assignment at the start of the chain was selected. The main part of this process is to scan array PR and restore to the available state all elements which contain a number $\geq$ n updating (or back-dating) all the associated arrays as each element is dealt with.

The process restores TEA, PI, NUF, TPA, N, PR, DCOUNT, DMARK, FPC and FPM to exactly the state they were in when assignment n was first made. Apart from the lists, the only array not restored perfectly is IFM because, without doing a large amount of computation, it is not possible to find out which cancelled elements in this array should be restored to the available state. Since it is better to have an element which should be cancelled in the available state than to have an element which should be available in the cnacelled state the operation defined by the following ALGOL segment is performed when all the other arrays have been restored:-

```
for a := 1 step 1 until NI do
if TEA[a] > 0 then
for b := 1 step 1 until NP do
if FPM[TYPE[a],b] = 1 then IFM[b,a] := 1;
```

Any elements wrongly set by this operation are normally quickly corrected when the process advances again.

Not all the lists can be restored to their correct state since some elements may have been removed from them by the list maintenance operations (10.5.3) during the chain which has had to be removed. The List 0 pointer is set back to n-1 i.e. to the last assignment of the previous chain. Since List 1 must have been empty at the start of the chain (otherwise the previous chain would have continued) its pointer is simply put back to zero. For Lists 2 and 3 the pointers are moved back until a record is found in which the assignment number is less than n. The pointer on List 4 is also put back to zero.

When these operations have been completed the retreat marker and the assignment counter are reset equal to zero and n-1 respectively.

The final step of the retreat operation is to alter array PR so that the same chosen assignment cannot be chosen again. If the assignment was at period p and in requirement r and M[r] = 1 this means that cancel PR(p,r) is performed but if M[r] > 1 then only the possibility of a lesson from requirement r starting at period p must be removed. Because in the majority of cases a chosen assignment is selected from the entries on List 2 or List 3 which contain elements involved in tight situations,

this final step usually causes an entry to be made on List 1 hence causing an extension of the previous chain rather than the selection of an alternative chosen assignment.

10.9 The overall structure

We have now described all the most important aspects of the Main Program and it remains to show how they are built into an algorithm.

Figure 10.10 is a flow diagram of the main control process but since the details of the program are relatively complex we do not attempt to give detailed flow diagrams of all its sections. The operations "initialise" and "assign" are as shown in figures 10.8 and 10.9.

10.10 Input data

The Main Program accepts data in exactly the same form as the Data Checking Routine (5.1.1).

10.11 Output

The output from the program consists of two parts:-

i) monitoring of the construction process;

and ii) the current state of the main arrays at the end of a run and the time-table if it has been completed.

The monitoring gives the following information:-

i) The time elapsed from the start of the run, measured in ATLAS 'INSTRUCTIONS' each of which is equivalent to 2048 basic operations, at the start of each problem, at the end of the initialisation stage, when a retreat starts and ends and on termination of each problem.

ii) The assignment number, the selection code, the external period name and the external requirement name for each assignment as it is made.

iii) Messages indicating the occurrence of impossible situations and other events. A full list of these, for the version of the program which accepts numeric data, is given in figure 10.11.

The output of the main arrays is as shown in figures 10.12 and 10.13. In order to reduce the amount of data printed these are not output in exactly the forms described in Chapter 6. The main change is the packing of the arrays PI, NUF, IFM and TPA. The number actually printed in the composite array

as element (p,i) is:-

((PI[p,i] * 100 + NUF[p,i]) * 100 + TPA[p,i]) * 10 + IFM[p,i]

Since these arrays are normally only used by the analyst to investigate a failure to complete the problem, their relatively complex form is not a disadvantage.

If the time-table has been completed it is output in the form shown in figure 2.6.

10.12 Comment

This program has evolved over a period of about five years during which it has undergone almost constant revision. The aim has been to produce a program which would use the retreat facility as infrequently as possible and which would either obtain a solution to each problem at its first attempt or reach a stage from which the analyst could see that no solution existed. The extent to which this objective has been achieved is discussed in Chapters 11 and 12.

Figure 10.8




ENTER

Define arrays of the appropriate size and, having read Documents 1, 2 and 6 initialise them.

Read Document 7, allocate free periods to the units listed and update the arrays without applying Rules B8 and C3.

Read Document 8, cancel, in array PR, the elements listed and update the other arrays without applying Rules B8 and C3.

Read Document 9, make the assignments listed and update the arrays without applying Rules B8 and C3.

Apply Rules B8 and C3 to each day of each item band and search the complete arrays for impossible, forcing and tight situations.

Complete the initial chain using the operation 'assign'.

EXIT

Figure 10.9




ENTER

Make the selected assignment; update the arrays without applying Rules B8 and C3; record on List 4 those item bands to which Rules B8 and C3 should be applied; allocate free periods to units which have become unemployable; search for impossible, forcing and tight situations.

EXIT

Figure 10.10

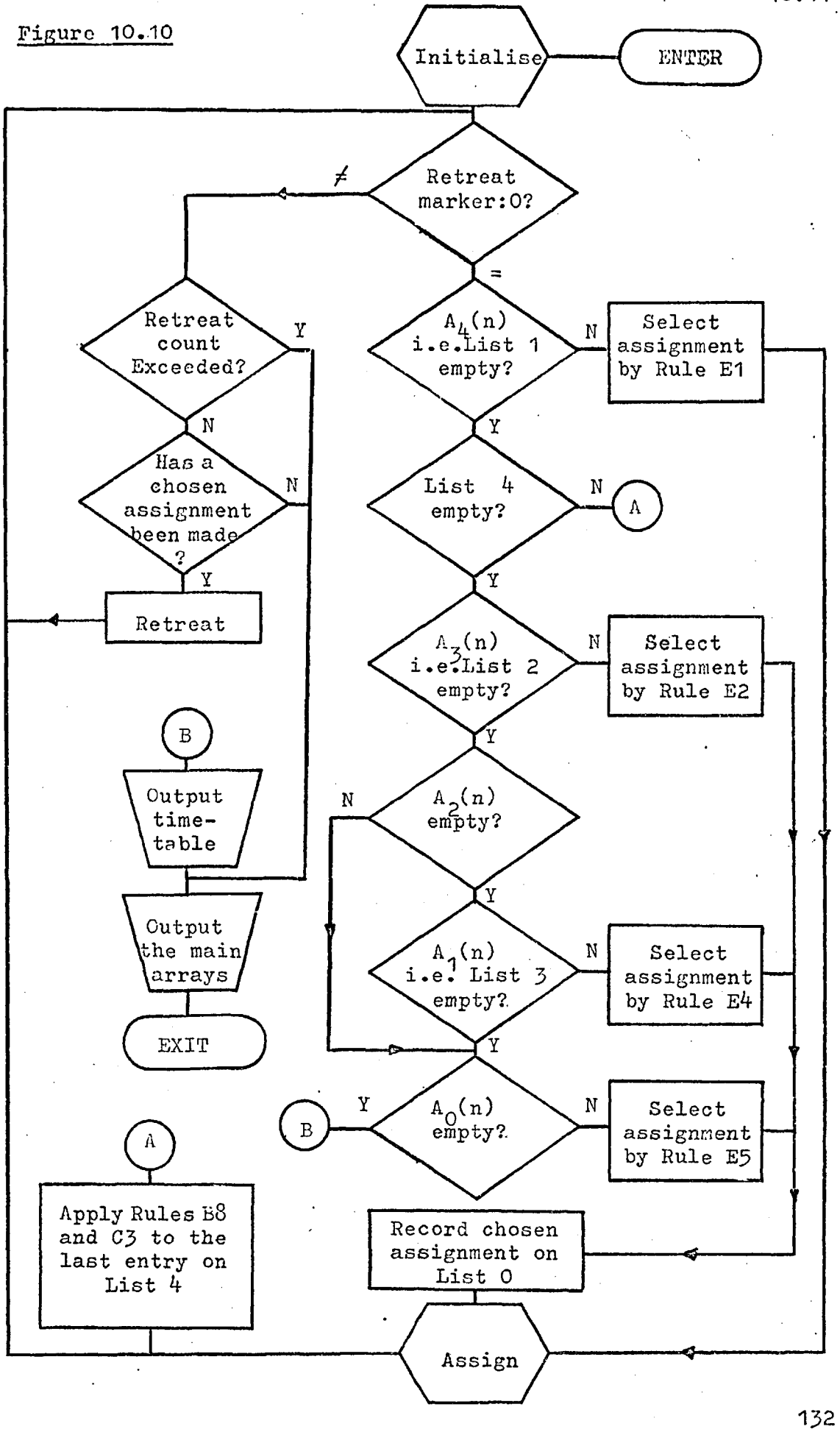

Initialise
ENTER
≠
Retreat marker:0?
=
Retreat count Exceeded?
Y
N
Has a chosen assignment been made ?
N
Y
Retreat
$A_4(n)$ i.e.List 1 empty?
N
Select assignment by Rule E1
Y
List 4 empty?
N
A
Y
$A_3(n)$ i.e.List 2 empty?
N
Select assignment by Rule E2
Y
$A_2(n)$ empty?
N
Y
$A_1(n)$ i.e. List 3 empty?
N
Select assignment by Rule E4
Y
$A_0(n)$ empty?
Y
B
N
Select assignment by Rule E5
B
Output time-table
Output the main arrays
EXIT
A
Apply Rules B8 and C3 to the last entry on List 4
Record chosen assignment on List 0
Assign

Figure 10.11

| ERROR | MESSAGE OUTPUT | ACTION CODE (see below) |
|---|---|---|
| Undefined name for item, period or requirement appears on data document | ILLEGAL NAME x ON DOCUMENT NUMBER y. | 1 |
| Incomplete tie cycle | TIE CYCLE ERROR INVOLVING REQUIREMENT x. | 1 |
| Impossible situations: | | |
| detected by Rule D1 | RULE D1 FAILURE: PERIOD x, ITEM y. | 2 |
| detected by Rule D3 | RULE D3 FAILURE: DAY x, REQUIREMENT y. | 2 |
| detected by Rule D5 | RULE D5 FAILURE: DAY x, ITEM y. | 2 |
| detected by Rule D6 | RULE D6 FAILURE: REQUIREMENTS x,y. | 2 |
| detected by Rule D7 | RULE D7 FAILURE: PERIOD x. | 2 |
| Pre-assignment requested at a time when it cannot occur | A PRE-ASSIGNMENT AT PERIOD x IN REQUIREMENT y IS IMPOSSIBLE. IT HAS BEEN OMITTED. | 3 |
| List overflow | LIST x HAS OVERFLOWED | 4 |
| Too many retreats | THE RETREAT RESTRICTION HAS CAUSED TERMINATION | 5 |

Action codes

1 - Continue to the end of the initialisation phase, output the main arrays and then skip to the next problem.

2 - If this occurs in the initialisation phase action is as for code 1 otherwise the retreat marker is set equal to 1.

3 - Continue without making the pre-assignment.

4 - Delete the second half of the list concerned and continue. (This error normally occurs only on Lists 3 and 4.)

5 - Output the current state of the main arrays and skip to the next problem.

Figure 10.12

Note: This figure shows part of a page of the item arrays as they are printed out by the Main Program.

ITEM ARRAYS

| IIN | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|
| EIN | 40 | 50 | 60 | 70 | 80 | 90 |
| NOLIVES | 1 | 1 | 1 | 1 | 1 | 1 |
| TYPE | 1 | 2 | 2 | 2 | 2 | 2 |
| TEA | 0 | 4 | 3 | 2 | 1 | 4 |

| EPN | PI, NUF, TPA, IFM | | | | | |
|---|---|---|---|---|---|---|
| 11 | 100050 | 100011 | 100031 | 0 | 100041 | 100041 |
| 12 | 100050 | 100011 | 100031 | 0 | 100041 | 100041 |
| 13 | 100030 | 100011 | 100031 | 100021 | 0 | 100041 |
| 14 | 100050 | 0 | 100021 | 100021 | 100041 | 100041 |
| 15 | 100020 | 0 | 0 | 0 | 100021 | 0 |
| 21 | 100040 | 100021 | 100021 | 0 | 100031 | 0 |
| 22 | 100030 | 100021 | 100021 | 0 | 0 | 100021 |
| 23 | 100030 | 0 | 0 | 1000 | 100031 | 100020 |
| 24 | 100040 | 100021 | 100021 | 0 | 100031 | 0 |
| 25 | 100030 | 100021 | 100021 | 0 | 0 | 100021 |
| 31 | 0 | 0 | 0 | 0 | 0 | 0 |
| 32 | 0 | 0 | 1000 | 0 | 0 | 0 |
| 33 | 0 | 0 | 0 | 0 | 1000 | 0 |
| 34 | 0 | 0 | 1000 | 0 | 0 | 0 |
| 35 | 0 | 0 | 0 | 1000 | 0 | 0 |
| 41 | 100040 | 0 | 100020 | 1000 | 100040 | 100020 |
| 42 | 100040 | 100011 | 100031 | 0 | 100041 | 100021 |
| 43 | 100040 | 100011 | 100031 | 0 | 100041 | 100021 |
| 44 | 100020 | 0 | 0 | 0 | 100020 | 0 |
| 45 | 0 | 0 | 1000 | 0 | 0 | 0 |
| 51 | 100040 | 100020 | 100021 | 100031 | 100041 | 100041 |
| 52 | 0 | 0 | 0 | 100021 | 100021 | 100011 |
| 53 | 0 | 0 | 0 | 100021 | 100021 | 100011 |
| 54 | 0 | 0 | 0 | 0 | 0 | 1000 |
| 55 | 100040 | 100020 | 100021 | 100031 | 100041 | 100041 |

Figure 10.13

Note: This figure shows part of a page of the requirement arrays as they are printed out by the Main Program. In array PR elements greater than 5000 denote assignments.

REQUIREMENT ARRAYS

| IRN | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| ERN | 10 | 20 | 30 | 40 | 45 | 50 | 60 | 70 |
| M | 1 | 1 | 1 | 2 | 1 | 2 | 1 | 1 |
| N | 3 | 3 | 3 | 0 | 0 | 0 | 1 | 1 |
| TIE | 1 | 2 | 3 | 5 | 4 | 6 | 8 | 7 |
| SET | 1 | 2 | 3 | 31 | 31 | 6 | 32 | 32 |
| EPN | PR | | | | | | | |
| 11 | -1 | -1 | -1 | 0 | 0 | 22 | -1 | 23 |
| 12 | -1 | -1 | -1 | 16 | 22 | 22 | -1 | 23 |
| 13 | -1 | 26 | -1 | 16 | 22 | 22 | -1 | -1 |
| 14 | 27 | -1 | -1 | 16 | 22 | 22 | -1 | -1 |
| 15 | 21 | 22 | 21 | 16 | 5022 | 22 | 21 | 22 |
| 21 | 30 | 34 | 5034 | 16 | 29 | 29 | 29 | 29 |
| 22 | 30 | 5035 | 34 | 16 | 29 | 29 | 29 | 29 |
| 23 | 5031 | 31 | 31 | 16 | 29 | 29 | 29 | 29 |
| 24 | 29 | 29 | 29 | 0 | 29 | 5029 | 29 | 29 |
| 25 | 29 | 29 | 29 | 0 | 0 | 5029 | 29 | 29 |
| 31 | 2 | 3 | 2 | 5003 | 3 | 3 | 2 | 3 |
| 32 | 3 | 3 | 3 | 5003 | 3 | 3 | 3 | 3 |
| 33 | 5006 | 6 | 4 | 3 | 3 | 3 | 4 | 6 |
| 34 | 0 | 11 | 5014 | 3 | 3 | 10 | 10 | 10 |
| 35 | 5 | 5012 | 12 | 3 | 3 | 10 | 10 | 10 |
| 41 | 18 | -1 | -1 | 16 | 16 | 17 | 21 | 21 |
| 42 | -1 | -1 | -1 | 16 | 16 | 17 | 21 | 17 |
| 43 | -1 | -1 | -1 | 16 | 16 | 17 | 21 | 17 |
| 44 | 15 | 16 | 15 | 5016 | 16 | 16 | 15 | 16 |
| 45 | 0 | 16 | 16 | 5016 | 16 | 16 | 16 | 16 |
| 51 | -1 | -1 | -1 | 16 | 35 | 29 | -1 | -1 |
| 52 | 28 | -1 | 38 | 16 | 28 | 29 | -1 | -1 |
| 53 | 28 | -1 | 38 | 16 | 28 | 29 | -1 | -1 |
| 54 | 36 | 36 | 36 | 16 | 5036 | 29 | 36 | 36 |
| 55 | -1 | -1 | -1 | 16 | 36 | 29 | -1 | -1 |
| DAY | DMARK | | | | | | | |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 5 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

# CHAPTER ELEVEN

## EVOLUTION AND TESTING OF THE MAIN PROGRAM

### 11.0 Introduction

The system which has been described in Chapters Two to Ten is really the current state of a system which has evolved over a period of time and which is still being developed. In this chapter we seek to explain some of the reasons for the course which this evolution has taken.

It must be said that neither the data preparation phase nor the Main Program has been put through the type of extensive field trial which would be essential to ascertain its value in any quantitative terms. The testing has been modest when compared for example with that carried out on the Ontario School Scheduling Program by J. Kates and Associates [10] or on the Newcastle-upon-Tyne University and Birmingham University Systems by the Local Government Operational Research Unit [12,13]. The Author still has hopes that the system will reach a state where such a full trial will be justified.

### 11.1 Parallel testing and development

During the main part of this research two versions of the Main Program have been in use at any given time. We will call these the 'A program' and the 'B program'. The A program at each stage is, in theory at least, a fully de-bugged program and is run mainly with data from real schools. The B program is a modification of the A program which has not been fully de-bugged and which is run mainly with small test case data sets based on imaginary schools. At appropriate stages, when the B program has performed satisfactorily on the test cases, the A program is scrapped, the B program takes its place and further development on a new B program is started. Invariably after such a change further bugs are detected in the new A program but the addition of larger imaginary problems to the data set for the B program has decreased this occurrence on the last two or three cycles of the process.

The changes made in the B program are of three main types:-

i) simple de-bugging;

ii) replacement of sections to perform a previously established process in a more efficient way;

and iii) addition of new features to try out new ideas.

The need for changes of type ii) has increased towards the end of the project due to the introduction of the more complicated rules like Rules B8 and B9. Even with more efficient methods for some of the basic operations, implementation of these rules has decreased the average rate of assignments on problems of real school size from around 430 per minute, with the first relatively successful version of the program, to about 200 per minute with the latest version. This decrease has been at least partly compensated for by a decrease in the number of retreats and, in the case of problems which have been solved, to a decrease in the number of runs needed to find a solution.

The changes of type iii) have in the main been based on analysis of the results from the A program. Most runs on this program are terminated during a long sequence of retreat operations and investigation of the circumstances which have caused these often leads to the discovery of a situation which "the computer should have been able to see". When a particular set of circumstances has arisen a sufficient number of times an attempt is made to alter the program so that in future it will be able to recognise those circumstances and hence avoid the difficulties which might arise. One of the most persistent problems has been the over allocation of free periods and the methods tested to counteract this tendency are discussed in 11.3.

Apart from some changes in storage methods the program described in Chapter 10 is the current A program.

## 11.2 Roots in the literature

### 11.2.1 The concept of units

When this project was first started what little literature there was on the subject [1, 3, 4 and 5] regarded a school time-table as a three-dimensional structure involving classes, teachers and periods or a four-dimensional structure if rooms were taken

into account. The first versions of the Author's program dealt only with classes, teachers and periods and were based on a similar model.

It was only when those sections of the program which dealt with classes became more and more like the sections which dealt with teachers that it was realised that as far as the time-table is concerned a class and a teacher are really the same sort of object. The only thing that distinguished them was that classes were originally assumed to have no free periods. Once this assumption was removed the need to distinguish between them vanished and the step which allowed rooms and other equipment to be viewed in the same light was a simple one. As we discuss in 11.3.5 a need to distinguish between classes, teachers and rooms (by including the type code on the Item List (2.1.6)) was later re-introduced as an aid to the problem of dealing with control of free period allocation but to a large extent the current model and program treat classes, teachers, rooms and equipment as equals.

11.2.2 Requirements and tied requirements

A second concept which was based on the existing literature was to regard the set of lessons during which a particular set of units meet as being associated in what we now call 'a single requirement' even when the lessons were of different lengths. A suitable format for such a requirement is:-

$NAME_r$, $TOTPER_r$, $DOUBLE_r$, $TRIPLE_r$, ($ITEM_{ri}$, i = 1,$n_r$);

where $NAME_r$ is the requirement name,

$TOTPER_r$ is the total number of periods involved,

$DOUBLE_r$ is the number of periods to be used in double period lessons;

$TRIPLE_r$ is the number of periods to be used in triple period lessons;

$ITEM_{ri}$ are the names of the $n_r$ units involved.

For want of a better name, we will call these compact requirements.

Example

Requirements 40 and 45 in the example problem namely,

40, 45, 2, 2, 10, 20, 50, 70, 100, 100;

45, 40, 1, 2, 10, 20, 50, 70, 100, 100;

can be expressed in a compact requirement as

4045, 6, 4, 0, 10, 20, 50, 70, 100, 100;

This approach was found to have a major disadvantage in that it makes it very difficult to record the current possible positions for lessons in a given compact requirement in such a way that lessons of different lengths can be considered independently. For example, the method of recording must be capable of showing cases where a single period lesson from a requirement might be allowed to occur at one of three consecutive periods while a double period lesson or a triple period lesson from the same requirement must not occupy any or all of these periods. While such a system can be devised it is not at all convenient when updating the arrays after an assignment or when detecting tight, forcing and other situations.

As the rules which the program had to apply became more complex this disadvantage became more and more apparent and the idea of expressing compact requirements as a set of requirements each of which involves only one lesson length was conceived. This step was necessarily accompanied by the concept of tied requirements in order to maintain methods of applying the distribution constraints. The previous assumption had been that some attempt should be made to assign the lessons in each compact requirement with a 'good' distribution but that no account needed to be taken of the relative distributions of lessons in different requirements.

In addition to relieving the difficulties in recording the current state of the time-table the revised system was found to make the model much more flexible in dealing with distribution problems. The following example illustrates this point.

Example

A class (CLASS1) has three Chemistry lessons per week, a single period in an ordinary classroom (ORDRM) and two double periods in a Chemistry laboratory (CHEMLB). All lessons are taught by Mr. Beatty.

Using compact requirements this may be expressed as:-

CHEMCL1A, 1, 0, 0, CLASS1, BEATTY, ORDRM;

CHEMCL1B, 4, 2, 0, CLASS1, BEATTY, CHEMLB;

Using the model defined in Chapter Two we have:-

CHEMCL1A, CHEMCL1B, 1, 1, CLASS1, BEATTY, ORDRM;
CHEMCL1B, CHEMCL1A, 2, 2, CLASS1, BEATTY, CHEMLB;

Since distribution constraints were applied only within a compact requirement the first method of describing these lessons does not say that the single period lesson should not occur on the same day as a double period lesson but the second method does.

A difficulty which had to be taken into account with the tied requirement system was that when a compact requirement was split into several requirements the units involved appeared to gain some potential activities at each period.

Example

Expressing as compact requirements all the requirements involving item 100 in the example problem we get:-

4045, 6, 4, 0, 10, 20, 50, 70, 100, 100;
160165, 5, 4, 0, 30, 70, 100;
170175, 5, 4, 0, 30, 50, 100;

Initially item 100 can be used twice in requirement 4045 and once in each of the other two making a total of four potential activities at each period plus the possibility of one or both units being free.

In expanded form these are:-

40, 45, 2, 2, 10, 20, 50, 70, 100, 100;
45, 40, 1, 2, 10, 20, 50, 70, 100, 100;
160, 165, 2, 2, 30, 70, 100;
165, 160, 1, 1, 30, 70, 100;
170, 175, 2, 2, 30, 50, 100;
175, 170, 1, 1, 30, 50, 100;

Counting in the same way this appears to give item 100 a total of eight potential activities at each period, again plus the possibility of one or both units being free.

This apparent increase in the total number of potential activities is significant when we come to detect when a tight or forcing situation has arisen since a true potential of one activity which should be detected may appear as a potential of two or more activities and hence not be detected. This problem is avoided by counting the potential activities as described

in 6.7. The number recorded in array TPA is then exactly as it would have been had compact requirements been used.

11.2.3 Assignment strategy versus opportunity list maintenance

A third basic idea which was originally unconciously adopted, presumably as a result of reading the existing literature, was that the operation of selecting the next assignment is of overriding importance. The methods of Appleby, Blake and Newman [1] for example, consisted of little more than a complex method for calculating the best assignment to make next. The operation of deciding which assignments not to select next was not given so much emphasis. Lions [11] calls these two processes 'assignment strategy' and 'maintenance of the opportunity list'.

Up to mid 1968 the sections of the program which were best developed were those for detecting tight, forcing and impossible situations in an effort to make sure that the next assignment would always be chosen from a tight part of the problem. Maintenance of the opportunity list i.e. cancellations in arrays PR and IFM, was confined to application of the simpler of the rules described in 7.3 and 8.2. Since then the developments have been aimed at making the cancellation rules for arrays PR and IFM as perfect as possible without making them so complex that they cannot be applied efficiently. We have already commented (7.6) that if the rules for cancellation in array PR could be made perfect then any available element could be chosen as the next assignment i.e. virtually no assignment strategy would be needed. This stage has certainly not been reached and all the developed techniques associated with the assignment strategy still form part of the current program.

11.3 The most persistent problem

All versions of the Main Program have made sure that a particular unit cannot be given more than the correct number of free periods but the earliest versions made no attempt to control the number of units allowed to be free at a particular period. When tested on real schools it was soon shown that some method of control on this factor was required, for repeatedly it was found that long series of advance-retreat operations had been caused by the allocation of too many free periods at a particular period. This problem still persists despite considerable

Figure 11.1

| | 4 | 7 | 10 | 13 | 22 | 30 | 34 | 55 | 56 | 59 | 60 | 61 | TPA | IFM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 102 | | | | | | X | X | X | | | | | 3 | 0 |
| 103 | | | | | | | X | | | X | | | 2 | 0 |
| 104 | | | | | | X | X | X | | | | | 3 | 0 |
| 105 | | | | X | | | X | | | | | | 2 | 0 |
| 106 | | | | | | X | X | | | X | | | 3 | 0 |
| 107 | | | X | | | | | | | X | | | 2 | 0 |
| 108 | X | | | | X | | | | | | X | | 3 | 0 |
| 109 | | | | | | X | X | | | | | | 2 | 0 |
| 110 | | X | | | X | | | | | | | | 2 | 0 |
| 111 | | | | | | X | X | | | | | X | 3 | 0 |
| 112 | | | | | | | X | | X | | | | 2 | 0 |
| TPA | 1 | 1 | 1 | 1 | 2 | 5 | 8 | 2 | 1 | 3 | 1 | 1 | | |
| IFM | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | |

Note: i) the numbers along the top represent teachers;

ii) the numbers down the side represent classes;

iii) an X indicates that a class and a teacher may meet at the period concerned;

iv) the values of TPA and IFM in the last two columns are for the classes while those in the last two rows are for teachers.

efforts to solve it.

When the problem was first considered the difficulties in calculating exactly the total number of units which can be free at a given period were quickly recognised. It was also seen that even when such a total is available most complicated impossible situations can arise. One of these is discussed in the following section.

11.3.1 A typical impossible situation involving free periods

The following situation arose in an attempted solution for a real school:-

At period 8 on Monday 11 classes named 102, 103, 104, 105, 106, 107, 108, 109, 110, 111 and 112 and 12 teachers named 4, 7, 10, 13, 22, 30, 34, 55, 56, 59, 60 and 61 remain to be employed and all possible lessons involve exactly one class and one teacher. The possible teachers for each class are shown in

figure 11.1.

Other relevant facts are:-

i) each class must be employed on a lesson i.e. none may have a free period, as shown by the values of IFM;

ii) each teacher may have a free period if necessary as shown by the values of IFM;

and iii) none of the lessons must take place on the day concerned and there are no other circumstances which cause any assignment to be forced.

Now since there are 11 classes and 12 teachers only one teacher may in fact be free. By looking at the possible lessons for class 108 we see that either 4 or 60 must have this free period and class 108 must be taught by the other. This means that 110 must be taught by 7 or 22 and the other must have a free period which is impossible. Since, however, there are many tight situations in this set up which do not involve items 108, 4 or 60 and numerous other tight situations at other periods in the time-table the chances of selecting to assign class 108 with either 4 or 60 are slim. The program is much more likely to make a series of chains of assignments both at period 8 and at other periods which do not reveal the existing impossibility. By the time the impossible situation is detected the stage at which it originally arose is so far back that it would take hundreds of retreats to find it. Such a long sequence always leads to the termination of the run.

In this particular case the problem was detected after assignment number 288 and investigation showed that it had been present from assignment 256 onwards. The situation which existed after assignment 255 is shown in figure 11.2. At this stage there are several possible solutions one of which is indicated by the underlined elements.

11.3.2 Attempts to avoid the problem

Investigation of a large number of situations like that discussed in 11.3.1 showed that frequently the set of units which were available just before the impossible situation was introduced included a 'busy' unit i.e. a teacher or a room involved in a large number of different requirements e.g. a teacher of Art or a gymnasium. Further, it was observed that in a significant

Figure 11.2

| | 4 | 5 | 6 | 7 | 10 | 13 | 16 | 22 | 30 | 34 | 55 | 56 | 59 | 60 | 61 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 102 | | | | | | | | X | X | X | X | | | | |
| 103 | | | | | | | | | | X | | | X | | |
| 104 | | | | | X | | X | | X | X | X | | | | |
| 105 | | | | | | X | | | | X | | | | | |
| 106 | | | | | | | | | X | X | | | X | | |
| 107 | | | X | | X | | | | | | | | X | | |
| 108 | X | | | | | | | X | | | | | | X | |
| 109 | | | | | | | | | X | X | | | | | |
| 110 | | | | X | | | | X | | | | | | | |
| 111 | | | | | | | | | X | X | | | | | X |
| 112 | | X | | | | | | | | X | | X | | | |

number of cases the impossible situation was only detected when an assignment was made involving this busy unit. Thus, accepting that impossible situations were bound to arise, the policy adopted was to try to make sure that, having arisen, they would be detected as quickly as possible. A priority was therefore given to all requirements involving one or more busy units so that the lessons in them would be assigned fairly early in the construction process. This principle is incorporated in the priority parameter (10.3).

Although this step was effective to some extent many long chains of retreats continued to occur and most of these were still caused by the same problem but no longer involved 'busy' units so frequently.

11.3.3 A first attempt to solve the problem

In discussing the example in 11.3.1 we knew that exactly one teacher could be free but in fact this information was not known by the computer. It was clearly desirable that in such situations the computer should have at least some idea of the number of units which could be given free periods.

The first method used to attempt to do this was to include in the data an upper bound for the number of units which could be free at each period of the week. This method was still in

use when Johnston and Wolfenden [6] was written and these bounds appeared on the data document which was called 'The Header Sheet'.

For some schools this upper bound could be calculated exactly at most periods of the week and the method was successful but, for schools with a large amount of setting where the number of teachers and rooms involved is not equal to the number of classes, the upper bound could not be obtained accurately at any period of the week. If the bound at a given period was just one too high the program could allocate one more free period than it should have done and would then find at some later stage that a class was left without a teacher at that period hence causing a retreat. Similarly if the bound was one too low the program would sometimes refuse to allow the allocation of another free period at times when such an allocation would have been perfectly in order. It then proceeded to remove assignments thinking that an impossible situation existed when in fact it did not.

On the whole this method caused more trouble than it prevented and so was rejected.

11.3.4 A second attempt to solve the problem

The next method used to attempt to provide the computer with a knowledge of the number of units which could be allocated free periods at a given period involved the following:-

i) Extra requirements were added to each set of data requesting the free periods for all classes to be treated as lessons i.e. no class could have a normal free period.

ii) A constant, BAL[r], was included in each requirement which was calculated as follows:-

BAL[r] = 2.(number of classes involved)
- (number of teachers involved)
- (number of rooms involved).

iii) A parameter called MEANFREE was also included in the data. This was calculated from:-

MEANFREE = total number of teachers
+ total number of rooms
- 2.(total number of classes)

Now if a requirement involved the same number of classes, teachers and rooms the value of BAL[r] was zero showing that the assignment of such a lesson would not alter the total number of

units which could be free. If, however, n extra teachers and m extra rooms were involved then BAL[r] was -(n + m) showing that the assignment of such a lesson at a given period would decrease by n + m the total number of units which could be free. Similarly on requirements with fewer teachers and/or rooms than classes BAL[r] was > 0 showing that extra free periods could be allocated to any period involving a lesson from that requirement.

The operation of the system involved setting a 'free period count' FPC[p] on period p, initially equal to MEANFREE, and then adding BAL[r] to FPC[p] if a lesson from requirement r was assigned to period p. FPC[p] therefore recorded a current estimate of the total number of units which could still be allocated free periods at period p. The priority parameter contained terms, as it still does, to ensure that requirements with negative balance counts were dealt with early on in the construction process.

Again this system was quite successful with some sets of data but not with others. It was most successful on data from cases where rooms are not taken into account at all but with the other sets of data chains of retreats continued to occur which on investigation were found to have been caused by faulty allocation of free periods. The fault was found to be, not that too many or too few free periods were being allocated overall, but that when teachers and rooms were considered separately either one or the other was getting more than its fair share. Usually a teacher had been allocated a free period which should have been reserved for a room.

The above method could have been adjusted to treat teacher free periods and room free periods separately by simply having two balance counts in each requirement, one for the balance between classes and teachers and the other for the balance between classes and rooms, and including two MEANFREE type parameters in the data. However, due to the effort of calculating all this additional data and the accompanying risk of errors it was decided to let the computer do these calculations.

### 11.3.5 The current attempt to solve the problem

So that the computer could calculate the balance counts for each requirement it became necessary to include in the data

which type of object each item name represents. Type codes were therefore introduced on the Item List. This step was taken reluctantly as the Author feels that this information should not be necessary.

The details of the current system have already been described in 10.7.3. One of the advantages of this system is that it is no longer necessary to have the restriction that all class free periods must be requested in requirements.

Since this system has only been incorporated in the A program for a few months it has not yet been as fully proved as the earlier systems were. Tentative conclusions from investigation of the results so far obtained are:-

i) Impossible situations associated with allocation of free periods, but not caused by over allocation of free periods, do still occur. These are basically the same as that shown in figure 11.1 where, although excessive allocation of free periods has not taken place, the circumstances at a given period, or more often at a set of periods, are such that such excess allocations are bound to take place sooner or later.

ii) The system depends for its success on the early assignment of all unbalanced requirements, thus it does not work so well for data sets which include a large number of such requirements.

11.4 Other main problem areas

We have already discussed the efforts made to combat the tendency of the program to cause impossible situations by excess allocation of free periods. There are two other main patterns of behaviour which cause trouble and for which there seem to be no simple remedies.

The first of these is the building up of complex situations involving three or more requirements in which there are tight, forcing or even impossible situations which depend on relationships between different days of the week. We can illustrate one such situation by the following example which is based on a situation from a real school time-table.

Example

Figure 11.3 shows a set of 5 requirements which form part

Figure 11.3

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | IRN | 15 | 16 | 17 | 18 | 19 | |
| | | ERN | 15 | 16 | 17 | 18 | 19 | |
| | | M | 2 | 1 | 1 | 1 | 1 | |
| | | N | 1 | 2 | 3 | 1 | 3 | |
| | | TIE | 15 | 16 | 17 | 18 | 19 | |
| IPN | 1 | PR | -1 | 2 | 2 | 2 | 2 | |
| | 2 | | -3 | 2 | 2 | 2 | 2 | |
| | 3 | | -2 | -1 | -1 | -1 | 4 | ← |
| | 4 | | 19 | 19 | 19 | 21 | -1 | |
| | 5 | | 19 | 19 | 19 | -1 | -1 | ← |
| | 6 | | 0 | 0 | 0 | 0 | 0 | |
| | 7 | | -1 | -1 | -1 | 6 | -1 | ← |
| | 8 | | -2 | 20 | 20 | 20 | -1 | |
| | 9 | | 10 | -1 | -1 | -1 | -1 | ← |
| | 10 | | 10 | 10 | 10 | 10 | 10 | |
| | 11 | | -1 | 8 | 8 | 9 | 7 | |
| | 12 | | -2 | 22 | 11 | 11 | 7 | |
| | 13 | | 7 | 7 | 7 | 7 | 7 | |
| | 14 | | -1 | -1 | 11 | 11 | 7 | ← |
| | 15 | | -2 | 23 | 11 | 11 | 7 | |
| | 16 | | 5 | 15 | 15 | 15 | 14 | |
| | 17 | | 5 | 14 | 14 | 14 | 14 | |
| | 18 | | 5 | 16 | -1 | -1 | 14 | ← |
| | 19 | | 5 | 5 | 5 | 5 | 5 | |
| | 20 | | 5 | 5 | 5 | 5 | 5 | |
| | 21 | | -1 | 13 | 17 | 13 | -1 | |
| | 22 | | -3 | 18 | 17 | 17 | -1 | |
| | 23 | | -2 | 18 | 17 | 17 | -1 | |
| | 24 | | 24 | -1 | -1 | -1 | -1 | ← |
| | 25 | | 12 | 12 | 12 | 12 | 12 | |

Figure 11.4

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | IRN | 15 | 16 | 17 | 18 | 19 |
| | ERN | 15 | 16 | 17 | 18 | 19 |
| | M | 2 | 1 | 1 | 1 | 1 |
| | N | 0 | 1 | 1 | 0 | 1 |
| | TIE | 15 | 16 | 17 | 18 | 19 |
| | | 25 | 2 | 2 | 2 | 2 |
| | | 25 | 2 | 2 | 2 | 2 |
| | | 25 | 25 | 25 | 25 | 4 |
| | | 19 | 19 | 19 | 21 | 30 |
| | | 19 | 19 | 19 | 29 | 29 |
| | | 0 | 0 | 0 | 0 | 0 |
| | | 25 | -1 | -1 | 6 | 25 |
| | | 25 | 20 | 20 | 20 | 31 |
| | | 10 | -1 | -1 | 25 | 25 |
| | | 10 | 10 | 10 | 10 | 10 |
| | | 25 | 8 | 8 | 9 | 7 |
| | | 25 | 22 | 11 | 11 | 7 |
| | | 7 | 7 | 7 | 7 | 7 |
| | | 25 | 28 | 11 | 11 | 7 |
| | | 25 | 23 | 11 | 11 | 7 |
| | | 5 | 15 | 15 | 15 | 14 |
| | | 5 | 14 | 14 | 14 | 14 |
| | | 5 | 16 | 26 | 26 | 14 |
| | | 5 | 5 | 5 | 5 | 5 |
| | | 5 | 5 | 5 | 5 | 5 |
| | | 25 | 13 | 17 | 13 | -1 |
| | | 25 | 18 | 17 | 17 | -1 |
| | | 25 | 18 | 17 | 17 | -1 |
| | | 24 | 27 | 27 | 27 | 27 |
| | | 12 | 12 | 12 | 12 | 12 |

Note: underlined elements denote assignments.

of the item band for an item with one life. In this case there is a tight situation involving elements PR[14,16] and PR[5,18] for, considering requirements 16, 17 and 18 together, we see that 6 single period lessons have to be assigned but there are only 7 possible periods (indicated by arrows). Since at period 14 only requirement 16 has an available element and at period 5 only requirement 18 has an available element an assignment must be made in at least one of these elements. None of the existing rules detects this situation.

If any assignment is made at period 5 or period 14 in the same item band but not in requirement 16, 17 or 18 then the tight situation becomes a forcing situation but it is still not detected by any of the existing rules.

If, however, any assignment is made at one of the other periods in the same item band, but not in requirement 16, 17 or 18, forcing and tight situations are located by existing rules. In particular, if an assignment were made in PR[2,15] causing cancel PR(3,16), cancel PR(3,17) and cancel PR(3,18) to be performed then applying the existing rules causes a chain of assignments in PR[18,17], PR[24,17], PR[14,16], PR[5,18], PR[4,19] and PR[8,19]. The resulting arrays are shown in figure 11.4.

From the above example it is clear that the existence of tightly knit relationships between several requirements involving more than one day of the week are a potential danger in a partial solution. At best such a situation may lead to a long chain of forced assignments and long chains themselves are more likely to lead to impossible situations than short ones. At worst the situation may involve an undetected impossible situation and be left untouched while large areas of the time-table elsewhere are inserted. In such a case the impossible situation must be revealed eventually but it may only be at a stage when to correct it, by the advance-retreat technique, would involve a large number of retreats.

Rules B9 and D6 were designed to combat this sort of occurrence but they only consider pairs of requirements. To extend these rules to cover larger sets of requirements is straight forward but the implementation of them becomes more difficult and execution time taken to apply the necessary tests is unacceptably

long.

The other troublesome pattern of behaviour is that the program sometimes detects an impossible situation and performs a retreat which removes the means of detecting the impossible situation but does not remove the impossible situation itself. The subsequent advance then fails to detect the impossible situation again until several chains of assignments have been made. When it is again detected the advance-retreat technique has little chance of ever working back to the source of the trouble unless a very large number of retreats is allowed.

Due to these two difficulties and the continuing recurrence of problems with the allocation of free periods it would appear that sole dependence on the advance-retreat technique to produce a solution is unwise. When we admit that for most sets of data from real schools there probably is no solution of the type required then it is clear that success will only be achieved if the program is so powerful that cases with no solution are diagnosed before completion of the initial chain. The fact that our rules for cancellation in array PR and IFM and for detecting forcing and impossible situations are known to be inadequate makes such a diagnosis in every case unachievable. It also means that even in a case where a solution does exist the program may introduce an impossible situation and fail to detect it sufficiently quickly to be able to correct it. Thereafter the problem is equivalent to one with no solution. Possible changes in policy which would make the program more effective in both these sets of circumstances are discussed in Chapter 14.

# CHAPTER TWELVE

## THE TEST DATA USED

### 12.0 Introduction

Throughout the development of the system data from a variety of sources has been used. Much of the basic program testing for de-bugging purposes has been done using small imaginary problems including the one which is being used as the example problem in this thesis. For evaluation purposes, however, it was essential to test the various versions of the program on realistic data from actual schools. Several such sets of data have been obtained during the project and we will now discuss these. Some relevant statistics are shown in figure 12.1.

Figure 12.1

| | School Code Number (see below) | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Number of days | 5 | 5 | 5 | 5 | 5 | 6 | 6 |
| Number of periods | 40 | 40 | 40 | 40 | 35 | 36 | 36 |
| Number of items | 93 | 107 | 17 | 34 | 205 | 56 | 86* |
| Number of units | 93 | 107 | 17 | 34 | 230 | 56 | 86 |
| Number of requirements | 296 | 410 | 74 | 131 | 537 | 216 | 216 |
| Number of assignments | 856 | 975 | 264 | 315 | 1025 | 479 | 479 |
| Number of single period lessons | 813 | 816 | 224 | 245 | 826 | 401 | 401 |
| Number of double period lessons | 43 | 132 | 40 | 65 | 143 | 78 | 78 |
| Number of triple period lessons | 0 | 25 | 0 | 5 | 54 | 0 | 0 |
| Number of other lessons | 0 | 2 | 0 | 0 | 2 | 0 | 0 |

School Code Numbers

1 = R.B.A.I.
2 = G.U.S.
3 = H.C.T. 1965-66
4 = H.C.T. 1966-67
5 = H.P.S.
6 = G.P.S.
7 = G.P.S. (revised data)

* In this case the 30 extra units were dummy classes used in the description of the Sixth Form (12.5).

## 12.1 R.B.A.I. - a Grammar School in Belfast

After only a few months work it was considered essential to study a few actual school time-tables thus, early in 1964, the time-table then in use at R.B.A.I., a grammar school of about 1100 pupils in Belfast, was obtained. This provided useful knowledge of the sort of complexity which can be expected in time-tables in medium and large schools and, together with the time-table for the school discussed in 12.2, strongly influenced the features which are built into the model. From the existing time-table a set of data was also prepared so that an attempt could be made to reconstruct it. The data structure at that stage was very different from the current structure but the data has been re-expressed several times to keep it up to date with developments.

As the structure of the Sixth Form curriculum for this school is very complex it was decided to pre-assign all the Sixth Form classes as they appear in the existing time-table. Although such a practice would not be acceptable in general, it has provided a most useful test bed for the initialisation phase of the Main Program for the inclusion of so many pre-assignments causes many elements in arrays PR and IFM to be cancelled and hence the initial chain is long. As the various sets of rules have been improved the length of this initial chain has steadily increased.

Even the early versions of the program showed that there is no solution of the type required but a series of alterations to the data has enabled a solution to be obtained. The types of alteration made are illustrated by the following examples.

Example

The requirement:-

4130, 0, 1, 5, 502, 503, 504, 40, 47, 45;

describes setted lessons for three classes from the Fifth Form.

In order to obtain a solution this had to be replaced by:-

4130, 0, 1, 4, 502, 503, 504, 40, 47, 45;

4135, 0, 1, 1, 502, 503, 504, 40, 47, 45;

This change allows two of the lessons to occur on one day leaving another day without a lesson. Such non-normal distributions were already present in the existing time-table and a few such distributions would be acceptable to most schools.

Example

The requirements:-

240, 0, 1, 4, 106, 47;

2230, 0, 1, 4, 305, 44;

describe a set of French lessons for a section of the First Form and a similar set of lessons for a section of the Third Form.

In order to obtain a solution these had to be replaced by

240, 0, 1, 4, 106, 44;

2230, 0, 1, 4, 305, 47;

i.e. the teachers had to be interchanged.

In practice such an interchange could rarely be carried out without first consulting the school concerned. It would be difficult for example if the numbers of lessons involved were not the same as the interchange would then decrease the teaching load of one teacher at the expense of the other.

Although a solution has been obtained on two occasions, in each case starting with the same original data, the number of runs taken (approximately 12 for the first solution and 9 for the second) is not acceptable for two main reasons. First of all the total computer time taken (approximately $2\frac{1}{2}$ minutes per run), about half of which is used in the initialisation phase and the output of results, is excessive. Secondly, the need to consult the school before changes to the data could be made would increase the time between runs and hence make the total time for the operation much too long.

12.2 G.U.S. - an Upper School in Leicestershire

Soon after the data from R.B.A.I. had been obtained an existing time-table was also obtained from G.U.S., an Upper School in Leicestershire. This school at that time had a Fourth Form and a Fifth Form each with 14 well defined streams, a Lower Sixth Form and an Upper Sixth Form each with approximately 9 less well defined streams and a Third Sixth Form with little or no definable structure.

A data set was produced in the same way as with R.B.A.I. except that the Sixth Form structure is described almost completely. Only a few difficult lesson patterns are dealt with using pre-assignments.

Attempts to obtain a solution to this problem have been

singularly unsuccessful. The type of behaviour encountered is very similar to that for school H.P.S. which is discussed in 12.4. Although numerous adjustments have been made to the data to remove proven impossibilities it has been concluded that there remain inherent in the problem features which make it impossible to produce a time-table with qualities approaching those implied in the model.

12.3 H.C.T. 1965-66 and 1966-67 - a small Canadian School

Following publication of Lions [10], which contains a report on the testing of the Ontario School Scheduling Program, the Author approached Dr. Lions with a view to obtaining some of the data used on these tests. As a result of this approach Dr. Lions kindly forwarded the data for 1965-66 for the school denoted by H.C.T. in his paper.

Tests on this set of data showed that there is definitely no solution of the type defined inherently in our model but no attempt to change the data in any way was made. Correspondence with Dr. Lions revealed that they too had had extreme difficulty in obtaining a solution and that the one which they finally accepted was not what would normally be described as a good time-table.

As a result of the correspondence on the 1965-66 data Dr. Lions forwarded data from the same school for the academic year 1966-67 when the school had become slightly larger and changes in staffing and facilities had removed some of the factors which had made the 1965-66 time-table so difficult.

Experiments with this set of data produced a solution initially in 5 runs and, with a later version of the program, in 3 runs. The data and the latter solution for this problem are given in the Appendix.

12.4 H.P.S. - a Comprehensive School in London

Following an enquiry about possible aid in time-table construction from a member of the staff at H.P.S., a large comprehensive school in West London, the Author agreed to attempt to make their school time-table but pointed out that the chances of success were slim. This led to a most useful exercise in data collection and checking but not unfortunately to any complete time-tables.

At that stage in the development of the system all the data used had been obtained from the sources discussed in 12.1, 2 and 3 and the errors contained were relatively minor thus, although a data checking program was used, it was not highly developed and the data preparation system had not been designed for use by school teachers nor with a view to avoiding errors.

The first method used to obtain data from H.P.S. was simply to ask each Head of Department to supply the data relevant to his Department in the form in which he was accustomed to giving it to the manual time-tabler. Excellent co-operation was obtained from all the staff concerned but the mass of information which was produced took quite a bit of sorting out and when it had been sorted out it was found to contain a very large number of serious inaccuracies and inconsistencies.

This experience showed very clearly that some firmer guidance was necessary to enable the school staff to present their data in a form which could easily be prepared for acceptance by the computer and to help them to avoid making so many mistakes. The data preparation system which was described in Johnston and Wolfenden [6], and which differs only in minor respects from the current system, was therefore developed.

A 22 page booklet describing this system was prepared and copies sent to those members of staff at H.P.S. who were in any way involved in the time-tabling operation. When they had had time to read it a meeting was held at the school at which the Author discussed the system and the general problems involved in getting a computer to make a time-table. Also, the staff had an opportunity to ask questions and to make suggestions. In time for the same meeting the Item List, the Curriculum, an index card for each requirement and forms, similar to but less elaborate than those shown in Chapter Four, had been prepared by a senior member of the school staff and the Author. These were distributed to the members of staff concerned and they were given a few weeks to prepare their data. Once again the staff co-operated fully in the exercise and many were extremely interested in the whole project. The Author visited the school on several occasions to further explain the system to individual members of staff and to deal with problems which had arisen.

The end product of this exercise was a bundle of over 500 index cards containing the requirements and a sheaf of forms containing the Teacher Duty Chart and the other documents. Little or no sorting out was required and the conversion of the data to punched cards was a straightforward task.

Processing this data with the Data Checking Routine revealed only 21 mistakes as follows:-

3 punching errors;

8 errors of omission because requirement cards for the Sixth Form options had not been circulated to the Mathematics Department;

3 errors involving over-loading of teachers or rooms due to misunderstanding of the system by one Head of Department;

7 errors on the pre-assignment list due to changes being made on the Curriculum without making the corresponding changes to the pre-assignments.

For a school involving a total of 205 items this was regarded as a satisfactory result.

When the detected errors had been removed a series of runs, 14 in all, was carried out using the Main Program.

The first few runs were terminated in the initial chain and each revealed one or more reasons why no solution existed. The following examples illustrate the type of difficulties which were detected at this stage.

Example

A Mathematics teacher, who was required to have 5 single period lessons with a Second Form Class, had a complete day of fixed free periods so that he could attend a course outside the school. This prevented the proper distribution of the 5 lessons but could only be detected after the fixed periods had been assigned.

Example

A section of the Fourth Form had Games pre-assigned on Tuesday afternoon and a double period of Engineering on Tuesday morning leaving only 2 periods unassigned on Tuesday. They also had 5 single period lessons per week of each of English and Mathematics, thus the two remaining periods on Tuesday had to be reserved for these subjects. However, they had 4 periods of French with a teacher whose other commitments meant that he

could not teach them on Friday thus a French lesson was also essential on Tuesday i.e. an impossible situation existed.

Once all difficulties detected in the initial chain had been removed the next few runs were terminated due to a long chain of retreats after only a small number of assignments (50 to 60). In these cases it was often possible to trace the cause of the trouble back to the data. Some of the difficulties of this type which arose are described in the following examples.

Example

A part-time teacher, who was only present for 10 periods per week, had been asked to take 2 3-period lessons with the Lower Sixth Form and 2 2-period lessons with the Upper Sixth Form. Her hours of duty were 6 consecutive periods on one day and 4 consecutive periods on another. At most 3 of her 4 lessons were therefore possible.

Example

An advanced Biology laboratory was required for 33 out of the possible 35 periods in the week thus it could only be free at 2 periods. It was used by Sixth Form classes for 28 periods all in double or triple period lessons, by a Fifth Form class for one single period lesson and by a Fourth Form class for 2 double period lessons taught by a part time teacher.

On Monday morning the whole of the Sixth Form was pre-assigned to have Games thus it could not use the laboratory at periods 1, 2, 3 or 4. Also the part time teacher was not available at all on a Monday. The allocation of the Fifth Form lesson to Monday period 1 was possible but no lesson could be assigned to periods 2, 3 and 4 thus the laboratory had to be free for at least 3 periods.

Example

All Physics and Chemistry laboratories had been given a 100% work load and although no specific failure could be blamed on this fact it was suspected that some of the many difficulties were caused indirectly by it.

Most of the remaining runs in the series were again terminated by long sequences of retreats but this occurred after so many assignments had been made that it was no longer possible to

trace the difficulties back to the data. In some cases the cause of the difficulty could be traced back to a point, X say, in the process where one of the situations discussed in 11.3 and 11.4 could be seen to have arisen but it could rarely be said with certainty whether this had been caused by the program, through its choice of assignments, or was inevitable due to inherent impossibilities in the data set. The policy with such situations was to add to the Pre-Assignment List all assignments up to the end of the chain before the chain containing the point X and then to make further pre-assignments which attempted to avoid the occurrence of the same problem again. If the impossible situation had not been removed by this process the next run usually failed with a different set of circumstances still involving the same area of the time-table. By this method evidence could be obtained on which requirements were likely to be involved if inherent impossible situations existed in the data. The evidence, however, was rarely conclusive.

Clearly, if the series of runs attempting to solve any problem gets to this stage it may take several runs to detect one impossible situation in the data and the total number of runs to obtain a solution, if a solution was ever obtained, could be very large and therefore unacceptable.

This type of behaviour occurred in attempting to solve H.P.S., G.U.S. and one of the larger imaginary problems used and in all three cases it was concluded that there is no solution of the type required. In the cases of H.P.S. and the imaginary problem, further short series of runs have been carried out using the latest version of the program starting with the data decks in use at the end of the previous series. The only significant difference between these tests and the earlier ones is that the long chains of retreats tend to occur earlier in the process i.e. the program appears to be able to make less of the time-table than the earlier version. This is interpreted as indicating that the more powerful processes applied by the newer program track down the difficulties more rapidly. The other possible interpretation, that the new program is just not as good as the old one, is ruled out by its better performance on problems including H.C.T. 1966-67 (12.3), which do have solutions.

The program, however, is still not powerful enough to produce solutions to the more difficult problems with a sufficiently small number of runs.

12.5 G.P.S. - a Public School in Scotland

Following publication of Johnston and Wolfenden [6], a teacher from G.P.S. offered to express the data for the school time-table in the required form. This offer was gratefully accepted and the data was, in due course, received.

When checked using the Data Checking Routine the only serious errors were in pre-assigning too many free periods for several of the items. A first run with the Main Program, however, revealed much more serious difficulties for, in an effort to describe the required Sixth Form structure within the model, almost 75% of the Sixth Form lessons had been pre-assigned. Some of these pre-assignments were inconsistent with one another or inconsistent with requests on the Fixed Free Period List but they also caused an unknown number of impossible situations in other parts of the time-table. Five further runs were carried out each of which revealed additional difficulties all of which had been caused by the pre-assignments. It was then concluded that the problem was not likely to yield a solution and tests with this data set were suspended.

As the failure with the data set discussed above was assumed to have been caused by the Sixth Form pre-assignments the Author redescribed the Sixth Form requirements so that the same structure was achieved but no pre-assignments were required. A further six runs with this modified data were carried out the best of which assigned just over 50% of the lessons including all but 3 of the 78 double period lessons required.

An error during this series of runs showed that a fairly small reduction in the number of lessons to be assigned could make the difference between success and failure for on one run a set of seven requirement cards involving a total of 19 lessons was accidently omitted from the data deck and a complete solution was obtained. Investigation failed to reveal why the inclusion of these requirements should make a solution impossible and the Author is unable to decide whether the failure with the full data set is due to an inherent impossibility in the data or to

the program making unwise decisions, although the latter seems the more likely reason. The possibility of allowing the program to omit troublesome lessons and hence perhaps produce an almost complete solution is discussed in 14.5.

## 12.6 Conclusions

From the complete set of tests done on the Main Program, not only on the real school data discussed in this chapter but also on the imaginary test problems we can draw the following conclusions:-

i) If a set of data has a complete solution then the program has a very good chance of finding it but, in the case of a large data set, may make unwise decisions which result in long series of retreats and have to be corrected by the analyst thus requiring too many runs to be done.

ii) If a set of data does not have a complete solution but one or more reasons for this are fairly obvious then the program will probably detect them while it is in the initial chain or they will cause a long sequence of retreats at an early stage in the construction process from which the analyst can detect the source of the trouble.

iii) If a set of data does not have a complete solution but the reasons are complex then the program is unlikely to provide a means of finding them since they will probably cause a long sequence of retreats after quite a few chains of assignments have been made and it will be difficult for the analyst to determine the source of the trouble.

The overall conclusion must therefore be that for real schools the Main Program is not a satisfactory means of producing a time-table sufficiently rapidly and at a low enough cost. A possible method for improving the situation is discussed in Chapter 14.

# CHAPTER THIRTEEN

## AN ALTERNATIVE APPROACH

### 13.0 Introduction

In a paper presented at the 1966 British Joint Computer Conference N. L. Lawrie [7], said while discussing the formation of a time-table:-

> "It may be useful at this point to describe this combinatorial problem less in terms of classes and teachers than in terms of 'lines', each of which describes a class-teacher combination which must occur a specified number of times in the completed time-table. We call two lines 'compatible' if they do not have a class or a teacher in common and describe as an 'arrangement' a set of lines from the table with the following two properties:-
>
> (1) the lines in the arrangement must be compatible in pairs
>
> (2) each class must occur in some line of the arrangement.
>
> The problem then is to choose as many arrangements as there are periods in the time-tabling week so that each line meets its correct number of times and so that constraints on good distribution and on the occurrence of certain lines at specified times are satisfied."

Lawrie proceeds to discuss how this approach to the problem can be expressed as a linear programming problem but does not indicate that, at that time, he had attempted to put his ideas into practice. He has since published a further paper [8] in which he describes the first phase of an algorithm to apply the process.

Before the publication of this second paper the present Author recalled the ideas expressed in the earlier paper and realised that such an approach would avoid the difficulty encountered by the Main Program in allocating free periods (11.3). Some work was therefore commenced to determine whether or not this approach can be made practical and although it has not been brought to a satisfactory conclusion it is felt worthwhile to discuss the progress made.

## 13.1 Definitions

Using our term 'requirement' in place of Lawrie's term 'line' and allowing for the fact that our items may have more than one life we may re-express and extend Lawrie's terminology for the problem by the following set of definitions.

### 13.1.1 Compatible requirements

We define two requirements to be compatible if the units involved are such that it is possible for a lesson, or part of a lesson, from each to occur at the same period.

Example

In the example problem requirement 10 is compatible with requirements 140, 160, 165, 180, 210, 220, 225, 240 and 250.

### 13.1.2 A complete arrangement

We define a complete arrangement to be a set of requirements such that:-

i) the requirements in the set are compatible in pairs; and ii) no requirement which is compatible with all members of the set is absent from the set.

Example

In the example problem we can start to form an arrangement by selecting requirement 10. This rules out requirements 20, 30, 40, 45, 50, 60, 70, 80, 90, 100, 110, 120, 130, 150, 170, 175, 190, 200, 130 and 235. The next member of the set must therefore be chosen from 140, 160, 165, 180, 210, 220, 225, 240 and 250. If we choose 140 then requirements 160, 165, 180, 210 and 240 are ruled out which leaves 220, 225 and 250. If we now take 220 then the remaining two requirements are ruled out thus 10, 140 and 220 form a complete arrangement.

### 13.1.3 A partial arrangement

We define a partial arrangement to be any part of a complete arrangement.

Example

The partial arrangements which can be formed from the complete arrangement discussed in the previous example are 10; 140; 220; 10, 140; 10, 220; and 140, 220.

13.1.4 A group

We define a group to be a set containing partial and/or complete arrangements such that:-

i) there is exactly one arrangement in the set for each period in the week but a particular arrangement may occur more than once if necessary;

and ii) each requirement occurs a total number of times equal to the total length of the lessons involved in it.

Example

For the example problem the 25 arrangements shown in figure 13.1 form a group. In this case every arrangement in the group is a complete arrangement.

13.1.5 An ordered group

We define an ordered group to be a permutation of the arrangements in a group so that when the arrangements are associated with the periods of the week they form a time-table in which all the constraints inherent in the model are satisfied.

Example

In the example problem a suitable ordering of the group in figure 13.1 is shown in figure 13.2. This is in fact just the solution given in figure 2.4.

13.2 Acceptable arrangements

Let us for the moment forget about the process to be used for the generation of an ordered group and consider some of the properties which its members must have.

The arrangement to be associated with period p must obey the following set of rules.

13.2.1 Rule F1

It must employ any unit which may not have a free period at period p.

13.2.2 Rule F2

It must not employ any unit which must have a free period at period p.

13.2.3 Rule F3

It must include any requirement in which a lesson is essential at period p.

13.2.4 Rule F4

It must not include any requirement in which a lesson is forbidden at period p.

13.2.5 Rule F5

It must not include any requirement involving multiple period lessons unless the remaining periods of the lesson can be or have been included in the arrangements for periods before or after period p.

We will refer to an arrangement which obeys these rules as being acceptable at period p.

Assuming that the selection process determines the arrangements to be associated with the periods one at a time, there will be, at any stage in the process, a finite number of acceptable arrangements for each period for which an arrangement has not been selected. Let us use $A_{0,0}$ to denote the set of arrangements which are initially acceptable at at least one period and $A_{p,s}$ to denote the subset of $A_{0,0}$ whose members remain acceptable at period p when arrangements have been selected for s periods. $A_{p,0}$ is then the subset of $A_{0,0}$ whose members are initially acceptable at period p.

[NOTE: In using this notation in the discussion which follows the parameter p will be the external period name and not the internal period number.]

The sets $A_{0,0}$ and $A_{p,0}$ are uniquely determined by the data for the problem but $A_{p,s}$ for $s > 0$ will depend on which s arrangements have already been selected and the s periods with which they have been associated.

In practice it is not a straightforward operation to enumerate the members of any of these sets. The problems involved are the same as those which prevent a perfect set of rules from being written down for cancelling elements in array PR. We must be content to work with approximations $A'_{0,0}$ and $A'_{p,s}$ which are super-sets to $A_{0,0}$ and $A_{p,s}$ i.e. each contains the members of the exact set but may contain additional members which cannot easily be seen to be unacceptable.

13.3 Arrangement counts

A useful parameter in determining the difficulty to be

expected in assigning the lessons in a given requirement is obtained by counting the number of times the requirement name occurs in the set $A_{0,0}$. We will use $C_{r,0,0}$ to denote this quantity for requirement r and will refer to it as the initial arrangement count.

A related count is the number of times requirement r occurs in $A_{p,s}$, to be denoted by $C_{r,p,s}$ and called the current arrangement count for requirement r at period p after s selections.

In practice the quantities obtained are upper bounds $C'_{r,0,0}$ and $C'_{r,p,s}$ since the sets from which they can be calculated are only approximate.

[Note: In using this notation in the discussion which follows the parameters r and p will be the external requirement name and the external period name respectively.]

## 13.4 Illustrations using the example problem

To clarify the ideas expressed so far in this chapter we now consider the example problem.

### 13.4.1 An approximation $A'_{0,0}$ to $A_{0,0}$

We first note that items 10,20,30 and 40 must be employed at every period of the week (figure 5.3). We can, therefore obtain an approximation to $A_{0,0}$ by generating all complete and partial arrangements which employ at least these four units. This produces 248 arrangements, as tabulated in figure 13.3. All of these are in fact complete arrangements. Figure 13.4 tabulates $C_{r,0,0}$ for this set.

There are many other complete arrangements for this problem which are not members of $A'_{0,0}$. Some of these, with a list of the units which they employ, are shown in figure 13.5.

### 13.4.2 An approximation $A'_{11,0}$ to $A_{11,0}$

To obtain $A'_{11,0}$ from $A'_{0,0}$ we consider in turn the rules given in 12.2:

Rule F1.: Initially only 10, 20, 30 and 40 must be employed at this period thus the method of forming $A'_{0,0}$ has satisfied this rule.

Rule F2 : From the Fixed Free Period List (figure 5.3) we see that one unit from item 100 must be free at period 11. Now requirements 40 and 45 employ both units of item

100 thus all arrangements involving these requirements must be eliminated. This removes 24 arrangements.

Rule F3 : Initially no lesson is essential at period 11.

Rule F4 : From the Pre-Assignment List (figure 5.3) we can see that lessons in requirements 150 and 200 are pre-assigned to periods on the same day as period 11 thus, since neither of these requirements needs more than one lesson per day, we must eliminate all arrangements involving them. This includes some of those already eliminated but also removes a further 62.

Rule F5 : From the period patterns for day 1 (figure 5.3) we see that any 2-period lesson which starts at period 11 must be completed at period 12. Now consider the arrangement 10, 160, 220 which is the fifth member of $A'_{0,0}$. Requirement 10 involves 1-period lessons but 160 and 220 both involve 2-period lessons (figure 5.3) thus if this arrangement is associated with period 11 then a different arrangement which does not involve requirement 10 but which does involve requirements 160 and 220 must be associated with period 12. No such alternative arrangement exists thus 10, 160, 220 is not acceptable at period 11. Similar considerations eliminate 10 other members of $A'_{0,0}$.

The set of 151 arrangements which remain as $A'_{11,0}$ are those marked X on figure 13.3. There are more complex considerations which eliminate many more arrangements from this set.

13.4.3 An approximation $A'_{15,0}$ to $A_{15,0}$

At period 15 $A'_{15,0}$ is found from $A_{0,0}$ simply by applying Rule F3. From the Pre-Assignment List we see that a lesson from requirement 150 is pre-assigned to period 15, thus only arrangements including this requirement are acceptable. From figure 13.4 we find that $C_{150,0,0} = 4$ thus $A'_{15,0}$ can contain at most four arrangements. The four acceptable ones are marked Y in figure 13.3. Application of the other rules does not further reduce this set.

13.4.4 An approximation $A'_{34,0}$ to $A_{34,0}$

At period 34 Rule F2 is the one which eliminates a large number of the members of $A_{0,0}$. From the Fixed Free Period

List we see that item 60 must have a free period at period 34. Noting, from the listing of Documents Two, Three and Four (figure 5.3) that item 60 is involved in requirements 10, 150, 190, 200, 230 and 235 and obtaining $C_{r,0,0}$ for each of these requirements from figure 13.4, we see that a total of 168 arrangements are eliminated. Application of the other rules does nor eliminate any further arrangements. The remaining 80 arrangements, which form $A'_{34,0}$ are marked Z in figure 13.3.

13.4.5 An approximation $A'_{14,1}$ to $A_{14,1}$

Let us suppose that the arrangement 40, 150, 210 is selected for period 15 then the number of arrangements acceptable at several other periods is changed. In particular we consider period 14.

First of all requirement 40 involves 2-period lessons thus the lesson already assigned to period 15 must commence at period 14. Secondly, from the Pre-Assignment List (figure 5.3) a lesson from requirement 200 is essential at period 14. By Rule F3 we then find that only two arrangements are acceptable. These are 40, 140, 200 and 40, 180, 200 thus these are the members of $A'_{14,1}$.

If an alternative choice is made at period 15 then the members of $A'_{14,1}$ are completely different.

13.4.6 The initial arrangement counts at period 15

In 13.4.3 we showed that the set $A'_{15,0}$ contains only the four arrangements:-

| | | |
|---|---|---|
| 40 | 150 | 210 |
| 40 | 150 | 240 |
| 45 | 150 | 210 |
| 45 | 150 | 240 |

The initial arrangement counts for all the requirements at period 15 can easily be found from these arrangements and are tabulated in figure 13.6.

Now clearly if $C_{r,p,s}$ is zero then the assignment of a lesson from requirement r at period p is impossible. This array therefore tells us some elements in array PR which can be cancelled. The operation of making the pre-assignment and applying the rules for cancellations in array PR given in Chapter

Seven would of course also have cancelled some elements in array PR but there are 8 elements, marked by arrows in figure 13.6, which would not have been cancelled by this process. The calculation of the arrangement counts has therefore given us a further method of finding those elements in array PR which appear to be available but are not. If such counts could be obtained at every stage of a time-table construction process we would be one step nearer the perfect set of rules discussed in 7.6. Unfortunately the arrangement counts could only be obtained with a relatively large amount of effort.

13.5 An arrangement generator program

As we have seen in 13.4.1 an approximation to $A_{0,0}$ can be obtained by generating all complete and partial arrangements which employ at least those units which may not have a free period at any period i.e. units of items for which TEA[i] = 0 initially. As a first stage in the investigation of the use of arrangements in time-table production a program was written to generate this set and to calculate the corresponding arrangement counts. The information in figures 13.3 and 13.4 are examples of its output.

Application of this program to a few problems quickly showed that any method of time-table construction which requires explicit formation of $A_{0,0}$ is bound to suffer from the disadvantage that the total number of initially acceptable arrangements is in general very large and storage of them within the machine in a rapidly accessible form is a problem.

Also the time taken for the process was found to be comparable to, and in some cases greater than, the time taken to obtain a complete solution using the Main Program. The example problem is not typical in that its tightly woven requirements greatly reduce the number of possible arrangements.

Thus, however useful the facility to examine all possible arrangements for a given problem simultaneously may be, it is not in general a practical proposition. A similar conclusion might be reached simply by observing that a finished time-table contains only one arrangement for each period thus the generation of hundreds, thousands or even tens of thousands in order to find thirty to forty suitable ones is clearly inefficient. In fact,

bearing in mind that the same arrangement can sometimes be used several times in a time-table, the total number of different arrangements used may be even less than one per period.

13.6 A group generator program

A second stage in the investigation of the use of arrangements in time-table production was to follow the suggestion made by Lawrie in his 1966 paper [7], namely to "separate the problem of finding a set of arrangements which will give each line its correct number of periods from the construction of a good time-table from the arrangements by permuting them". A program was therefore written to generate groups of arrangements where a group is as defined in 13.1.4.

This program was applied to several test cases and an example of a group produced for the example problem is shown in figure 13.7.

A few moments examination of this group shows that it cannot be permuted to give anything approaching a good time-table. The most obvious difficulty is the pairing together of arrangements to form the correct 2-period lessons without at the same time producing other unwanted ones.

Example

The four arrangements in figure 13.7 which involve requirement 40 are:-

| | | |
|---|---|---|
| 40 | 150 | 210 |
| 40 | 150 | 210 |
| 40 | 150 | 210 |
| 40 | 180 | 240 |

When these are grouped in pairs to form the two 2-period lessons needed in requirement 40 2-period lessons are also created for requirements 150 and 210. In fact from the group given in figure 13.7 only one 2-period lesson can be formed correctly.

Although the program was such that it could, if given sufficient time, produce all possible groups the chances of obtaining an acceptable group seem very remote. One possible solution to this problem is to include some further constraints in the definition of a group.

## 13.7 A modified group generator program

To avoid the difficulty with 2-period lessons in the groups generated by the program discussed in 13.6 a modified definition of a group was devised.

### 13.7.1 Modified definition of a group

We redefine a group to be a set containing partial and/or complete arrangements such that:-

i) there is exactly one arrangement in the set for each period in the week but a particular arrangement may occur more than once if necessary;

ii) each requirement occurs a total number of times equal to the total length of the lessons involved in it;

and iii) the total number of arrangements $TNA[r_1,r_2]$ in the set in which any pair of requirements $r_1$ and $r_2$ both occur satisfies the condition:-

$$TNA[r_1,r_2] \leq \text{Min } N[r_1],N[r_2] \text{ .Min } M[r_1],M[r_2]$$

Relatively minor modifications to the group generator program enabled it to produce groups satisfying this new constraint and one of the groups produced for the example problem is shown in figure 13.8.

Although this modified group is such that the necessary 2-period lessons can be formed without at the same time creating unwanted ones it is still not capable of being permuted to give a good time-table as we can demonstrate very simply.

First of all consider the two arrangements which contain requirement 50:-

| | | | |
|---|---|---|---|
| 50 | 90 | 130 | 210 |
| 50 | 80 | 175 | 230 |

They must be associated with consecutive periods to form the necessary 2-period lesson. The second one, however, includes requirement 230 which also involves a 2-period lesson thus the next arrangement must be:-

| | | | |
|---|---|---|---|
| 20 | 100 | 130 | 230 |

Similar additions must be made first to complete the 2-period lesson in requirement 100 and finally to complete one of the 2-period lessons in requirement 170 thus producing the following set of 5 arrangements:-

| | | | |
|---|---|---|---|
| 50 | 90 | 130 | 210 |
| 50 | 80 | 175 | 230 |
| 20 | 100 | 130 | 230 |
| 30 | 100 | 170 | 240 |
| 60 | 80 | 170 | 235 |

Now this set of arrangements fits into one complete day of the time-table but there are the following faults:-

i) A 2-period lesson starts at the second period of the day which is not allowed by the period patterns.

ii) Requirements 10, 140 and 200, each of which should have exactly one lesson per day, do not appear.

iii) Requirement 130, which should have exactly one lesson per day, appears twice.

It was, of course, only good luck which prevented the set of arrangements from growing to more than five. In larger problems, where each arrangement involves many parts of multiple period lessons, enforced stringing of arrangements would probably produce even larger sets. Since the school day rarely allows for more than three or four consecutive periods this would invalidate most of the groups produced. Further, for the example problem the fixed free periods, forbidden assignments and pre-assignments are not very numerous and they do not conflict with the group shown in figure 13.8. If, however, we were to add the pre-assignment 20, 34, i.e. at the same period as one of the fixed free periods for item 60, then there is no acceptable arrangement in figure 13.8 for period 34.

It would, therefore, once again appear that even if the program were allowed to generate a large number of modified groups the chance of obtaining an acceptable one seems very small. Similar conclusions were drawn from the results of all the problems tested. The question arises as to whether the chance could be further improved by more constraints in the definition of a group. Looking at the faults in the above example we see that such further constraints would have to involve at least the following factors:-

i) the occurrence of part of a multiple period lesson in a selected arrangement;

ii) the period patterns;

iii) the distribution laws;

iv) the Fixed Free Period List;

v) the Forbidden Assignment List;

and vi) the Pre-Assignment List.

This in effect means that each arrangement in the group formed will have to be associated with a particular period when it is selected. In other words the group formed will be an ordered group without the need for a permutation stage.

13.8 Conclusion

From the results discussed in 13.6 and 13.7 we must conclude that Lawrie's idea of obtaining a time-table in two stages is not a process which is likely to be useful in practice. This does not mean that arrangements cannot be usefully employed in time-table construction. If we apply the term 'horizontal constraints' to the conditions which must be satisfied at a given period of the time-table, and the term 'vertical constraints' to the conditions which must be satisfied over the whole week by the lessons assigned in a given requirement or associated set of requirements, then we can say that the use of arrangements is a powerful tool for ensuring that the horizontal constraints are met but provides no easy method of applying the vertical constraints.

One method of putting these ideas into practice in the existing Main Program would be to test, when each assignment is made, that at least one acceptable arrangement still exists for the period or periods concerned. This would not guarantee the eventual completion of that period of the time-table but it might improve the overall power of the program to detect impossible situations, especially of the type discussed in 11.3.1, and hence the overall performance.

This suggestion has not been tested out in the Main Program thus the cost in execution time which it would involve has not been assessed. To apply the test after every assignment would probably prove too expensive but tests when say 75% of the units had been employed at a given period could prove to be almost as useful and much less time consuming.

Figure 13.1

A group

| | | | |
|---|---|---|---|
| 10 | 140 | 220 | |
| 10 | 140 | 225 | |
| 10 | 160 | 225 | |
| 10 | 160 | 240 | |
| 10 | 165 | 240 | |
| 20 | 90 | 130 | 250 |
| 20 | 90 | 160 | 235 |
| 20 | 90 | 170 | 200 |
| 20 | 100 | 180 | 200 |
| 20 | 110 | 175 | 230 |
| 30 | 80 | 130 | 200 |
| 30 | 80 | 160 | 200 |
| 30 | 80 | 190 | |
| 30 | 100 | 130 | 235 |
| 30 | 120 | 170 | 210 |
| 40 | 140 | 200 | |
| 40 | 140 | 225 | |
| 40 | 150 | 210 | |
| 40 | 150 | 210 | |
| 45 | 140 | 230 | |
| 45 | 150 | 240 | |
| 50 | 80 | 170 | 220 |
| 50 | 90 | 170 | 210 |
| 60 | 80 | 130 | 235 |
| 70 | 90 | 130 | 210 |

Figure 13.2

An ordered group

| | | | |
|---|---|---|---|
| 10 | 165 | 240 | |
| 20 | 90 | 130 | 250 |
| 30 | 80 | 190 | |
| 40 | 140 | 200 | |
| 40 | 150 | 210 | |
| 50 | 90 | 170 | 210 |
| 50 | 80 | 170 | 220 |
| 10 | 140 | 220 | |
| 20 | 100 | 180 | 200 |
| 30 | 100 | 130 | 235 |
| 30 | 80 | 130 | 200 |
| 20 | 90 | 160 | 235 |
| 10 | 160 | 240 | |
| 40 | 140 | 225 | |
| 40 | 150 | 210 | |
| 60 | 80 | 130 | 235 |
| 10 | 140 | 225 | |
| 45 | 150 | 240 | |
| 20 | 90 | 170 | 200 |
| 30 | 120 | 170 | 210 |
| 10 | 160 | 225 | |
| 30 | 80 | 160 | 200 |
| 70 | 90 | 130 | 210 |
| 45 | 140 | 230 | |
| 20 | 110 | 175 | 230 |

Figure 13.3

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 10 | 140 | 220 | | X | |
| 2 | 10 | 140 | 225 | | X | |
| 3 | 10 | 140 | 250 | | X | |
| 4 | 10 | 160 | 210 | | X | |
| 5 | 10 | 160 | 220 | | | |
| 6 | 10 | 160 | 225 | | X | |
| 7 | 10 | 160 | 240 | | X | |
| 8 | 10 | 165 | 210 | | X | |
| 9 | 10 | 165 | 220 | | X | |
| 10 | 10 | 165 | 225 | | X | |
| 11 | 10 | 165 | 240 | | X | |
| 12 | 10 | 180 | 210 | | X | |
| 13 | 10 | 180 | 240 | | X | |
| 14 | 20 | 90 | 130 | 200 | | |
| 15 | 20 | 90 | 130 | 235 | X | |
| 16 | 20 | 90 | 130 | 150 | X | Z |
| 17 | 20 | 90 | 160 | 200 | | |
| 18 | 20 | 90 | 160 | 235 | X | |
| 19 | 20 | 90 | 165 | 200 | | |
| 20 | 20 | 90 | 165 | 235 | X | |
| 21 | 20 | 90 | 170 | 200 | | |
| 22 | 20 | 90 | 170 | 235 | X | |
| 23 | 20 | 90 | 170 | 250 | X | Z |
| 24 | 20 | 90 | 175 | 200 | | |
| 25 | 20 | 90 | 175 | 235 | X | |
| 26 | 20 | 90 | 175 | 250 | X | Z |
| 27 | 20 | 90 | 190 | | X | |
| 28 | 20 | 100 | 130 | 200 | | |
| 29 | 20 | 100 | 130 | 220 | | Z |
| 30 | 20 | 100 | 130 | 225 | X | Z |
| 31 | 20 | 100 | 130 | 230 | X | |
| 32 | 20 | 100 | 130 | 235 | X | |
| 33 | 20 | 100 | 170 | 200 | | |
| 34 | 20 | 100 | 170 | 220 | | Z |
| 35 | 20 | 100 | 170 | 225 | X | Z |
| 36 | 20 | 100 | 170 | 230 | X | |
| 37 | 20 | 100 | 170 | 235 | X | |
| 38 | 20 | 100 | 175 | 200 | | |
| 39 | 20 | 100 | 175 | 220 | | Z |
| 40 | 20 | 100 | 175 | 225 | X | Z |
| 41 | 20 | 100 | 175 | 230 | X | |
| 42 | 20 | 100 | 175 | 235 | X | |
| 43 | 20 | 100 | 180 | 200 | | |
| 44 | 20 | 100 | 180 | 230 | X | |
| 45 | 20 | 100 | 180 | 235 | X | |
| 46 | 20 | 110 | 130 | 200 | | |
| 47 | 20 | 110 | 130 | 220 | X | Z |
| 48 | 20 | 110 | 130 | 225 | X | Z |
| 49 | 20 | 110 | 130 | 230 | X | |
| 50 | 20 | 110 | 130 | 235 | X | |
| 51 | 20 | 110 | 170 | 200 | | |
| 52 | 20 | 110 | 170 | 220 | X | Z |
| 53 | 20 | 110 | 170 | 225 | X | Z |
| 54 | 20 | 110 | 170 | 230 | X | |
| 55 | 20 | 110 | 170 | 235 | X | |
| 56 | 20 | 110 | 175 | 200 | | |
| 57 | 20 | 110 | 175 | 220 | X | Z |
| 58 | 20 | 110 | 175 | 225 | X | Z |
| 59 | 20 | 110 | 175 | 230 | X | |
| 60 | 20 | 110 | 175 | 235 | X | |
| 61 | 20 | 120 | 130 | 200 | | |
| 62 | 20 | 120 | 130 | 220 | X | Z |
| 63 | 20 | 120 | 130 | 230 | X | |
| 64 | 20 | 120 | 170 | 200 | | |
| 65 | 20 | 120 | 170 | 220 | X | Z |
| 66 | 20 | 120 | 170 | 230 | X | |
| 67 | 20 | 120 | 175 | 200 | | |
| 68 | 20 | 120 | 175 | 220 | X | Z |
| 69 | 20 | 120 | 175 | 230 | X | |
| 70 | 20 | 120 | 180 | 200 | | |
| 71 | 20 | 120 | 180 | 230 | X | |
| 72 | 30 | 80 | 130 | 200 | | |
| 73 | 30 | 80 | 130 | 235 | X | |
| 74 | 30 | 80 | 130 | 250 | X | Z |
| 75 | 30 | 80 | 160 | 200 | | |
| 76 | 30 | 80 | 160 | 235 | X | |
| 77 | 30 | 80 | 165 | 200 | | |
| 78 | 30 | 80 | 165 | 235 | X | |
| 79 | 30 | 80 | 170 | 200 | | |
| 80 | 30 | 80 | 170 | 235 | X | |
| 81 | 30 | 80 | 170 | 250 | X | Z |
| 82 | 30 | 80 | 175 | 200 | | |
| 83 | 30 | 80 | 175 | 235 | X | |
| 84 | 30 | 80 | 175 | 250 | X | Z |
| 85 | 30 | 80 | 190 | | X | |
| 86 | 30 | 100 | 130 | 200 | | |
| 87 | 30 | 100 | 130 | 210 | X | Z |
| 88 | 30 | 100 | 130 | 235 | X | |
| 89 | 30 | 100 | 130 | 240 | X | Z |
| 90 | 30 | 100 | 140 | 200 | | |
| 91 | 30 | 100 | 140 | 235 | X | |
| 92 | 30 | 100 | 170 | 200 | | |
| 93 | 30 | 100 | 170 | 210 | X | Z |
| 94 | 30 | 100 | 170 | 235 | X | |
| 95 | 30 | 100 | 170 | 240 | X | Z |
| 96 | 30 | 100 | 175 | 200 | | |
| 97 | 30 | 100 | 175 | 210 | X | Z |
| 98 | 30 | 100 | 175 | 235 | X | |
| 99 | 30 | 100 | 175 | 240 | X | Z |
| 100 | 30 | 110 | 130 | 200 | | |

Figure 13.3 continued

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 101 | 30 | 110 | 130 | 210 | X Z | 151 | 50 | 80 | 130 | 230 | |
| 102 | 30 | 110 | 130 | 235 | X | 152 | 50 | 80 | 130 | 235 | X |
| 103 | 30 | 110 | 130 | 240 | X Z | 153 | 50 | 80 | 170 | 200 | |
| 104 | 30 | 110 | 140 | 200 | | 154 | 50 | 80 | 170 | 220 | Z |
| 105 | 30 | 110 | 140 | 235 | X | 155 | 50 | 80 | 170 | 225 | X Z |
| 106 | 30 | 110 | 170 | 200 | | 156 | 50 | 80 | 170 | 230 | |
| 107 | 30 | 110 | 170 | 210 | X Z | 157 | 50 | 80 | 170 | 235 | X |
| 108 | 30 | 110 | 170 | 235 | X | 158 | 50 | 80 | 175 | 200 | |
| 109 | 30 | 110 | 170 | 240 | X Z | 159 | 50 | 80 | 175 | 220 | Z |
| 110 | 30 | 110 | 175 | 200 | | 160 | 50 | 80 | 175 | 225 | X Z |
| 111 | 30 | 110 | 175 | 210 | X Z | 161 | 50 | 80 | 175 | 230 | |
| 112 | 30 | 110 | 175 | 235 | X | 162 | 50 | 80 | 175 | 235 | X |
| 113 | 30 | 110 | 175 | 240 | X Z | 163 | 50 | 80 | 180 | 200 | |
| 114 | 30 | 120 | 130 | 200 | | 164 | 50 | 80 | 180 | 230 | |
| 115 | 30 | 120 | 130 | 210 | X Z | 165 | 50 | 80 | 180 | 235 | X |
| 116 | 30 | 120 | 130 | 240 | X Z | 166 | 50 | 90 | 130 | 200 | |
| 117 | 30 | 120 | 140 | 200 | | 167 | 50 | 90 | 130 | 210 | X Z |
| 118 | 30 | 120 | 170 | 200 | | 168 | 50 | 90 | 130 | 235 | X |
| 119 | 30 | 120 | 170 | 210 | X Z | 169 | 50 | 90 | 130 | 240 | X Z |
| 120 | 30 | 120 | 170 | 240 | X Z | 170 | 50 | 90 | 140 | 200 | |
| 121 | 30 | 120 | 175 | 200 | | 171 | 50 | 90 | 140 | 235 | X |
| 122 | 30 | 120 | 175 | 210 | X Z | 172 | 50 | 90 | 170 | 200 | |
| 123 | 30 | 120 | 175 | 240 | X Z | 173 | 50 | 90 | 170 | 210 | X Z |
| 124 | 40 | 140 | 200 | | | 174 | 50 | 90 | 170 | 235 | X |
| 125 | 40 | 140 | 220 | | Z | 175 | 50 | 90 | 170 | 240 | X Z |
| 126 | 40 | 140 | 225 | | Z | 176 | 50 | 90 | 175 | 200 | |
| 127 | 40 | 140 | 230 | | | 177 | 50 | 90 | 175 | 210 | X Z |
| 128 | 40 | 140 | 235 | | | 178 | 50 | 90 | 175 | 235 | X |
| 129 | 40 | 150 | 210 | | Y | 179 | 50 | 90 | 175 | 240 | X Z |
| 130 | 40 | 150 | 240 | | Y | 180 | 60 | 80 | 130 | 200 | |
| 131 | 40 | 180 | 200 | | | 181 | 60 | 80 | 130 | 230 | X |
| 132 | 40 | 180 | 210 | | Z | 182 | 60 | 80 | 130 | 235 | X |
| 133 | 40 | 180 | 230 | | | 183 | 60 | 80 | 160 | 200 | |
| 134 | 40 | 180 | 235 | | | 184 | 60 | 80 | 160 | 230 | X |
| 135 | 40 | 180 | 240 | | Z | 185 | 60 | 80 | 160 | 235 | X |
| 136 | 45 | 140 | 200 | | | 186 | 60 | 80 | 165 | 200 | |
| 137 | 45 | 140 | 220 | | Z | 187 | 60 | 80 | 165 | 230 | X |
| 138 | 45 | 140 | 225 | | Z | 188 | 60 | 80 | 165 | 235 | X |
| 139 | 45 | 140 | 230 | | | 189 | 60 | 80 | 170 | 200 | |
| 140 | 45 | 140 | 235 | | | 190 | 60 | 80 | 170 | 230 | X |
| 141 | 45 | 150 | 210 | | Y | 191 | 60 | 80 | 170 | 235 | X |
| 142 | 45 | 150 | 240 | | Y | 192 | 60 | 80 | 175 | 200 | |
| 143 | 45 | 180 | 200 | | | 193 | 60 | 80 | 175 | 230 | X |
| 144 | 45 | 180 | 210 | | Z | 194 | 60 | 80 | 175 | 235 | X |
| 145 | 45 | 180 | 230 | | | 195 | 60 | 80 | 190 | | X |
| 146 | 45 | 180 | 235 | | | 196 | 60 | 100 | 130 | 200 | |
| 147 | 45 | 180 | 240 | | Z | 197 | 60 | 100 | 130 | 210 | X Z |
| 148 | 50 | 80 | 130 | 200 | | 198 | 60 | 100 | 130 | 230 | X |
| 149 | 50 | 80 | 130 | 220 | Z | 199 | 60 | 100 | 130 | 235 | X |
| 150 | 50 | 80 | 130 | 225 | X Z | 200 | 60 | 100 | 130 | 240 | X Z |

Figure 13.3 continued

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 201 | 60 | 100 | 140 | 200 | | 225 | 60 | 120 | 175 | 210 | X | Z |
| 202 | 60 | 100 | 140 | 230 | X | 226 | 60 | 120 | 175 | 230 | X | |
| 203 | 60 | 100 | 140 | 235 | X | 227 | 60 | 120 | 175 | 240 | X | Z |
| 204 | 60 | 100 | 170 | 200 | | 228 | 70 | 80 | 130 | 200 | | |
| 205 | 60 | 100 | 170 | 210 | X Z | 229 | 70 | 80 | 130 | 220 | X | Z |
| 206 | 60 | 100 | 170 | 230 | X | 230 | 70 | 80 | 130 | 230 | X | |
| 207 | 60 | 100 | 170 | 235 | X | | | | | | | |
| 208 | 60 | 100 | 170 | 240 | X Z | 231 | 70 | 80 | 170 | 200 | | |
| 209 | 60 | 100 | 175 | 200 | | 232 | 70 | 80 | 170 | 220 | X | Z |
| 210 | 60 | 100 | 175 | 210 | X Z | 233 | 70 | 80 | 170 | 230 | X | |
| | | | | | | 234 | 70 | 80 | 175 | 200 | | |
| 211 | 60 | 100 | 175 | 230 | X | 235 | 70 | 80 | 175 | 220 | X | Z |
| 212 | 60 | 100 | 175 | 235 | X | 236 | 70 | 80 | 175 | 230 | X | |
| 213 | 60 | 100 | 175 | 240 | X Z | 237 | 70 | 80 | 180 | 200 | | |
| 214 | 60 | 120 | 130 | 200 | | 238 | 70 | 80 | 180 | 230 | X | |
| 215 | 60 | 120 | 130 | 210 | X Z | 239 | 70 | 90 | 130 | 200 | | |
| 216 | 60 | 120 | 130 | 230 | X | 240 | 70 | 90 | 130 | 210 | X | Z |
| 217 | 60 | 120 | 130 | 240 | X Z | | | | | | | |
| 218 | 60 | 120 | 140 | 200 | | 241 | 70 | 90 | 130 | 240 | X | Z |
| 219 | 60 | 120 | 140 | 230 | X | 242 | 70 | 90 | 140 | 200 | | |
| 220 | 60 | 120 | 170 | 200 | | 243 | 70 | 90 | 170 | 200 | | |
| | | | | | | 244 | 70 | 90 | 170 | 210 | X | Z |
| 221 | 60 | 120 | 170 | 210 | X Z | 245 | 70 | 90 | 170 | 240 | X | Z |
| 222 | 60 | 120 | 170 | 230 | X | 246 | 70 | 90 | 175 | 200 | | |
| 223 | 60 | 120 | 170 | 240 | X Z | 247 | 70 | 90 | 175 | 210 | X | Z |
| 224 | 60 | 120 | 175 | 200 | | 248 | 70 | 90 | 175 | 240 | X | Z |

Figure 13.4

| Requirement Number r | $C_{r,0,0}$ | Requirement Number r | $C_{r,0,0}$ | Requirement Number r | $C_{r,0,0}$ |
|---|---|---|---|---|---|
| 10 | 13 | 100 | 50 | 180 | 22 |
| 20 | 58 | 110 | 29 | 190 | 3 |
| 30 | 52 | 120 | 35 | 200 | 66 |
| 40 | 12 | 130 | 57 | 210 | 28 |
| 45 | 12 | 140 | 26 | 220 | 20 |
| 50 | 32 | 150 | 4 | 225 | 14 |
| 60 | 48 | 160 | 11 | 230 | 36 |
| 70 | 21 | 165 | 11 | 235 | 46 |
| 80 | 59 | 170 | 57 | 240 | 28 |
| 90 | 38 | 175 | 57 | 150 | 7 |

Figure 13.5

| Arrangement | | | The items employed | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 20 | 160 | 225 | 10 | 30 | 40 | 70 | 80 | 90 | 100 | 130 |
| 30 | 140 | 250 | 10 | 30 | 40 | 70 | 80 | 90 | 110 | 120 |
| 50 | 80 | 150 | 10 | 20 | 30 | 60 | 70 | 80 | 90 | 110 |
| 60 | 160 | 210 | 10 | 30 | 40 | 70 | 80 | 90 | 100 | 120 |
| 70 | 180 | 240 | 10 | 30 | 40 | 70 | 80 | 90 | 120 | 130 |

Note: These arrangements are not members of $A_{0,0}$ because they do not employ all the classes i.e. items 10, 20, 30 and 40.

Figure 13.6

| IRN | ERN | $C_{r,15,0}$ | |
|---|---|---|---|
| 1 | 10 | 0 | |
| 2 | 20 | 0 | ← |
| 3 | 30 | 0 | |
| 4 | 40 | 2 | |
| 5 | 45 | 2 | |
| 6 | 50 | 0 | |
| 7 | 60 | 0 | ← |
| 8 | 70 | 0 | ← |
| 9 | 80 | 0 | ← |
| 10 | 90 | 0 | |
| 11 | 100 | 0 | ← |
| 12 | 110 | 0 | ← |
| 13 | 120 | 0 | ← |
| 14 | 130 | 0 | |
| 15 | 140 | 0 | |
| 16 | 150 | 4 | |
| 17 | 160 | 0 | |
| 18 | 165 | 0 | |
| 19 | 170 | 0 | |
| 20 | 175 | 0 | |
| 21 | 180 | 2 | |
| 22 | 190 | 0 | |
| 23 | 200 | 0 | |
| 24 | 210 | 2 | |
| 25 | 220 | 0 | |
| 26 | 225 | 0 | |
| 27 | 230 | 0 | |
| 28 | 235 | 0 | |
| 29 | 240 | 2 | |
| 30 | 250 | 0 | ← |

Figure 13.7

| | | | |
|---|---|---|---|
| 20 | 90 | 190 | |
| 40 | 150 | 210 | |
| 40 | 150 | 210 | |
| 40 | 150 | 210 | |
| 10 | 140 | 250 | |
| 10 | 160 | 225 | |
| 10 | 160 | 225 | |
| 10 | 160 | 210 | |
| 10 | 160 | 210 | |
| 20 | 90 | 165 | 235 |
| 40 | 180 | 240 | |
| 45 | 140 | 220 | |
| 45 | 140 | 220 | |
| 20 | 110 | 130 | 225 |
| 70 | 90 | 140 | 200 |
| 50 | 90 | 140 | 235 |
| 50 | 90 | 170 | 240 |
| 30 | 120 | 170 | 240 |
| 20 | 100 | 170 | 230 |
| 20 | 100 | 170 | 230 |
| 30 | 80 | 130 | 235 |
| 60 | 80 | 130 | 200 |
| 30 | 80 | 130 | 200 |
| 30 | 80 | 130 | 200 |
| 30 | 80 | 175 | 200 |

Figure 13.8

| | | | |
|---|---|---|---|
| 40 | 150 | 210 | |
| 40 | 150 | 210 | |
| 45 | 150 | 210 | |
| 10 | 160 | 225 | |
| 10 | 160 | 225 | |
| 10 | 140 | 225 | |
| 10 | 140 | 220 | |
| 10 | 140 | 250 | |
| 40 | 140 | 200 | |
| 40 | 140 | 200 | |
| 20 | 90 | 160 | 200 |
| 20 | 90 | 160 | 200 |
| 20 | 90 | 190 | |
| 45 | 180 | 210 | |
| 50 | 90 | 130 | 210 |
| 20 | 90 | 130 | 200 |
| 20 | 100 | 130 | 230 |
| 30 | 100 | 170 | 240 |
| 30 | 120 | 170 | 240 |
| 30 | 110 | 130 | 240 |
| 50 | 80 | 175 | 230 |
| 70 | 80 | 170 | 220 |
| 30 | 80 | 165 | 235 |
| 30 | 80 | 130 | 235 |
| 60 | 80 | 170 | 235 |

# CHAPTER FOURTEEN

## ANALYSIS AND SUGGESTIONS

### 14.0 Introduction

We have defined a model which enables us to describe a school for the purpose of making its time-table where the model itself defines very rigidly what we mean by a time-table. We have also described a program which implements the model, not in its full generality, but to a sufficient extent to deal with most real schools and we have discussed the ability of this program to produce the required results. At the end of Chapter 12 we concluded that it is not a satisfactory means of producing time-tables.

In this chapter we seek to analyse our approach and to suggest possible changes which might enable a time-table of some sort to be produced for every set of data. We make this analysis and these suggestions in the light of our own findings and with the aid of information published by others working in the same field. The time-tabling systems which are of particular interest are:-

i) The Ontario School Scheduling Program [9, 10, 11] - developed for the Ontario Institute for Studies in Education by J.Kates and Associates.

ii) The Newcastle-upon-Tyne University Program [2,12] - developed at Newcastle-upon-Tyne University by a group led by N.S.M.Cox and tested by the Local Government Operational Research Unit, Reading.

iii) The Norwegian Time-Tabling Program [14] - developed at the Norges Tekniske Hogskole, Trondheim, Norway by a group led by H.Michalsen.

iv) The Birmingham University Program [13] - developed at Birmingham University by A.M.Clementson and tested by the Local Government Operational Research Unit , Reading.

We will refer to these programs by the initials O.S.S.P., N.U.P.,N.T.P. and B.U.P. respectively and to the Local Government

Operational Research Unit by the initials L.G.O.R.U..

14.1 A basic fault in the approach

The ideal algorithm for making school time-tables needs only one basic ability. It is to be able to determine at any stage in constructing a time-table whether or not an acceptable solution still exists. In making this statement we pre-suppose that we know what we mean by 'an acceptable solution' and for the moment we retain the notion that the only acceptable solutions are of the type defined inherently in our model. The algorithm applied by the Main Program, because its rules for detecting impossible situations are not perfect, does not have this ability and the basic fault in our whole approach to the problem is that we have failed to make satisfactory allowances for this weakness.

Much of the development of the algorithm has, as we discussed in 11.4, been aimed at finding better and better methods of detecting impossible situations hence producing better and better approximations to the ideal algorithm but perhaps the time has come to admit that a sufficiently good approximation to the ideal algorithm cannot be achieved and that the basic approach should be changed to allow for this fact.

14.2 A basic change in the approach

If we cannot produce the ideal algorithm then there will always be circumstances when an impossible situation exists in a partial solution and reveals itself by the occurrence of a long sequence of retreats at a stage when it is too late to do anything about it. Such a situation may be due to an impossible situation inherent in the data or due to the assignments made by the algorithm but these two cases will sometimes be indistinguishable.

Since the assignment strategy in the algorithm is to make each chosen assignment in the tightest area of the time-table an impossible situation, caused by the assignments made, is most likely to be detected in the chain in which it arises and a single retreat has a very high chance of removing it. It is therefore not unreasonable to assume that any impossible situation which has not been resolved after one retreat has either been present from the start of the solution or at least for several chains. In the current program such a situation would cause

a long sequence of retreats and the eventual termination of the attempt to find a solution.

A basic change in the approach which would enable the program to deal with these impossible situations and proceed with the solution is to allow some form of relaxation of the constraints in such circumstances.

The constraints which lend themselves most readily to relaxation are those which determine:-

i) how lessons in the same or related subjects should be spread over the day and over the week;

ii) how multiple period lessons may be placed in relation to breaks in teaching for meals etc.;

and iii) that every lesson requested must appear in the time-table.

The relaxation of these constraints leads to badly spread lessons, split multiple periods and time-tables with some of the requested lessons missing but a solution with these faults may, to the school, be much more satisfactory than the information that because the computer cannot produce a perfect solution it has refused to produce any.

Before making suggestions for changes to introduce these types of relaxation we examine this aspect of the programs mentioned in 14.0 all of which have some means or other of relaxing constraints to get out of difficult situations.

## 14.3 Relaxation of constraints in other programs

### 14.3.1 The Ontario School Scheduling Program

The model for O.S.S.P. uses two types of requirements. A 'simple' requirement involves only one class and one teacher for single period lessons. All more complicated requirements, for multiple period lessons, setted lessons etc., are described in 'special' requirements. The system accepts that it is usually not possible to produce a time-table which satisfies every special requirement and provides in the data a means of assigning two priority indicators to each special requirement. The relevant information is given in the following extract from the Data Preparation Manual:-

"PRIORITY INDICATORS

i) Two priority indicators are associated with each Special Requirement. Each is a single letter or blank, and they are written on the Requirement Coding Form in the space provided after that for the requirement number and its suffix in the top right hand corner.

ii) The first priority indicator indicates the importance of the requirement, and can take the following values:

E = Essential
H = Highly desirable
D = Desirable
Blank

There are thus four categories of importance into which the requirement can be placed.

iii) Requirements should only be classed as Essential if they are in fact just that and must be made just as requested, in the final timetable.

iv) The second priority indicator indicates the first step of relaxation of the requirement in the event that it cannot be made as requested. It can take the following values:

P, R, G, Blank

The value P implies that any restrictions confining the requirement to certain periods of the day may be dropped, and that any double or triple period may be broken up.

The value R implies that any requirements for shared special rooms may be abandoned.

The value G implies that some of the teachers or classes grouped together by the requirement may be eliminated.

v) The two priority indicators may be assigned values independently of each other."

The program starts by breaking the problem for the week into a set of problems for the days and in so doing goes a long way towards obtaining what we call a normal distribution. However, if some of the daily problems cannot be solved the analyst is allowed to adjust the breakdown into days and may find it necessary to accept less than satisfactory distributions for

some requirements.

In solving the daily problems the computer uses the first priority indicator to arrange the requirements in an order of difficulty and, if in making the assignments, it cannot fit in a particular lesson it uses the second priority parameter to relax the constraints. If this still fails to resolve its difficulty it simply omits the troublesome lesson from the timetable. The analyst can then adjust the requirements or the priorities for a later run in the hope that it will reduce the number of lessons which are omitted. Lions [10] reports that the number of re-runs on each daily problem averages about 2 to 3.

14.3.2 The Newcastle-upon-Tyne University Program

The N.U.P. bases its methods of distribution on the assignment of a type code to each subject and a concept of 'soft' and 'hard' requirements. A typical type code scale is shown in figure 14.1 and the idea is that when considering distribution priority should be given first to lessons in subjects with the same type code, then to those with type codes differing by 1 etc. Thus using the scale shown it would spread all English and History lessons as well as possible and would put an Art or Latin lesson beside one of them rather than a French or Geography lesson. Setted classes involving options have to be given codes near the middle of the range. The idea of 'soft' and 'hard' requirements is that in soft requirements two single or multiple period lessons in the same subject may be allowed to coalesce if necessary but in a hard requirement they may not.

As far as multiple periods spanning breaks is concerned N.U.P. will allow a lesson to span a break or will leave the lesson out completely rather than split it into separate shorter lessons.

Figure 14.1

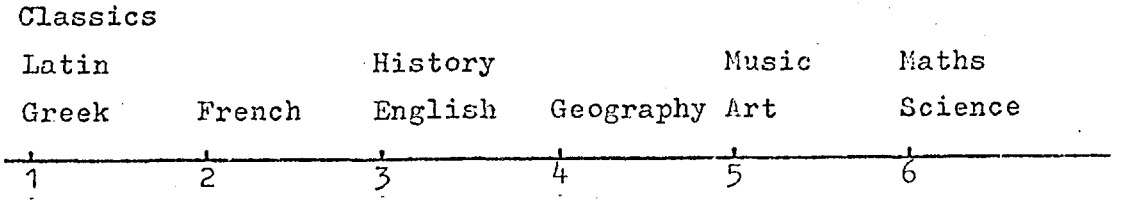


Figure 14.2

| No. of requirements per week | Distribution Type | Frequency |
|---|---|---|
| 2 | 11000 | 253 |
| | 20000 | 6 |
| 3 | 11100 | 88 |
| | 21000 | 4 |
| | 30000 | 0 |
| 4 | 11110 | 21 |
| | 21100 | 16 |
| | 22000 | 1 |
| | 31000 | 1 |
| | 40000 | 0 |
| 5 | 11111 | 8 |
| | 21110 | 9 |
| | 22100 | 4 |
| | others | 4 |

Note: The distribution type is given in the form of the no. of requirements in each day of a 5 day week. For example, with the 2 requirements per week case the code 20000 means that both requirements occur on the same day of the week, whereas, the code 11000 means that the two requirements occur on separate days of the week.

Figure 14.3

| | | |
|---|---|---|
| No. of Double Periods crossing | no breaks | 1537 |
| " " " " " | short break | 24 |
| " " " " " | long break | 4 |
| No. of Triple periods crossing | no break | 43 |
| " " " " " | short break | 14 * |
| " " " " " | long break | 1 |

*Note: This figure is higher than expected because in a few schools it was impossible to timetable a triple period without it crossing a break.

As an aid to the assessment of the success of these techniques we reproduce in figures 14.2 and 14.3 the results included in the L.G.O.R.U. report on N.U.P.[12]. Figure 14.2 is an overall summary for the distribution of subjects over the week based on several time-tables. In this figure the word 'requirement' means 'a single lesson'. Figure 14.3 is a similar summary of lessons placed across short and long breaks. Another relevant fact is that the program omitted an average of 12% of all lessons requested.

14.3.3 The Norwegian Time-tabling Program

Information on the procedures used by N.T.P. for relaxing constraints has not, to the best of the Author's knowledge, been published but a certain amount of information has been obtained from the data manual [14] and in conversation with Michalsen.

The main provision in the data for relaxing the constraints is the specification of one or more alternatives for each room type. For example, the data might state that if an ordinary classroom is not available then the best alternative is the Music Room and the next best alternative is an Art Room. During the construction of the time-table a failure due to the unavailability of the room type requested can then sometimes be resolved by substituting one of the alternatives.

N.T.P. also allows lessons to be omitted from the time-table in the hope that an analyst will be able to adjust the data so that they can be included in a later run. The adjustment is not based on formal information included in the data by the school staff but simply on the experience of the analyst to make only such changes as would be acceptable to the school.

14.3.4 The Birmingham University Program

Information on the B.U.P. is confined to an interim report [13] obtained through correspondence with L.G.O.R.U. This states that B.U.P. makes some effort to spread lessons correctly over the week but no details of how it is done are given. The L.G.O.R.U. considers it to be less successful than the N.U.P. system and on a scale where -1 represents 'all requirements spread badly' and +1 represents 'all requirements spread well' they assign a measure of 0.92 to N.U.P. and 0.85 to B.U.P. This,

however, is not unsatisfactory for on average B.U.P. only leaves out 8% of all lessons compared with the 12% left out by N.U.P.

In dealing with multiple period lessons B.U.P. regards the breaks as sacrosanct and will not assign lessons across them. It will, however, split a multiple period lesson rather than leave it out. Results show that the number of lessons actually split varies considerably from one school to another but the average of 36 per time-table seems unacceptably high.

The data documents for B.U.P. request information about teachers who may be interchanged but it is not stated whether this information is used by the computer or by the analyst.

14.4 Disadvantages of the methods for relaxing constraints

In the methods discussed in 14.3 there are two basic weapons for dealing with impossible situations. They are:-

i) the automatic variation of the constraints by the computer where its decisions are based on additional data;

and ii) the omission from the time-table of lessons which the computer cannot fit in.

Each of these, however powerful it may be, has some disadvantages. In i) the main disadvantage is that extra data has to be provided. This is not unreasonable since if the computer is to produce time-tables at least as good, from the point of view of the individual school, as those produced manually it should have the same information as the manual time-tabler has at his disposal. Much of this information unfortunately is in the form of experience or detailed knowledge of the whims of individual members of the school staff and is very difficult to present to the computer in coded form. A second disadvantage is the difficulty in controlling the extent to which the computer may use its power to vary the constraints. The fact that B.U.P. splits an average of 36 multiple period lessons per time-table is an example of using such power too freely.

In ii) the main disadvantage is that somehow or other all the lessons left out of the time-table have to be fitted in eventually and this, whether done manually or with the aid of the computer, is a difficult task. In fact, if the omitted lesson is a multiple period lesson or involves setted classes the manipulation needed to insert it could involve changes in

large areas of the time-table. Again, a second disadvantage is the difficulty in controlling the extent to which the computer can leave out lessons. The averages of 12% left out by N.U.P. and 8% left out by B.U.P. are surely much too high.

14.5 The need for modifications to completed time-tables

Before deciding which methods should be used to improve the performance of the Author's program there is one other valid consideration. This is that a time-table which is satisfactory at the beginning of a school year is likely to need changes before the first term is many days old and at all too frequent intervals during the school year due to unexpected changes in staff and to other unforeseen events. This means that there is a very good case for including in a school time-tabling system a program which can insert lessons in an already partly complete time-table and make other types of adjustments. If such a program could be developed then it could be used not only to adjust the time-table during the school year but also to adjust parts of the time-table during the actual construction process.

14.6 Suggestions for a revised approach

In the light of the above considerations the Author suggests that the most satisfactory method of overcoming the difficulties encountered would not be to adopt any of the methods used by the other programs discussed but to combine the existing powers of the program to make large sections of a time-table with the powers of a human to resolve difficulties by recourse to information not available to the computer. This could be achieved using modern on-line computing techniques by writing a program, which we will call the Adjustment Routine, to make changes to a partial or complete time-table under the control of the user via an on-line terminal.

Among the operations which this routine should be able to perform are:-

i) display, in whatever form is available, elements 1 to m of any one dimensional array;

ii) display elements 1 to m of row n of any two-dimensional array;

iii) display elements 1 to m of column n of any two-dimensional array;

iv) display day d of the item band for item i;

v) remove an already assigned lesson starting at period p in requirement r;

vi) assign a lesson starting in period p in requirement r irrespective of whether such an assignment breaks the normal constraints;

vii) remove requirement r;

viii) insert a new requirement;

ix) remove the ties on requirements $r_1$, $r_2$, ... $r_n$;

x) insert ties on requirements $r_1$, $r_2$, ... $r_n$;

xi) decrease the number of lessons in requirement r by n;

xii) increase the number of lessons in requirement r by n;

xiii) remove a unit of item i from requirement r;

xiv) insert a unit of item i in requirement r;

xv) perform a retreat removing assignments back to assignment n;

xvi) terminate the run and output the current state of the arrays.

This program could then be used in conjunction with the Main Program as indicated in the flow diagram in figure 14.4. This shows that the Adjustment Routine would be called by the Main Program at the end of the initial chain to make the sort of adjustments discussed in 12.1 to remove the impossible situations detected in the initial chain and it would also be called during the main construction process to remove any impossible situations which the Main Program had failed to remove after a single retreat.

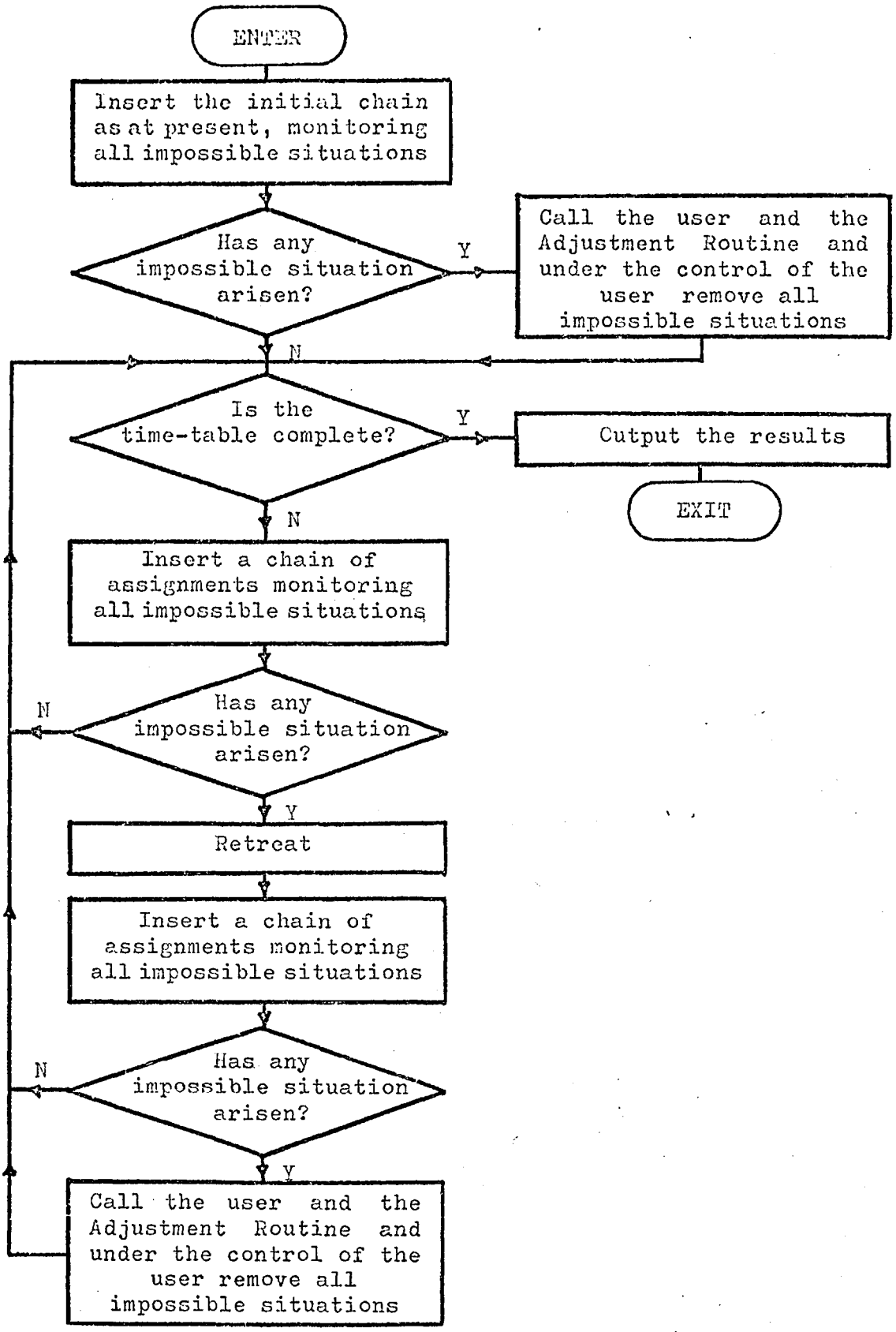
Some of the advantages of such a system are:-

i) no extra data has to be supplied to the computer to enable it to vary the constraints automatically itself since the necessary information is held by the user who could be a member of the school staff;

ii) relaxation of the constraints would not be restricted to acceptance of non-normal distributions, the splitting of multiple period lessons and the complete omission of lessons but would be completely flexible;

iii) impossible situations would be dealt with as they arose rather than just leaving out troublesome lessons and then

Figure 14.4


ENTER
Insert the initial chain as at present, monitoring all impossible situations
Has any impossible situation arisen?
Y
Call the user and the Adjustment Routine and under the control of the user remove all impossible situations
N
Is the time-table complete?
Y
Cutput the results
EXIT
N
Insert a chain of assignments monitoring all impossible situations
Has any impossible situation arisen?
N
Y
Retreat
Insert a chain of assignments monitoring all impossible situations
Has any impossible situation arisen?
N
Y
Call the user and the Adjustment Routine and under the control of the user remove all impossible situations

trying to fit all of them into an almost complete time-table;

iv) the total time taken for the construction of the time-table would be considerably reduced as most time-tables could probably be completed in one or two runs;

v) extensions of the system to allow data preparation, data checking and data correction via the terminal could make it possible to offer a complete on-line time-table construction and modification service to any school with access to a terminal and a suitable data link to the computer.

Some disadvantages are:-

i) the user would need considerable experience to be able to use the system effectively and without long periods of 'thinking time' at the terminal;

ii) the cost of the use of a terminal and possibly a data link to a remote computer could counteract the advantage gained by decreasing the number of runs needed to complete each time-table.

However, the Author is convinced that such an approach would enable not only more complete time-tables to be produced but that it would also allow many of the aesthetic quatilies which each school expects in its time-table to be included. Even if the current program were able to produce a complete solution of the type implied in our model for every set of data in a single run, few schools would be completely satisfied with the results. And certainly if some members of the staff were satisfied there would be others unsatisfied. By suggesting that the powers of the computer to perform clearly defined operations and the powers of the human to make decisions without the need for clearly defined rules should be combined we are seeking to make the best use of all the available resources and we are also making sure that whatever faults there may be in the finished product can only be blamed on the user and not on the machine.

## 14.7 A final comment

Since the advent of the computer school teachers have been wondering why they are capable of producing school time-tables year after year, admittedly with some considerable effort, while

the mighty machine has such difficulty. In this thesis the Author has described his efforts to prove that the machine can make time-tables with aid from the school staff only in the data preparation stage but regrettably these efforts have not been rewarded with total success. It can only be hoped that, if it has not already happened, someone will soon be able to offer to the schools of Great Britain a computer service capable of making good time-tables. Although the teachers may at present be superior their superiority is a crown which most of them would willingly relinquish.

***

# APPENDIX

This Appendix contains the data and a solution for the 1966-67 time-table for the small Canadian school discussed in 12.3. The numbering of the classes, teachers and rooms and the order of the requirements is exactly as in the data sent by Dr. J.Lions.

## The Item List

34

| | | | | | |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 18 | 1 | 2 |
| 2 | 1 | 1 | 19 | 1 | 2 |
| 3 | 1 | 1 | 20 | 1 | 2 |
| 4 | 1 | 1 | 21 | 1 | 2 |
| 5 | 1 | 1 | 22 | 1 | 2 |
| 6 | 1 | 1 | 24 | 1 | 2 |
| 7 | 1 | 1 | 26 | 1 | 2 |
| 8 | 1 | 1 | 27 | 1 | 2 |
| 9 | 1 | 1 | 29 | 1 | 2 |
| 10 | 1 | 1 | 33 | 1 | 2 |
| 11 | 1 | 1 | 34 | 1 | 2 |
| 12 | 1 | 1 | 35 | 1 | 2 |
| 13 | 1 | 1 | 39 | 1 | 3 |
| 14 | 1 | 1 | 42 | 1 | 3 |
| 15 | 1 | 1 | 51 | 1 | 3 |
| 16 | 1 | 2 | 53 | 1 | 3 |
| 17 | 1 | 2 | 54 | 1 | 3 |

9999

The Time Structure List

```
5
1    8
2    8
3    8
4    8
5    8

11   12   13   14   16   17   18   19
21   22   23   24   26   27   28   29
31   32   33   34   36   37   38   39
41   42   43   44   46   47   48   49
51   52   53   54   56   57   58   59

3    0

1    1    1    1    1    1    1    1
1    1    1    1    1    1    1    1
1    1    1    1    1    1    1    1
1    1    1    1    1    1    1    1
1    1    1    1    1    1    1    1

1    0    1    0    1    0    1    0
1    0    1    0    1    0    1    0
1    0    1    0    1    0    1    0
1    0    1    0    1    0    1    0
1    0    1    0    1    0    1    0

0    0    0    0    1    1    0    0
0    0    0    0    1    1    0    0
0    0    0    0    1    1    0    0
0    0    0    0    1    1    0    0
0    0    0    0    1    1    0    0

9999
```

The Requirement List

131

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 5 | 26 | 6 | 0 | | | | |
| 2 | 0 | 1 | 5 | 33 | 7 | 0 | | | | |
| 3 | 0 | 1 | 5 | 17 | 8 | 0 | | | | |
| 4 | 0 | 1 | 5 | 20 | 9 | 0 | | | | |
| 5 | 0 | 1 | 5 | 16 | 10 | 0 | | | | |
| 6 | 0 | 1 | 5 | 18 | 11 | 0 | | | | |
| 7 | 0 | 1 | 5 | 24 | 12 | 0 | | | | |
| 8 | 0 | 1 | 5 | 21 | 13 | 0 | | | | |
| 9 | 0 | 1 | 5 | 34 | 14 | 0 | | | | |
| 10 | 0 | 1 | 5 | 19 | 15 | 0 | | | | |
| 11 | 0 | 1 | 5 | 26 | 6 | 0 | | | | |
| 12 | 0 | 1 | 5 | 29 | 7 | 0 | | | | |
| 13 | 0 | 1 | 5 | 35 | 8 | 0 | | | | |
| 14 | 0 | 1 | 5 | 20 | 9 | 0 | | | | |
| 15 | 0 | 1 | 5 | 16 | 10 | 0 | | | | |
| 16 | 0 | 2 | 5 | 26 | 6 | 0 | | | | |
| 17 | 0 | 2 | 5 | 33 | 7 | 0 | | | | |
| 18 | 0 | 2 | 5 | 20 | 9 | 0 | | | | |
| 19 | 0 | 2 | 4 | 17 | 8 | 0 | | | | |
| 20 | 0 | 2 | 1 | 17 | 6 | 0 | | | | |
| 21 | 0 | 2 | 1 | 17 | 7 | 0 | | | | |
| 22 | 0 | 2 | 1 | 17 | 8 | 0 | | | | |
| 23 | 0 | 2 | 1 | 17 | 9 | 0 | | | | |
| 24 | 0 | 2 | 1 | 17 | 10 | 0 | | | | |
| 25 | 0 | 2 | 1 | 17 | 11 | 0 | | | | |
| 26 | 0 | 2 | 1 | 17 | 12 | 0 | | | | |
| 27 | 0 | 2 | 1 | 17 | 13 | 0 | | | | |
| 28 | 0 | 2 | 1 | 17 | 14 | 0 | | | | |
| 29 | 0 | 2 | 1 | 17 | 15 | 0 | | | | |
| 30 | 0 | 1 | 1 | 34 | 27 | 6 | 7 | 39 | 40 | 0 |
| 31 | 0 | 1 | 1 | 26 | 22 | 6 | 7 | 38 | 40 | 0 |
| 32 | 0 | 1 | 1 | 34 | 35 | 6 | 7 | 39 | 40 | 0 |
| 33 | 0 | 1 | 1 | 26 | 29 | 6 | 7 | 38 | 40 | 0 |
| 34 | 0 | 1 | 1 | 34 | 27 | 8 | 12 | 39 | 40 | 0 |
| 35 | 0 | 1 | 1 | 26 | 22 | 8 | 12 | 38 | 40 | 0 |
| 36 | 0 | 1 | 1 | 34 | 35 | 8 | 12 | 39 | 40 | 0 |
| 37 | 0 | 1 | 1 | 26 | 29 | 8 | 12 | 38 | 40 | 0 |
| 38 | 0 | 1 | 1 | 34 | 27 | 9 | 10 | 39 | 40 | 0 |
| 39 | 0 | 1 | 1 | 26 | 22 | 9 | 10 | 38 | 40 | 0 |
| 40 | 0 | 1 | 1 | 26 | 29 | 9 | 10 | 38 | 40 | 0 |
| 41 | 0 | 1 | 1 | 34 | 27 | 11 | 13 | 39 | 40 | 0 |
| 42 | 0 | 1 | 1 | 26 | 22 | 11 | 13 | 38 | 40 | 0 |
| 43 | 0 | 1 | 1 | 34 | 35 | 11 | 13 | 39 | 40 | 0 |
| 44 | 0 | 1 | 1 | 26 | 29 | 11 | 13 | 38 | 40 | 0 |
| 45 | 0 | 1 | 1 | 34 | 27 | 14 | 15 | 39 | 40 | 0 |
| 46 | 0 | 1 | 1 | 26 | 22 | 14 | 15 | 38 | 40 | 0 |
| 47 | 0 | 1 | 1 | 34 | 35 | 14 | 15 | 39 | 40 | 0 |
| 48 | 0 | 1 | 1 | 26 | 29 | 14 | 15 | 38 | 40 | 0 |
| 49 | 0 | 1 | 1 | 34 | 35 | 9 | 10 | 39 | 40 | 0 |
| 50 | 0 | 3 | 1 | 22 | 27 | 11 | 36 | 37 | 0 | |

continued

The Requirement List continued

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 51 | 0 | 3 | 1 | 22 | 27 | 12 | 36 | 37 | 0 |
| 52 | 0 | 3 | 1 | 22 | 27 | 13 | 36 | 37 | 0 |
| 53 | 0 | 3 | 1 | 22 | 27 | 14 | 36 | 37 | 0 |
| 54 | 0 | 3 | 1 | 22 | 27 | 15 | 36 | 37 | 0 |
| 55 | 0 | 2 | 1 | 22 | 27 | 6 | 36 | 37 | 0 |
| 56 | 0 | 2 | 1 | 22 | 27 | 7 | 36 | 37 | 0 |
| 57 | 0 | 2 | 1 | 22 | 27 | 8 | 36 | 37 | 0 |
| 58 | 0 | 2 | 1 | 22 | 27 | 9 | 36 | 37 | 0 |
| 59 | 0 | 2 | 1 | 22 | 27 | 10 | 36 | 37 | 0 |
| 60 | 65 | 1 | 1 | 26 | 1 | 2 | 40 | 0 | |
| 61 | 66 | 1 | 1 | 26 | 2 | 3 | 40 | 0 | |
| 62 | 67 | 1 | 1 | 26 | 3 | 4 | 40 | 0 | |
| 63 | 68 | 1 | 1 | 26 | 4 | 5 | 40 | 0 | |
| 64 | 69 | 1 | 1 | 26 | 1 | 5 | 40 | 0 | |
| 65 | 60 | 1 | 1 | 34 | 1 | 2 | 40 | 0 | |
| 66 | 61 | 1 | 1 | 34 | 2 | 3 | 40 | 0 | |
| 67 | 62 | 1 | 1 | 34 | 3 | 4 | 40 | 0 | |
| 68 | 63 | 1 | 1 | 34 | 4 | 5 | 40 | 0 | |
| 69 | 64 | 1 | 1 | 34 | 1 | 5 | 40 | 0 | |
| 70 | 0 | 2 | 1 | 17 | 8 | 0 | | | |
| 72 | 0 | 2 | 5 | 16 | 10 | 0 | | | |
| 73 | 0 | 2 | 5 | 18 | 11 | 0 | | | |
| 74 | 0 | 2 | 5 | 24 | 12 | 0 | | | |
| 75 | 0 | 2 | 5 | 21 | 13 | 0 | | | |
| 76 | 0 | 2 | 5 | 34 | 14 | 0 | | | |
| 77 | 0 | 2 | 5 | 19 | 15 | 0 | | | |
| 78 | 0 | 1 | 3 | 20 | 6 | 0 | | | |
| 79 | 0 | 1 | 3 | 16 | 6 | 0 | | | |
| 80 | 0 | 1 | 3 | 24 | 6 | 0 | | | |
| 81 | 0 | 1 | 3 | 20 | 7 | 0 | | | |
| 82 | 0 | 1 | 3 | 16 | 7 | 0 | | | |
| 83 | 0 | 1 | 3 | 24 | 7 | 0 | | | |
| 84 | 0 | 1 | 3 | 20 | 8 | 0 | | | |
| 85 | 0 | 1 | 3 | 16 | 8 | 0 | | | |
| 86 | 0 | 1 | 3 | 24 | 8 | 0 | | | |
| 87 | 0 | 1 | 3 | 20 | 9 | 0 | | | |
| 88 | 0 | 1 | 3 | 16 | 9 | 0 | | | |
| 89 | 0 | 1 | 3 | 24 | 9 | 0 | | | |
| 90 | 0 | 1 | 3 | 20 | 10 | 0 | | | |
| 91 | 0 | 1 | 3 | 16 | 10 | 0 | | | |
| 92 | 0 | 1 | 3 | 24 | 10 | 0 | | | |
| 93 | 0 | 1 | 3 | 18 | 11 | 0 | | | |
| 94 | 0 | 1 | 3 | 21 | 11 | 0 | | | |
| 95 | 0 | 1 | 3 | 19 | 11 | 0 | | | |
| 96 | 0 | 1 | 3 | 18 | 12 | 0 | | | |
| 97 | 0 | 1 | 3 | 19 | 12 | 0 | | | |
| 98 | 0 | 1 | 3 | 18 | 13 | 0 | | | |
| 99 | 0 | 1 | 3 | 21 | 13 | 0 | | | |
| 100 | 0 | 1 | 3 | 19 | 13 | 0 | | | |

continued

The Requirement List continued

| | | | | | | |
|---|---|---|---|---|---|---|
| 101 | 0 | 1 | 3 | 18 | 14 | 0 |
| 102 | 0 | 1 | 3 | 21 | 14 | 0 |
| 103 | 0 | 1 | 3 | 19 | 14 | 0 |
| 104 | 0 | 1 | 3 | 18 | 15 | 0 |
| 105 | 0 | 1 | 3 | 21 | 15 | 0 |
| 106 | 0 | 1 | 3 | 19 | 15 | 0 |
| 107 | 0 | 1 | 3 | 21 | 12 | 0 |
| 108 | 0 | 1 | 2 | 33 | 6 | 0 |
| 109 | 0 | 1 | 2 | 33 | 7 | 0 |
| 110 | 0 | 1 | 2 | 33 | 8 | 0 |
| 111 | 0 | 1 | 2 | 33 | 9 | 0 |
| 112 | 0 | 1 | 2 | 33 | 10 | 0 |
| 113 | 0 | 1 | 2 | 33 | 11 | 0 |
| 114 | 0 | 1 | 2 | 33 | 12 | 0 |
| 115 | 0 | 1 | 2 | 33 | 13 | 0 |
| 116 | 0 | 1 | 2 | 33 | 14 | 0 |
| 117 | 0 | 1 | 2 | 33 | 15 | 0 |
| 118 | 0 | 1 | 1 | 29 | 6 | 0 |
| 119 | 0 | 1 | 1 | 29 | 7 | 0 |
| 120 | 0 | 1 | 1 | 29 | 8 | 0 |
| 121 | 0 | 1 | 1 | 29 | 9 | 0 |
| 122 | 0 | 1 | 1 | 29 | 10 | 0 |
| 123 | 0 | 1 | 1 | 29 | 11 | 0 |
| 124 | 0 | 1 | 1 | 29 | 12 | 0 |
| 125 | 0 | 1 | 1 | 29 | 13 | 0 |
| 126 | 0 | 1 | 1 | 29 | 14 | 0 |
| 127 | 0 | 1 | 1 | 29 | 15 | 0 |
| 128 | 0 | 1 | 4 | 18 | 11 | 0 |
| 129 | 0 | 1 | 4 | 24 | 12 | 0 |
| 130 | 0 | 1 | 4 | 21 | 13 | 0 |
| 131 | 0 | 1 | 4 | 34 | 14 | 0 |
| 132 | 0 | 1 | 4 | 19 | 15 | 0 |

9999

The Fixed Free Period List

| | | | | | | |
|---|---|---|---|---|---|---|
| 35 | 11 | 21 | 31 | 41 | 51 | |
| | 16 | 26 | 36 | 46 | 56 | |
| | 47 | 48 | | | | |
| | 19 | 29 | 39 | 49 | 59 | 0 |

9999

The Forbidden Assignment List

9999

The Pre-assignment List

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 11 | 1 | 21 | 1 | 31 | 1 | 41 | 1 | 51 |
| 2 | 11 | 2 | 21 | 2 | 31 | 2 | 41 | 2 | 51 |
| 3 | 11 | 3 | 21 | 3 | 31 | 3 | 41 | 3 | 51 |
| 4 | 11 | 4 | 21 | 4 | 31 | 4 | 41 | 4 | 51 |
| 5 | 11 | 5 | 21 | 5 | 31 | 5 | 41 | 5 | 51 |
| 6 | 11 | 6 | 21 | 6 | 31 | 6 | 41 | 6 | 51 |
| 7 | 11 | 7 | 21 | 7 | 31 | 7 | 41 | 7 | 51 |
| 8 | 11 | 8 | 21 | 8 | 31 | 8 | 41 | 8 | 51 |
| 9 | 11 | 9 | 21 | 9 | 31 | 9 | 41 | 9 | 51 |
| 10 | 11 | 10 | 21 | 10 | 31 | 10 | 41 | 10 | 51 |
| 30 | 32 | 31 | 36 | 34 | 52 | 35 | 56 | 38 | 12 |
| 39 | 16 | 41 | 42 | 42 | 46 | 45 | 22 | 46 | 26 |
| 50 | 17 | 51 | 47 | 52 | 27 | 53 | 37 | 54 | 57 |
| 55 | 13 | 56 | 23 | 57 | 33 | 58 | 43 | 59 | 53 |
| 70 | 58 | | | | | | | | |
| 9999 | | | | | | | | | |

Notes

1. Items 1 to 5 represent classes in the lower section of the school and only their Physical Education lessons have to be time-tabled.

2. The periods on day d are numbered d1 to d4 and d5 to d9 leaving out d5 because period 5 is a lunch period throughout the school.

3. Requirement 71 on Dr. Lions' data requests 17 free periods for teacher number 35, all of which are pre-assigned. This has therefore been expressed on the Fixed Free Period List rather than as a requirement.

4. All classes spend the first period of each day with their Class Teacher thus all first period lessons are pre-assigned.

The Solution

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 11 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 12 | 11 | 13 | 38 | 81 | 98 | 107 | 116 | 123 | 132 | | |
| 13 | 12 | 14 | 15 | 19 | 55 | 62 | 73 | 74 | 105 | 115 | 131 |
| 14 | 48 | 82 | 87 | 112 | 130 | | | | | | |
| 16 | 21 | 39 | 75 | 76 | 78 | 85 | 95 | 117 | 129 | | |
| 17 | 40 | 50 | 79 | 84 | 97 | 104 | | | | | |
| 18 | 16 | 17 | 18 | 27 | 36 | 72 | 77 | 101 | | | |
| 19 | 69 | 86 | 96 | 102 | | | | | | | |
| | | | | | | | | | | | |
| 21 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 22 | 11 | 13 | 14 | 45 | 82 | 92 | 94 | 100 | 124 | | |
| 23 | 18 | 20 | 37 | 56 | 72 | 75 | 103 | 104 | 113 | | |
| 24 | 47 | 86 | 95 | 96 | | | | | | | |
| 26 | 22 | 46 | 78 | 89 | 97 | 99 | 109 | 122 | 128 | | |
| 27 | 12 | 52 | 63 | 80 | 91 | 93 | 107 | 111 | 131 | 132 | |
| 28 | 15 | 16 | 17 | 19 | 73 | 74 | 76 | 77 | 121 | | |
| 29 | 88 | 90 | | | | | | | | | |
| | | | | | | | | | | | |
| 31 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 32 | 14 | 30 | 91 | 94 | 98 | 110 | 126 | 129 | 132 | | |
| 33 | 26 | 33 | 57 | 76 | 88 | 92 | 105 | 115 | 128 | | |
| 34 | 12 | 15 | 64 | 80 | 87 | 93 | 99 | 106 | | | |
| 36 | 29 | 31 | 73 | 85 | 89 | 90 | 97 | 130 | 131 | | |
| 37 | 11 | 49 | 53 | 83 | 84 | 114 | 125 | | | | |
| 38 | 13 | 16 | 17 | 18 | 25 | 65 | 72 | 74 | 75 | 77 | |
| 39 | 120 | | | | | | | | | | |
| | | | | | | | | | | | |
| 41 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 42 | 14 | 41 | 82 | 86 | 96 | 103 | 105 | 112 | 118 | | |
| 43 | 16 | 28 | 58 | 67 | 72 | 73 | 74 | 75 | 77 | 81 | 110 |
| 44 | 13 | 109 | | | | | | | | | |
| 46 | 12 | 23 | 42 | 85 | 90 | 102 | 104 | 108 | 129 | | |
| 47 | 11 | 51 | 66 | 83 | 84 | 91 | 94 | 100 | 101 | 117 | |
| 48 | 15 | 17 | 18 | 19 | 60 | 76 | 80 | 128 | 130 | 132 | |
| 49 | 44 | 79 | 92 | 106 | | | | | | | |
| | | | | | | | | | | | |
| 51 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 52 | 15 | 34 | 81 | 89 | 100 | 102 | 108 | 127 | 128 | | |
| 53 | 16 | 18 | 19 | 43 | 59 | 77 | 83 | 101 | 114 | | |
| 54 | 98 | 107 | 113 | 119 | 131 | | | | | | |
| 56 | 17 | 24 | 35 | 76 | 78 | 88 | 93 | 99 | 106 | | |
| 57 | 13 | 14 | 54 | 61 | 79 | 95 | 129 | 130 | | | |
| 58 | 32 | 70 | 72 | 73 | 74 | 75 | 103 | 111 | | | |
| 59 | 11 | 12 | 68 | 87 | 116 | | | | | | |

Note: This solution is expressed in its most compact form. The set of numbers associated with each period are the requirement numbers of those lessons which start at that period.

## BIBLIOGRAPHY


[1] Appleby,J.S., Blake,D.V. and Newman,E.A.
1961 - 'Techniques for Producing School Timetables on a Computer and their Application to other Scheduling Problems';
The Computer Journal, Vol. 3, p 237.

[2] Barraclough,Elizabeth D.
1965 - 'The application of a digital computer to the construction of timetables';
The Computer Journal, Vol. 8, p 136.

[3] Berghuis,J., van der Heiden,A.J. and Bakker,R.
1964 - 'The Preparation of School Timetables by Electronic Computer';
BIT, Vol. 4, p 106.

[4] Csima,J. and Gotlieb,C.C.
1964 - 'Tests on a Computer Method for Constructing School Timetables';
Comm. A.C.M., Vol. 7, p 160.

[5] Gotlieb,C.C.
1962 - 'The Construction of Class-Teacher Timetables';
Proceedings of the I.F.I.P. Congress, p 73,
North Holland Publishing Co., Amsterdam (1963).

[6] Johnston,H.C. and Wolfenden,K.
1968 - 'Computer Aided Construction of School Timetables'
I.F.I.P.Congress Submitted Papers, Booklet G, p 73.

[7] Lawrie,N.L.
1966 - 'Linear programming as an aid to school time-tabling';
Proc. British Joint Computer Conference, p 16.

[8] Lawrie,N.L.
1969 - 'An integer linear programming model of a school timetabling problem';
The Computer Journal, Vol. 12, p 307.

[9] Lions,J.
1966 - 'Matrix Reduction using the Hungarian Method for the Generation of School Timetables';
Comm. A.C.M., Vol. 9, p 349.

[10] Lions,J.
1967 - 'The Ontario School Scheduling Program';
The Computer Journal, Vol. 10, p 14.

[11] Lions,J.
1968 - 'A Generalisation of a method for the construction of class-teacher timetables';
I.F.I.P.Congress Submitted Papers, Booklet G, p68.

[12] Local Government Operational Research Unit
1967 - 'The Use of Computers for School Timetables';
Report Number C19.

[13] Local Government Operational Research Unit
1970 - 'The Birmingham University Program';
Interim Report (not published).

[14] Michalsen,H.
1968 - 'Automatisk Timeplanlegging';
published by Selskapet for Industriell og Teknisk Forskning ved Norges Tekniske Høgskole, Trondheim.

[15] The Ontario Institute for Studies in Education (Toronto)
1966 - 'The Ontario School Scheduling Program Mark II Data Preparation Manual'.

The publications listed above are those which have been referred to in the Thesis. A much fuller Bibliography, listing some 68 papers and articles on school timetabling, school scheduling (as used in the U.S.A.), university timetabling, examination scheduling, conference scheduling and associated topics is given in 'Time-Table Construction by a Computer' by Folkers,J.S. This paper was published as Report THLR68 by the Department of Mathematics, Delft University of Technology, Delft, Netherlands.