Thesis submitted for the degree of Doctor of Philosophy

"THE GENERATION BY COMPUTER OF TIMETABLES FOR SOUTH AUSTRALIAN SECONDARY SCHOOLS"

by

M. B. HEMMERLING, B.Sc.(Hons), Dip.T.(Sec).

Department of Computing Science,

University of Adelaide.

30th March, 1973.

ACKNOWLEDGEMENTS

I am greatly indebted to Dr. J. L. C. Macaskill, Department of Computing Science, University of Adelaide, for his helpful advice, sympathetic support and constant encouragement during the period of the research.

I thank Professor J. A. Ovenstone and Professor F. Hirst in their capacities of successive permanent heads of the computing department for their encouragement and support. Also Dr. I. N. Capon and his staff have been helpful and tolerant during the course of this research.

All practical data was supplied by the Education Department of South Australia to whom I am greatly indebted. In particular, I acknowledge with gratitude the assistance and advice given me by Mr. P. Robinson of Craigmore High School. To the school administrators involved in the numerous preliminary investigations and discussions associated with this work, I offer my thanks.

Miss H. Bovington undertook a difficult typing assignment and I am indebted to her for the presentation of this thesis.

I greatly appreciate the discussions and valuable comments of the following people :

Professor G. A. Rose ; Professor G. Bennett ; Dr. C. Bellamy ; Mr. I. Oliver ; Mr. I. Sefton ; Mr. D. Ryan ; Mr. J. Whittle

This thesis contains no material that has been accepted for the award of any other degree or diploma in any University, and to the best of my knowledge and belief contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

(M. B. HEMMERLING)

March, 1973.

TABLE OF CONTENTS

CHAPTER		PAGE
3	(Contd.)	

CHAPTER		PAGE

APPENDICES

APPENDIX A :

Publication entitled "A Computer Timetable Solution for the South Australian Secondary Schools" by Malcolm B. Hemmerling, presented at the 5th Australian Computer Conference, Brisbane, 1972.

APPENDIX B :

APPENDIX C :

APPENDIX C (Contd.) :

APPENDIX D :

SUMMARY

The research associated with this thesis has been undertaken with an objective to add to the understanding of the school timetable problem, and of producing a solution technique that is practical for the South Australian Secondary school timetable problems. In the introduction, the main investigations already published on this topic are reviewed, and the nature of the problem in general terms is presented.

Next, the theory necessary for the formulation of the mathematical model is summarised. Such disciplines as set theory, combinatorics, systems of distinct representatives and graph theory are included. The Marriage Problem, closely allied to the timetable problem is discussed.

Then the school timetable is theoretically formulated as a resource allocation problem with constraints, for defined activities. Practical features existing within schools are discussed. The problem is described in two related parts. First, the simple tight timetable problem is defined, so that a basis for comparison with other solution methods is established. Second, the practical problem is derived by expanding the constraints on the simple problem, so that the method will be of practical benefit to education administrators.

The theoretical formulation is preceded by a detailed discussion of the South Australian secondary school timetable problem. Each aspect of the practical problem is defined, and provision for its inclusion is made in the solution method. The aims of existing manual techniques are present-

ed and the shortcomings of manual systems for solving timetables discussed.

The algorithms for this study are then presented. Extensive use of binary operations, set theory and combinatorics, combined with the daily activity requirements of a school and computer application form the basis for this solution technique. Considerable attention is given to the economy in use of data storage and working arrays within the computer model. Methods for the inclusion of special features in timetables such as teacher-class sets involving several school resources for a specific time-period are included in the algorithms, and have important assignment implications. It is concluded that a proper understanding of these and other implications in the timetable problem is needed if the computer techniques are to be applied effectively.

Then the computer program is discussed. Core storage problems and various techniques to increase efficiency are presented. The effect of special features required in practical problems are investigated. The input in the form of daily activities for the timetable problem are discussed.

The mathematical model is general enough to be applicable to a variety of school timetable problems. The benefits of allocating time-periods to the daily class-activity requirements are discussed. It is noted that the technique used in this study can be expanded to other larger weekly problems at the expense of more computer time. A direct application of the solution method to the Craigmore High School timetable problem is presented, to demonstrate the practical nature of this system to an existing timetable problem. The solution produced is presently in use at that school and the program will be used for other schools in the future.

The benefits of an automated solution method are examined, and several conclusions that had important implications for the school system were reached. The method added a considerable contribution to the topics of school resource allocation problems, staffing requirements and timetable error-detection techniques. The program demonstrated that significant economies in core storage can be obtained through the use of bit patterns within computer words, without any accompanying penalty in computational speed. The economies of generating a computer solution was a prime consideration.

The thesis concludes with a general discussion in which various conclusions are drawn. New and extended areas of research are identified.

CHAPTER 1

INTRODUCTION

1.1 HISTORICAL BACKGROUND

A school timetable is required to co-ordinate the complex daily activities of the school environment. The efficiency of the school organization is reflected in the quality of this schedule of activities. During early times, the construction of the school timetable was a relatively simple exercise, since one teacher taught all subjects and the number of subjects offered was few. Now, with the specialist subject teachers, and the wide variety of subjects offered to students, the generation of school timetables is becoming increasingly more difficult. Manual methods are time consuming, and the increased complexity of the schedules has directed investigations to the production of timetables by computer methods.

The situation is further compounded by the variety of school types now in existence, such as the High, Technical High and Area Schools of South Australia. Each type has its own requirements, and the structure of an acceptable solution differs for each school.

One method of solution involves the enumeration of all arrangements of activities, ignoring those that do not satisfy the conditions of the timetable problem. e.g. no resource shall be allocated to more than one activity during any one time-period. From this set of solutions the 'best' solution is chosen. However this method fails through the volume of computation necessary to produce the millions of arrangements. This was noted previously by Appleby et al. (2), and this method can be dismissed, even though high speed computers are available.

1.1.1 Manual Methods

As mentioned by Sefton (52), many early articles on school timetable methods were included in various teachers journals not readily available today. Probably a typical example of such publications is by Robinson (45). This paper described prepared forms and detailed a list of steps for the generation of a timetable solution. The technique, however, would need adaption for schools other than the Canadian school for which it was designed, and is therefore not generally applicable.

In 1961 Lewis (28) compiled a comprehensive manual for the hand generation of English Grammar school timetables. The timetables considered were complex in their structure and included subject options through the grouping of lessons into "sets" allocated to common time-periods in the timetable solution. The variation of the cycle length of the school week of 6 to 10 days was discussed. Once again the method was designed for hand techniques in solving English Grammar school problems and was not widely applicable. However the "sets" are related to the teacher-class sets used in this thesis and described in a paper attached as Appendix A.

More recent attempts to formalize the school timetable problem have been published by Lawrie (27) and Clague (12). The approach used by Lawrie described 'layouts' which are due to Lewis (28). The layouts are "expressions of the curricula of groups of pupils" and the approach uses larger units of departments and groups of students of the same year level. The formulation is a prelude to a computer solution method using linear programming techniques. e.g. Lawrie (27). However the method as described is not directly applicable to the South Australian situation because of part-time teachers and the present curricula organization. The paper by Clague assumes that there could be some agreement "for formulating a preliminary timetable which may then require minor adjustments to meet particular requirements". The paper then discusses a systematic approach to solving the timetable problem, and suggests that the method may lend itself to computer implementation. No further work appears to have been done using this approach.

Various techniques using mechanical aids such as magnetic boards for interchanging timetable entries have been observed, with the common faults of being time consuming, labourious and giving no guarantee of a solution.

1.1.2 Early Publications

A series of publications, quoting work on computer generated timetables began in the late 1950's and early 1960's. Of these, was a group of 3 papers by Bush, Caffrey, Oakford and Allen (8), Oakford (38), and Bush (7) on the Secondary Education Project at Stanford University. The problem is described in the first paper and possible approaches to the solution are considered by Oakford, who also indicated the aim to combine the allocation of students to classes with the timetable problem. The third paper of Bush reports success in the design of a computer program. Three publications (58), (49,), (50) from Stanford outline the approach of producing a preliminary master schedule by computer and incorporating changes, additions and corrections using other programs. Up-dating is done manually and the computer is used to modify and record the effects of these changes. No indication of the techniques employed is given.

Other short papers of Flanagan (16), Welton (57), Wulff (58), and Blackford (6) are also noted but none given any indication of the methods or techniques that are applied.

1.1.3 Human Imitation and Heuristic Procedure

Early attempts to imitate manual methods were developed in the early 1960's and examples may be found in publications by a number of authors, e.g. Appleby et al (2), Barraclough (3), Berghuis et al (5).

There are basically two approaches that are evident, as noted by Ryan (46), these being :-

1. The interchange of pairs of entries to "improve" a trial timetable.

2. Generated assignments are entered in an evolving timetable if feasible or rejected otherwise. When an assignment can not be made, the program retreats to a previous stage of production and restarts. Several sophisticated heuristic techniques were incorporated to formalise the abstract features of manual methods. However these early efforts usually failed to produce solutions acceptable to schools. The main reasons were :-

    (a) the computer could not view the problem as a whole, nor did it possess the experience or intuition of humans.

    (b) the initial requirements and constraints were presented to the computer as inflexible conditions, but manual methods allow for modification of requirements when difficulties arise.

(c) the recognition of infeasibility as a result of an assignment was not programmed, and thus it was difficult to determine the critical assignment causing infeasibility later. Look-ahead feature is included in the paper by Hemmerling (24) to overcome such infeasibilities.

Oliver (40) using heuristic techniques reported some success using a "stable method" to keep track of assignments and back-tracks, but solutions produced bear little resemblance to actual school timetables. However, a tree-search approach similar to Oliver's, was adopted in this thesis.

1.1.4 Theoretical Methods

The first mathematical formulation of the timetable problem was proposed by Gotlieb (18) who recognised the need for conditions indicating feasibility. The model used a result of set theory to derive these conditions, the Hall's conditions (21), which Gotlieb suggested as necessary and sufficient for feasibility. Extensions to this theory were developed by Cisma (9, 10), Duncan (14, 15), and Lion's (29, 31). Lions (30) also demonstrated that the Hall conditions were not sufficient for feasibility by a counter-example. The main difficulty with the method, was the inflexibility of the initial conditions that necessitated procedures for reruns with revised requirements.

The method has however been used successfully in Ontario, Canada and the implementation has been well documented by Lions (31 , 32 , 33) who also draws attention to the experimental nature of the method at present. The execution times at present are large due to the number of reruns needed for a solution.

1.1.5 Other Methods

Several other methods have been published, e.g. Mihoc and Balas (35) based on the theory of mathematical programming and Johnston et al. (26) with a two-dimensional allocation problem involving items and time-periods. However, methods have not been extensively tested and the application to real-school problems have not been established. The method of Johnston and Wolfenden was promising and the items, (resources of a school) were grouped together for lessons in the timetable solution. The approach used by Hemmerling (24) is similar, in that the activities of Hemmerling related to the items are of Johnson et al, and consists of groups of resources meeting together for a lesson.

The method of solution by Hemmerling has been applied to real-school situations and results have been presented in this thesis.

Other recent work has been published by De Warra (13), based on Swiss schools, but at this stage only theoretical results are available. Clacher (11) has generated timetables for real-schools using PERT, but work has been terminated and no further results published. Several reruns were also needed for this method of approach. Other more general publications such as (1, 25, 51, 53, 54) were also noted.

1.2 METHOD OF PRESENTATION

In the presentation of this thesis, practical features of the school timetable problem are examined. Computer techniques for the solution of real-school problems are developed and results are discussed. The subject matter of the thesis is presented as follows.

Chapter 2, contains the fundamental theory associated with the school timetable problem. Relevant aspects of set theory are introduced. This leads to the bijective mapping generator which plays an important part in the allocation of the activities of the timetable problem. This is followed by a discussion of graph theory that is used in the formulation of activity paths for required school activities. Then the theory of combinatorics, permutations and systems of distinct representatives is presented. Systems of distinct representatives are particularly relevant in the detection of infeasible situations in the solution method of this work.

Chapter 3 contains a discussion of the practical features of the South Australian Secondary School timetable problem. Various characteristics such as teacher-class sets, fixed time-periods, block-periods and school policies are discussed. The effects of limited resources in relation to the school timetable are noted.

A theoretical formulation of the school timetable problem is given in Chapter 4. The mathematical model is discussed together with the method of solution. The problem, as presented, is combinatorial, and bijective mappings for the allocation of activities are used in the solution method. Important aspects of the work are discussed in this chapter, and are related to the solution algorithms of chapters 5 and 6. The simple tight timetable problem is defined at this stage to give a basis for comparison with other solution methods and also for testing purposes. This is followed by a mathematical description of the practical problem.

In chapters 5 and 6 the algorithms for the computer program are presented. The composite availability vector is discussed in detail along with the implication algorithm. The importance of the two aspects in relation to the rejection of infeasible mappings is discussed. This leads to techniques for the reduction of the binary matrices containing all mappings for the timetable solution. The bijective mapping generator is formulated and the philosophy of the use of mappings in the solution method is discussed. Two important aids for the detection and correction of infeasible problems are presented. They are the resource load analysis and clash matrix. The use of these aids and the method of construction of the clash matrix is given. These are practical devices and are useful for both computer and manual solution methods for the school timetable problem.

Chapters 7 and 8 are concerned with the computer program and solutions of timetable problems. Methods of data presentation and conversion of the algorithms of chapters 5 and 6 into program form are discussed. The method of application of the computer program to timetable problems is presented and results are given. Test runs are described, results analysed and conclusions reached. Chapter 8 is primarily concerned with a description of the application of the computer program to a practical school problem, selected by the Education Department of South Australia (the Craigmore problem). A discussion of the difficulties of the real problem is given together with the solution presently in use at Craigmore High School.

Chapter 9 contains a discussion of the major conclusions of the research. Future research topics are outlined and suggestions are made for future extensions of the work of this thesis.

It is submitted in Chapter 9 that the findings of the research of the thesis have a direct application to school timetable problems and will be of considerable value to those involved in the preparation of school timetables in practice.

CHAPTER 2

MATHEMATICAL THEORY FOR THE SCHOOL TIMETABLE PROBLEM

2.1 INTRODUCTION

The initial sections of this chapter summarise the combinatorial theory required for the approach to the school timetable problem contained in this thesis. A short resume of set theory, combinatorics and graph theory has been compiled from the references of H. Ryser (47), M. Hall (19), J. Riordan (44), C. Liu (34) and F. Harary (23). Various theorems have been included in the text without proof, but reference has been given. The theory of distinct representatives has an important application in the solution method, and has been quoted from the authors L. Mirsky and H. Perfect (37), C. Berge (4) and D. Raghavao (43).

The final section of this chapter summarises graph theory, to be used in the formulation of the mathematical model of the timetable problem. This chapter is a theoretical review, in preparation for the mathematical procedures presented in chapters 4, 5 and 6.

2.2 SET THEORY

This section summarises the aspects of set theory, that have direct relevance to the work of this thesis. It will be shown in chapters 3 and 4 that the school timetable problem requires the allocation of resource vectors to time-periods (section 3.2, chapter

3). The various set operations used are quoted in this section.

The notation for a set is described as follows :-

$$T = \{1, 2, 3, \ldots\ldots\}$$

denotes a set T of elements labelled 1, 2, 3, ... A set may also be described by listing all elements of the set.

$t_1 \in T$ signifies that t_1 __is__ a member of the set T, and

$t_1 \notin T$ the contrary, that t_1 __is not__ a member of the set T.

If T is a __finite set__ of n elements, T is called an __n-set*__. When $n = 0$, T is the __null set__, denoted by :- $\emptyset$

An __r-subset__ of the n-set T is a collection of any r elements of T, $r \leq n$. When $r < n$, the r-subset is called a __proper r-subset__ of T.

It will be shown that the teacher resources and class resources, described in chapters 3 and 4, are proper subsets of the set of resources of a school.

If A(t) is some statement about the element $t \in T$, then the set T*, containing all elements of T for which A(t) is valid is denoted by,

$$T^* = \{t \; ; \; A(t), \; t \in T\}$$

Associated with each set T is a unique number denoted by $|T|$ called its __cardinality__ or the __cardinal number__.

If T is an n-set, then the number of elements in T is given by its cardinal number and is,

$$|T| = n$$

---

*The timetable problem will always be concerned with finite sets.

Let T_1 and T_2 be two sets (not necessarily finite).

$T_1 \cup T_2$ is defined to be the <u>union</u> of sets T_1 and T_2 containing all elements of both T_1 and T_2.

$T_1 \cap T_2$ is defined to be the <u>intersection</u> of sets T_1 and T_2 containing all elements common to both T_1 and T_2.

If $T_1 \cap T_2 = \phi$ then T_1 and T_2 are said to be <u>disjoint</u> (have no common elements).

In general,

$$\bigcup_{i=1}^{m} T_i \quad \text{and} \quad \bigcap_{i=1}^{m} T_i$$

will denote the union and intersection of the sets T_1, T_2, ... , T_m.

Let T be a set. Subsets T_1, T_2, T_3, ... , T_m of T, form a <u>partition of T</u> if,

$$T_i \neq \phi$$

$$i \neq j \text{ implies } T_i \cap T_j = \phi$$

$$T_1 \cup T_2 \cup \ldots\ldots\ldots \cup T_m = T$$

The T_i are called <u>classes</u> of the partition.

<u>EXAMPLE 2.1</u>

An example of a partition is given by the following

$$T = \{1, 2, 3, 4, 5, 6\}$$

Then

$$\{1, 2, 3\}, \{4\}, \{5, 6\}$$

is a partition of T with classes

$$T_1 = \{1, 2, 3\}$$

$$T_2 = \{4\}$$

$$T_3 = \{5, 6\}$$

The difference of sets T and S, denoted by T-S, is the set containing the elements of T that are not elements of S.

$$T-S = \{t \,;\, t \in T,\ t \notin S\}$$

Set theory is used extensively in the solution method described in chapters 5 and 6. The operations of union and intersection are applied to the availability arrays of chapters 3 and 4 to determine common time-periods available for a defined sub-set of resources. The solution method is based on the generation of feasible mappings, and a discussion of mappings is now given.

Let $T = \{1, 2, 3, \ldots, n\}$ and $S = \{s(1), s(2), \ldots, s(m)\}$ be two sets.

A mapping Δ_i of T into S denoted by

$$\Delta_i = \begin{pmatrix} 1 & 2 & 3 & & n \\ s_i(1) & s_i(2) & s_i(3) & \ldots & s_i(n) \end{pmatrix}$$

is a rule that associates an element $s_i(t_j) \in S$ with each element $t_j \in T$.

Each $s_i(t_j) = \Delta_i(t_j)$ for each $t_j \in T$ and is called the image of t_j under the mapping Δ_i

T is the domain of Δ_i and S is the range of Δ_i

For brevity, the mapping is sometimes written

$$\Delta_i : T \xrightarrow{\text{into}} S$$

A mapping Δ_i is surjective (a surjection) if, for every $s_i(t_j) \varepsilon S$, there exists at least one $t_j \varepsilon T$ such that

$$\Delta_i (t_j) = s_i(t_j)$$

(every element of S is an image for at least one $t_j \varepsilon T$.)

A mapping Δ_i is injective (an injection) if,

$$t_j \neq t_k \quad \text{implies} \quad \Delta_i (t_j) \neq \Delta_i (t_k)$$

for every $t_j, t_k \varepsilon T$.

(Distinct elements of T have a 1-1 correspondence with distinct images of S).

A mapping Δ_i is bijective (a bijection) if it is both surjective and injective, and is called a permutation of the images (every element of S is an image of one and only one element of T).

Note that mappings of T into S that have a maximum range of values in S as possible, are necessarily

injective if $|T| < |S|$

bijective if $|T| = |S|$

surjective if $|T| > |S|$

EXAMPLE 2.2

Let T = {1, 2, 3, 4} and S = {a, b, c}

An example of a surjective mapping is

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ a & b & a & c \end{pmatrix}$$

An example of an injective mapping for the sets T = {1, 2, 3} and S = {a, b, c, d} is

$$\Delta_2 = \begin{pmatrix} 1 & 2 & 3 \\ a & c & d \end{pmatrix}$$

An example of a bijective mapping for the sets T $=$ {1, 2, 3, 4}

and S $=$ {a, b, c, d} is

$$\Delta_3 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ c & a & b & d \end{pmatrix}$$

A <u>family of elements of T</u> indexed by I is denoted by

$$\Gamma = (t_i : i \varepsilon I)$$

where

$$I = \{1, 2, 3, \ldots, n\}$$

is the set of natural numbers, and Γ is an injective mapping, such that

$$\Gamma : I \xrightarrow{\text{into}} T$$

with

$$\Gamma(i) = t_i \quad , \quad t_i \varepsilon \ T, \quad i = 1, 2, 3, \ldots.$$

<u>EXAMPLE 2.3</u>

Let T $=$ {a, b, c, d}

Then (a, c, d) defined by

$$\Gamma = \begin{pmatrix} 1 & 2 & 3 \\ a & c & d \end{pmatrix}$$

is a family of T indexed by I $=$ {1, 2, 3}

Let Δ_1 and Δ_2 be two bijective mappings that map T onto itself. Then the mapping

$$\Delta_1(\Delta_2)$$

is given by

$$\Delta_1(\Delta_2(t_i))$$

for each $t_i \ \varepsilon \ T$

EXAMPLE 2.4

The set T is given by T = {1, 2, 3, 4} and the two mappings Δ_1, Δ_2 are defined as

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}, \quad \Delta_2 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 3 & 2 \end{pmatrix}$$

Then

$$\Delta_1(\Delta_2) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 2 & 3 \end{pmatrix}$$

The solution method described in this thesis generates feasible bijective mappings at each stage of the solution. This generation, of feasible bijections, will be discussed in section 5.3 of chapter 5, when the bijection generator is stated.

Before proceeding with the theory relevant to the enumeration of the solution space for the problem, a brief resume of graph theory is given. The mathematical model of chapter 4, section 4.2 formulates the timetable problem as a set of undirected acyclic graphs. The notation and definitions of this section are quoted from F. Harary (23).

2.3 GRAPH THEORY

A graph G consists of a finite non-empty set V of n-points together with a described set X of m unordered pairs of distinct points of V. Each pair x = {u,v} of points in X is a line of G and x is said to join u and v. A graph with n points and m lines is called an (n, m) graph. Under this present definition the graph G is an undirected graph.

EXAMPLE 2.5

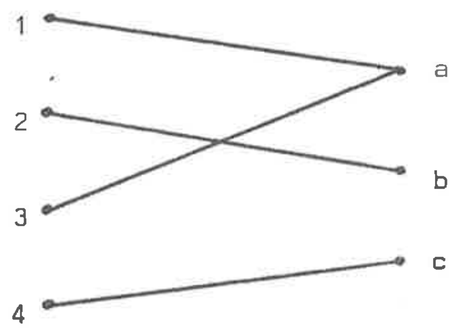
A surjective mapping, for example, on the two sets

$$T = \{1, 2, 3, 4\}, \quad S = \{a, b, c\}$$

defined by

$$\Delta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ a & b & a & c \end{pmatrix}$$

may be represented by the labelled graph



The point set V = {1, 2, 3, 4, a, b, c} and lines X = {1a, 3a, 2b, 4c}

A graph G is labelled when the n points are distinguished by names, (as in example 2.5).

A bigraph (or bipartite graph) G, is a graph whose point set V can be partitioned into two subsets V_1 and V_2, such that every line of G joins a point of V_1 with a point of V_2. This is the case in example 2.5 where $V_1 = T$, $V_2 = S$ and $V_1 \cup V_2 = V$.

If G contains every line joining V_1 and V_2 then G is a complete bigraph.

Note that by definition, a graph does not permit a line joining a point to itself (called a loop).

For the purpose of this thesis, much of the theory involves undirected graphs that do not have more than one line joining any two distinct points. When more than one line is allowable, these are called multiple lines and a graph that contains loops and multiple lines is called a pseudograph (Figure 2.1)

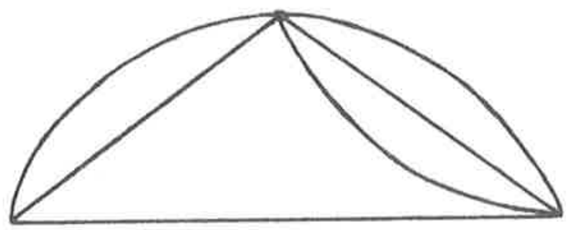
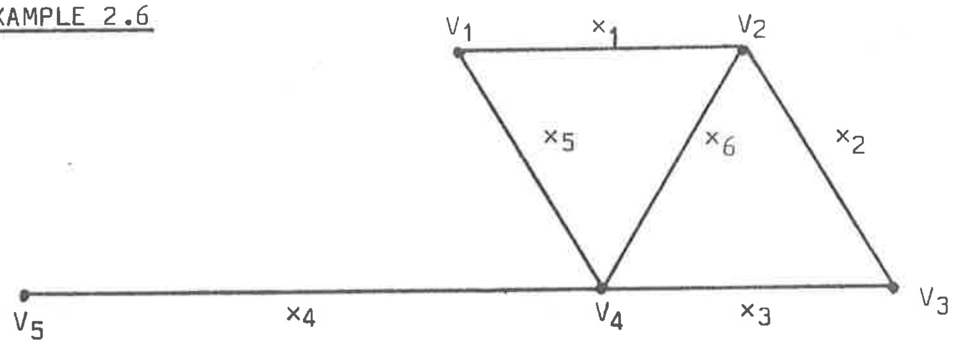


Figure 2.1 : A pseudograph

A <u>walk</u> of a graph G is an alternating sequency of points and lines v_0, x_1, v_1, x_2, v_2, ... , x_n, v_n, beginning and ending on points, and each line joins the points preceeding and succeeding it. If the walk is closed, $v_0 = v_n$, then it is a cycle, provided the n points are distinct and $n \geqslant 3$. (see example 2.6).

<u>EXAMPLE 2.6</u>



$v_1\ x_1\ v_2\ x_6\ v_4\ x_3\ v_3\ x_2\ v_2$ is a walk

$v_1\ x_1\ v_2\ x_2\ v_3\ x_3\ v_4\ x_5\ v_1$ is a cycle

A walk is closed if $v_o = v_n$ and is open otherwise. It is a trail if all lines are distinct and a path if all the points (and hence all the lines) are distinct.

From example 2.6, $v_5\ x_4\ v_4\ x_6\ v_2\ x_2\ v_3$ is a path.

A graph is connected when every pair of points are joined by a path.

A graph is acyclic[(1)] if it has no cycles.

A tree is a connected acyclic graph. Any graph without cycles will be called a forest and the components of a forest are trees.

The timetable problem is represented as a set of trees, that describe all activities for each course taught within the school. (Refer to chapter 4)

A directed graph (digraph) D consists of a finite non-empty set V of points, and a defined collection X of ordered pairs of distinct points. The elements of X are the arcs (directed lines) of D. (See Figure 2.2)

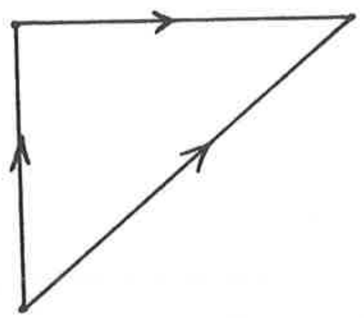


Figure 2.2 : A directed graph
(a digraph)

(1) The term acyclic is sometimes used to mean a graph that has no circuits. However, the above definition will be adopted within **this thesis.**

2.4 COMBINATORICS

Proofs of the following theorems have been omitted, but may be found in the references of C. Berge (4), and H. Ryser (47). Combinatorial theory is used for the enumeration of the size of the solution space for the timetable problem (section 4.3, chapter 4), and also in the determination of feasibility during the stages of the solution method (chapter 6).

A x B, the Cartesian Product of the sets A and B is the set of ordered pairs (a, b) where $a \in A$, $b \in B$.

$A^n = A \times A \times \ldots \times A$ is the set of n-tuple $(a_1, a_2, \ldots, a_n)$, $a_i \in A$ for $i = 1, 2, \ldots, n$.

2.4.1 THEOREM

The number of subsets $|P(T)|$ of the m-set $T = \{t_1, t_2, \ldots, t_m\}$ is

$|P(T)| = 2^m$ H. Ryser (47)

An ordered r-tuple $(t_1, t_2, \ldots, t_r)$ of not necessarily distinct elements of the n-set T is called an r-sample of T. $N(t_i)$ denotes the multiplicity of the element t in the r-sample.

2.4.2 THEOREM

The number of r-samples of an n-set is n^r.

H. Ryser (47)

This theorem is equivalent to the proposition that the number of mappings of an r-set T into an n-set A is n^r.

C. Berge (4)

An r-sample $(t_1, t_2, \ldots, t_r)$ of an n-set T, $1 \leqslant r \leqslant n$, such that $N(t_i) = 1$ for all $i = 1, 2, \ldots, r$ is called an r-permutation of n-elements.

An n-permutation is called a permutation, and is a bijective mapping of the n-set T onto itself. The graph associated with an r-permutation is a bigraph (section 2.3).

2.4.3 THEOREM

The number of r-samples without repetition of an n-set is P(n, r) where

$$P(n, r) = n(n-1)(n-2) \ldots (n-r+1)$$

H. Ryser (47)

This theorem is equivalent to determining the number of injections of an r-set T onto an n-set A.

n-factorial is written n! and represents

$$n! = \begin{cases} n(n-1)(n-2) \ldots 1 & \text{if } n > 0 \\ 1 & \text{if } n = 0 \end{cases}$$

and is the number of n-samples (permutations) of an n-set.

2.4.4 THEOREM

The number of permutations of n elements consisting of p elements of type 1

q elements of type 2

.

.

.

is given by

$$n! \,/\, (p!\ q!\ \ldots\ldots)$$

J. Riordan (44)

2.4.5 <u>THEOREM</u>

The number of bijections of an n-set X onto an m-set A, $m = n$ is $n!$

C. Berge (21)

2.4.6 <u>THEOREM</u>

The number of injections of the n-set X into the m-set A, $n < m$ is

$$n! \binom{n}{m}$$

where

$$\binom{n}{m} = m! \,/\, n!\,(m-n)!$$

C. Berge (21)

2.5 <u>PERMUTATIONS</u>

The generation of permutations is the basic feature for the solution method of this thesis. Each stage of the method requires the bijection generator (section 5.4, chapter 5) to produce a feasible mapping for the resource requirement. A review of permutation theory is therefore included.

A permutation of degree n is a bijection, written

$$\Delta = \begin{pmatrix} 1 & 2 & 3 & \cdots\cdots & n \\ k_1 & k_2 & k_3 & \cdots\cdots & k_n \end{pmatrix}$$

of the set T = {1, 2, 3, ... , n} onto itself.

Assuming T to be an ordered sequence of elements 1, 2, 3, ... , n, then to effect the permutation Δ on these elements is to replace each element i by $k_i = \Delta(i)$. The resulting n-tuple is called the re-arrangement of the sequence 1, 2, 3, ... n by the permutation Δ.

2.5.1 THEOREM

The permutation of degree n form a group S_n, called the symmetric group of degree n.

C. Berge (4)

From the theory of the preceeding section a directed pseudo-graph is described as a directed graph that may contain loops and multiple lines.

Each permutation Δ can be associated with a directed pseudograph, by representing the elements of T by the points labelled i = 1, 2, ... , n and by an oriented line directed by an arrow joining i to Δ(i) for each i.

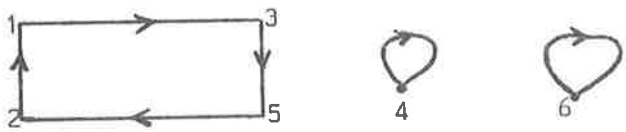
Since Δ is a bijection, there is only one incoming and one outgoing arc for each vertex i.

EXAMPLE 2.7

The permutation

$$\Delta = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 1 & 5 & 4 & 2 & 6 \end{pmatrix}$$

may be described by the directed pseudograph



Each component of the directed pseudograph is a cycle of the permutation, and these cycles partition the n-set of the permutation.

EXAMPLE 2.8

The components of example 2.7 are {1 3 5 2} , {4}, {6} and form a partition of the set {1, 2, 3, 4, 5, 6}.

If Δ has the first row in standard order 1, 2, 3, ... , n, then Δ may be denoted by

$$(k_1, k_2, \ldots, k_n)$$

where

$$k_i = \Delta(i) \quad \text{for} \quad i = 1, 2, \ldots, n.$$

Then the mapping Δ is characterized by the permutation $(k_1, k_2, \ldots, k_n)$.

If Δ contains cycles, then Δ may be completely defined by listing the cycles of the permutation (the components of its associated directed pseudograph).

EXAMPLE 2.9

Consider the permutation

$$\Delta = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 1 & 5 & 4 & 2 & 6 \end{pmatrix}$$

with components that give the partition

$$\{1, 2, 3, 5\}, \{4\}, \{6\}$$

and characterized by

(3 1 5 4 2 6)

may be completely described by listing its cycles

(1 3 5 2) (4) (6)

The length of a permutation, is the number of elements in its longest cycle.

A Right Cyclic Permutation (RCP) of length n is denoted by

$$\begin{pmatrix} 1 & 2 & 3 & \cdots & n \\ n & 1 & 2 & \cdots & n-1 \end{pmatrix}$$

and a Left Cyclic Permutation (LCP) of length n is denoted by

$$\begin{pmatrix} 1 & 2 & 3 & \cdots & n \\ 2 & 3 & 4 & \cdots & 1 \end{pmatrix}$$

A circular permutation consists of one and only one cycle of the permutation of length greater than one. The length of this cycle is the length of the permutation.

EXAMPLE 2.10

The permutation

$$\Delta = (1\ 3\ 5\ 2)\ (4)\ (6)$$

of the previous example is of length 4 and since this is a circular permutation, it may be more simply written as

$$\Delta = (1\ 3\ 5\ 2)$$

the single components being implied.

Let $\Delta_1 = (a_1, a_2, \ldots, a_r)$ and $\Delta_2 = (b_1, b_2, \ldots, b_r)$ be two permutations of an n-set T, $r < n$.

Then Δ_2 is incongruent to Δ_1 if, for every $i = 1, 2, 3, \ldots, r$

$$a_i \neq b_i$$ H. Ryser (47)

Further, when $\Delta_1 = (1, 2, 3, \ldots n)$ then Δ_2 is called a derangement of Δ_1, having no element in its natural position.

H. Ryser (47) shows that if

$$D_n = \text{number of derangements of } \Delta_1$$

$$D_o = 1$$

$$D_1 = 0$$

then

$$D_n = (n-1)(D_{n-2} + D_{n-1})$$

In a later chapter it is shown that the school timetable problem requires the generation of incongruents and derangements at each stage of the solution method. With teacher-class sets (defined in section 3.2 of chapter 3) coincidences are admitted and these will now be described.

Let $\Delta_1 = (a_1, a_2, \ldots, a_n)$ be a bijective mapping of $T = \{1, 2, \ldots, n\}$ onto itself.

Then Δ admits a coincidence at i if

$$a_i = \Delta(i) = i$$

2.5.2 THEOREM

The number of permutations admitting exactly p coincidences is

$$P^{p}_{(n)} = \binom{n}{p} P_{(n-p)}$$

where

$$P(1) = 0$$

$$P_{(n)} = n\,P(n-1) + (-)^{n}$$

C. Berge (4)

2.6 SYSTEMS OF DISTINCT REPRESENTATIVES

This section summarises the combinatorial theory associated with the theorem of P. Hall (21) on distinct representatives. Theorems are quoted from the references of L. Mirsky (36), M. Hall (20) and L. Mirsky and M. Perfect (37).

The theory of distinct representatives (or transversal theory) has been shown to be important in the timetable problem (see for example J. Cisma (9), C. C. Gotlieb and J. Cisma (10)). For the purpose of this thesis, the following theory will be used to determine the feasibility of unassigned requirements of the timetable problem, at each stage of the solution (see chapter 5). Feasibility will be demonstrated, through the determination of a system of distinct representatives for the CAV (section 5.2, chapter 5) of each class of the problem. The feasibility test will be discussed fully in later chapters of this thesis.

Consider the family $\rho = (C_i, i \varepsilon I)$ of the subsets of an n-set E indexed by I. Choose an element $t_i \varepsilon C_i$ for each $i \varepsilon I$. Denote the family of elements chosen by

$$\delta = (t_j ; j \varepsilon J)$$

Then the family of δ of elements of ρ is a system of representatives of ρ if there exists a bijection $\Delta : J \xrightarrow{\text{onto}} I$ such that

$$t_j \varepsilon C_{\Delta(j)}$$

for all $j \varepsilon J$.

If in addition $t_j \neq t_k$ for $j \neq k$, then δ is a system of distinct representatives (SDR) of ρ and t_j is said to represent $C_{\Delta(j)}$.

The range $\{t_j : j \varepsilon J\}$ of the SDR is a transversal of ρ.

A subset S of E is a transversal of $\rho = (C_i : i \varepsilon I)$ if there exists a bijection $\Delta : S \xrightarrow{\text{onto}} I$ such that

$$S \varepsilon C_{\Delta(s)} \text{ for all } s \varepsilon S$$

EXAMPLE 2.11

Consider the sets

$$C_1 = \{2, 3\}, \quad C_2 = \{1, 4\},$$
$$C_3 = \{3, 4, 5\}, \quad C_4 = \{1, 5\}$$

and

$$\rho = (C_1, C_2, C_3, C_4)$$

Then {2, 4, 3, 1} is a transversal of ρ and (2, 4, 3, 1) is an SDR.

If however

$$C_1 = \{2, 3\} \qquad C_2 = \{2, 3\}$$

$$C_3 = \{3\} \quad \text{and} \quad C_4 = \{1, 4\}$$

then no transversal nor SDR exists

The condition for the existence of an SDR is contained in the following theorem.

2.6.1 THEOREM

A necessary and sufficient condition for the existence of a system of distinct representatives for subsets C_1, C_2, ... , C_m is condition C :

for every integral $k = 1, 2, \ldots m$ and indices $i(1), i(2), \ldots, i(k)$ such that $1 \leqslant i(1) < i(2) < \ldots < i(k) \leqslant m$ the condition

$$\left| C_{i(1)} \cup C_{i(2)} \cup \ldots \cup C_{i(k)} \right| \geqslant k$$

holds.

This theorem is a direct result of a theorem by P. Hall (21). Due to the importance of this result for the feasibility tests within the solution method, the proof is included. It has been taken from D. Raghavarao (43) and forms an important basis for feasibility tests (Chapter 5).

PROOF :

The necessity of the theorem may easily be shown.

If $\left|C_{i(1)} \cup C_{i(2)} \cup \cdots \cup C_{i(k)}\right| < k$

then there do not exist k distinct elements in the k subsets.

Hence the contrary must be true.

Sufficiency is proven in two parts by induction on m.

Part a whenever $1 \leq k \leq m$ and $1 \leq i(1) < i(2) < \cdots$

$< i(k) \leq m$ then

$$\left|C_{i(1)} \quad C_{i(2)} \quad \cdots \quad C_{i(k)}\right| \geq k + 1$$

Part b for some $1 \leq k \leq m$ there are subsets $C_{i(1)}$,

$C_{i(2)}, \ldots, C_{i(k)}$ such that

$$\left| C_{i(1)} \cup C_{i(2)} \cup \cdots \cup C_{i(k)} \right| = k$$

The result is true for m = 1 and may therefore be assumed true for $n < m$.

Proof of part a

Choose an element $a_1 \in C_1$ and form the sets

$$C_i^* = C_i - \{a_1\} \qquad i = 2, 3, \ldots, m$$

By the assumptions, whenever $1 \leq k \leq m-1$ and

$2 \leq j(1) < j(2) < \cdots < j(k) \leq m$ then,

$$|C_{j(1)} \cup C_{j(2)} \cup \ldots \cup C_{j(k)}| \geq k$$

and by the induction hypothesis there exists an SDR $(b_2, b_3, \ldots, b_m)$ for the sets

$$C_2^*, C_3^*, \ldots, C_m^*$$

Then $(a_1, b_2, b_3, \ldots, b_m)$ is an SDR for $C_1, C_2, \ldots, C_m$ thus completing the proof of part a.

<u>Proof of part b</u>

Without loss of generality, assume that the subsets satisfying

$$|C_{i(1)} \cup C_{i(2)} \cup \ldots \cup C_{i(k)}| = k$$

are the first k subsets.

Then by the sufficiency condition and induction hypothesis, there exists an SDR $(a_1, a_2, \ldots, a_k)$ for the subsets $C_1, C_2, \ldots, C_k$.

Form the subsets

$$C_i^* = C_i - \{a_1, a_2, \ldots, a_k\}$$

for $i = k+1, k+2, \ldots, m$

Whenever $1 \leq h \leq m-k$ and $k < j(1) < j(2) \ldots$

$< j(k) \leq m$ then,

$$\left| C^*_{j(1)} \cup C^*_{j(2)} \cup \ldots \cup C^*_{j(k)} \right| \geq h$$

else $\left| C_1 \cup C_2 \cup \ldots \cup C_k \cup C_{j(1)} \cup C_{j(2)} \cup \ldots \cup C_{j(k)} \right|$

$< k + h$

thus contradicting the sufficiency condition. Hence, by the induction hypothesis, there exists an SDR (d_{k+1}, d_{k+2}, ... , d_m) for subsets

$$C^*_{k+1}, C^*_{k+2}, \ldots, C^*_m$$

It is easily verified that (a_1, a_2, ... , a_k, d_{k+1}, ... d_m) is an SDR for C_1, C_2, ... , C_m thus completing the proof of the theorem.

D. Raghavarao (43)

Call a set of r subsets $C_{i(1)}$, $C_{i(2)}$, ... , $C_{i(r)}$ a <u>block</u>, designated by Br,s where s is the number of distinct elements in the r subsets.

Condition C is equivalent to the requirement that $s \leq r$ for any block Br,s.

If r = s then call Bs,s a <u>critical block</u>. Bo,o is the void block and is critical. M. Hall (19) states the following two lemmas with regards blocks.

2.6.2 LEMMA

The union $B_{r,r} \cup B_{t,t}$ and intersection $B_{r,r} \cap B_{t,t}$ of critical blocks are again critical blocks, assuming condition C.

2.6.3 LEMMA

If $B_{k,k}$ is a critical block, then the deletion of elements of $B_{k,k}$ from the sets not belonging to $B_{k,k}$ leaves condition C valid.

The application of these two lemmas will be shown in the CAV Reduction algorithm and the Implication algorithm of chapters 5 and 6. At some stage of the solution method situations may arise where the CAV form a critical block and the image positions will be deleted from the remaining CAV.

2.7 STRICT SYSTEMS OF DISTINCT REPRESENTATIVES

Let $\rho = (C_1, C_2, \ldots, C_m)$ be a family of subsets of E. A family of elements $\delta = (a_1, a_2, \ldots, a_m)$ is a system of representatives for ρ if for some permutation Δ of $\{1, 2, 3, \ldots, m\}$ then

$$a_1 \in C_{\Delta(1)},\ a_2 \in C_{\Delta(2)},\ \ldots,\ a_m \in C_{\Delta(m)}$$

Call δ a strict system of distinct representatives (SSDR) of ρ if the a_i are distinct and

$$a_1 \in C_1,\ a_2 \in C_2,\ \dots,\ a_m \in C_m$$

Two systems $\delta_1 = (a_1, a_2, \dots, a_m)$ and $\delta_2 = (b_1, b_2, \dots, b_m)$ are said to be different if

$$a_i \neq b_i \qquad \text{for all } i$$

2.7.1 THEOREM

Let $\rho = (C_1, C_2, \dots, C_m)$ be a family of subsets that satisfy condition C.

If $\min(|C_1|, |C_2|, \dots, |C_n|) = r$ then,

$$R_N(\rho) \geq \begin{cases} r! & \text{if } r \leq n \\ r!/(r-n)! & \text{if } r \geq n \end{cases}$$

where $R_N(\rho) = R_N(C_1, C_2, \dots, C_n)$ denotes the number of strict systems of distinct representatives.

THEOREM EXTENSION

Assuming $|C_1| \leq |C_2| \leq \dots \leq |C_n|$, then

$$R_N(\rho) \geq \prod_{k=1}^{n} \max(1, |C_k| - k + 1)$$

C. Berge (4)

2.8 THE MARRIAGE PROBLEM

The theory quoted in the previous sections of this chapter may be applied to a variety of assignment and scheduling problems. One closely allied to the timetable problem is the 19th Century Marriage Problem. The problem may be stated in the following form.

There exists a set of men M and a set of women W. Each member of M is associated with some subset of the set W. Each member of M desires to marry a fixed number (not necessarily the same number for each man) of wives. From each mans' acquaintance subsets of W, find wives for each member of M.

It will be shown that the subsets of acquaintances are similar to the requirement resource vectors (section 3.2, chapter 3) of the timetable problem. The following theorem and conditions resemble those quoted later in this thesis for the timetable problem.

Halmos and Vaughan (22) generalised Hall's theorem to give necessary and sufficient conditions for the Marriage Problem solution. From J. Cisma (9) the following theorem is quoted.

2.8.1 THEOREM

Let $C = C_1, C_2, \ldots, C_m)$ be a finite family of subsets of W, and let $r_1, r_2, \ldots, r_m$ be non-negative integers called requirements. There exists a generalised system of distinct representatives in which each C_i is represented exactly r_i times if and only if the following condition holds :-

for each $k = 1, 2, \ldots, m$ and set of indices $i(1), i(2), \ldots, i(k)$ such that

$$1 \leq i(1) < i(2) < \ldots < i(k) \leq m$$

then

$$\left| C_{i(1)} \cup C_{i(2)} \cup \ldots \cup C_{i(k)} \right| \geq k$$

The necessity and sufficiency for the theorem is proven by Halmos and Vaughan (22). They also have shown that the theorem holds for infinite C's.

The C_i represent the acquaintance set for the ith man and r_i his desired number of wives. These will be related to the timetable problem later.

CHAPTER 3

STATEMENT OF THE SOUTH AUSTRALIAN SECONDARY SCHOOL TIMETABLE PROBLEM

3.1 INTRODUCTION

This chapter defines the terminology applied to the subsequent chapters of this thesis. The reasons for the increasing complexity of the South Australian Secondary School timetable problems have been summarised. The variety of school types (section 3.3), each with their unique requirements and timetable difficulties have been examined. Factors contributing to the need for an automated solution method have been quoted. The objectives of section 7.4 of chapter 7 require the production of a generalised solution method, capable of solving all types of secondary school timetable problems existing within South Australia.

Courses are defined in section 3.2. These are related to the limited resources available within the state. Limited resources are a major contributing factor to the timetable difficulties present in South Australia.

Manual timetable aims are discussed with a view to later formulation of the aims of the computer solution method. South Australian schools resemble other Australian schools, but, in general, differ markedly from schools outside Australia. For this reason, together with the expenses involved, scheduling systems already in existence elsewhere, are not readily adaptable to South Australian timetables. The cost factor, when calculated in dollars per student head at school was not acceptable.

On account of the increased difficulties associated with the manual methods, it was decided that an investigation into automated methods of solution should be carried out. It will be shown in chapter 9 that the solution method produced in this thesis, is useful for other aspects with the timetable problem. (e.g. staffing of schools.)

3.2 DEFINITIONS OF TERMINOLOGY

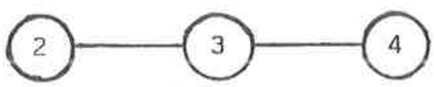
Some of the following definitions are extensions of earlier work by R. V. Oakford et al (39). Much of the terminology is related specifically to the South Australian education system, and is a refinement of an earlier publication attached as appendix A.

I. EVENT

An event is a moment in time. Events have no duration, and for the purpose of this thesis, define start and end points for time-periods.

EXAMPLE 3.1

Consider the labelled events 2, 3 and 4. They are represented in section 4.2 chapter 4 by the points of an undirected path of the following form :-



The lines (2, 3), (3, 4) represent activities during the time-periods described by the events, starting at events

2 and 3, and ending at events 3 and 4 respectively.

II. TIME-PERIOD

A time-period is the duration of a meeting involving some resources of the school. The duration of the time-period varies between the range of 30 to 50 minutes for timetables of the conventional type(1). A time-period is spanned by two events.

III. RESOURCES

A resource is an item involved in an activity at the school. The resources for the school timetable problem are teachers, classes, rooms, laboratories, workshops and special equipment.

A class resource, is a collection of students of the school, of the same academic level. There may exist several classes at the same academic level but all classes are disjoint from one another. Hence, a student may belong to at most one class, and classes will be considered as single items within this thesis.

In South Australian schools, classes are identified by alpha-numeric, or numeric codes that designate the academic level of the students within the class, and the class identifier to differentiate classes of the same level.

EXAMPLE 3.2

With South Australian secondary schools the academic levels are :-

---

(1)A conventional timetable consists of all time-periods of equal duration.

1st year, 2nd year, 3rd year, 4th year, 5th year.

A class 5C is a 5th year level class with a class name C.

The numeric convention is more usual with a code 501 signifying a 5th year level class, an 0-track where the track number designates the courses the students of the class pursue, and the 1st class. There are 5 tracks available within South Australian secondary schools. It will surfice to assume that a track number is related to courses in this thesis. More detail of tracks may be obtained from the reference "Our Secondary Schools" (59). The complete set of classes within a typical secondary school could be similar to :-

101, 102, 103, 111, 112, 121, 201, 202, 203, 211, 212, 301, 302, 311, 401, 402, 411, 501, 502.

IV. ACTIVITY

An activity is a meeting of resources available during a common time-period. The activity must involve two or more resources of the school, and requires the duration of one time-period for its completion. The minimum of two school resources is defined, since any activity must involve at least one class and one teacher resource. Other resources may also be included.

V. TIME-SPAN AND DAILY TIME-SPAN

School days are the days of the week when students attend school. The number of consecutive school days in a week, is called the weekly cycle length, and the days constitute a school week.

Each school day is divided into a number of time-periods of equal duration. The number of time-periods in a school day is called the daily time-span. The number of time-periods in a school week is called the time-span of the timetable.

For conventional secondary schools (using the conventional timetables), the daily time-span ranges from 6 to 9 time-periods. The weekly cycle length is invariably 5 days for all schools.

VI. BLOCK PERIOD

The number of consecutive time-periods required in a daily time-span, involving the same resources for a given activity, is called the block-period size. The duration of the consecutive activities is a block-period. All block-period sizes must be integer multiples of a single time-period for the conventional timetable. Start events that indicate the permitted start points for each block-period size for a daily time-span are usually laid down by the schools before the timetable is prepared. These are necessary, to prevent block-periods spanning lunch and recess breaks, thereby breaking the continuity of the block-period.

In South Australian secondary schools, block-period sizes range from 1 to 5 time-periods in length.

EXAMPLE 3.3

For clarity, label the events 0, 1, 2 ... , 8. Let recess breaks occur at events 3 and 7. (i.e. ... a recess break occurs between activities (2, 3) and (3, 4) and activities (6, 7), (7, 8).) Lunch breaks occur at event 5.

The daily activity pattern is described as follows :-

activities (0, 1), (1, 2), (2, 3), recess, activities (3, 4), (4, 5), lunch, activities (5, 6), (6, 7), recess, activities (7, 8).

Since block-periods may not span lunch or recess breaks, a block-period size 2 may occur as :-

(0, 1), (1, 2) ; (1, 2), (2, 3) ; (3, 4), (4, 5) ; (5, 6), (6, 7)

Block periods

(2, 3), (3, 4) ; (4, 5), (5, 6) ; (6, 7), (7, 8)

are not permitted because they span recess and lunch breaks at events 3, 5, 7.

In most schools the lesson patterns do not have 4 or 5 consecutive lessons without a recess or lunch break. Therefore on some occasions blocks must be broken. However in general they must not span lunch breaks and hence start periods for block-period sizes 4 and 5 are still defined in the usual form.

VII. TEACHER-CLASS SETS

A teacher-class set is a combination of more than one teacher and more than one class, required to be assigned to a common time-period of the timetable solution. The set may involve other resources, but the classes and teachers involved, define the teacher-class set.

South Australian schools endeavour to offer students a wide variety of subjects. However, the limited number of available teacher resources and facilities within the schools, do not allow classes to remain as a single learning body throughout the school day. A pseudo-class is a collection of students from each class of the teacher-class set, requiring instruction in a common subject area. Several classes may redivide into a set of pseudo-classes for some required activity. By using this technique, the school administrators found that they could offer a broad education, encompassing a variety of subject areas, within the physical limitations of the schooling system.

EXAMPLE 3.4

Consider the two 5th year level classes, 501, 502.

501 is oriented toward science disciplines

502 is oriented toward the humanities.

Each of the classes are offered electives but the school involved can not cater for two separate classes in all activities for all distinct time-periods.

Suppose the electives offered are Art, Film Study and General Affairs.

There are two methods of solution to the problem.

Let classes 501 and 502 each divide into 3 pseudo-classes containing students requesting the three electives.

EITHER ; the pseudo-classes remain as distinct units (6 in all) and are assigned as such during the same time-period

OR ; the corresponding pseudo-classes combine into 3 composite pseudo-classes for the common activity thus only involving 3 units.

The latter method is adopted with the use of teacher-class sets. The classes involved are defined by the resources of the activity and the set of pseudo-classes consist of students of these classes. The important advantage of this method is the resource saving and resource load reduction accomplished. (See later)

VIII. <u>REQUEST</u>

A list of desired resources for a given activity is called a <u>request</u>. When the request is included in a class requirement (defined later), the requested resources are called requirements. A required resource <u>must</u> be assigned to the relevant activity in the timetable solution.

Some of the resources contained in the request list for an activity do not necessarily appear in the resource requirements for that activity. During the manual solution procedure, the person involved with the timetable solution may decide that a particular resource request imposes severe restrictions on the problem. If it is decided that this request is not necessary then the involved resource is deleted from the activity. It would be undesirable to include such a procedure within the computer method of solution, and any such deletions occur during the manual data stage of this solution method.

EXAMPLE 3.5

Consider the activity involving the resources :-

A. Jones, 301, tape recorder, Room 3.

When assigned in the timetable solution, all of the resources would be dedicated to this activity. The tape recorder may be a heavily required item since the school has only one, and would therefore impose a restriction on the assignment procedure. If the recorder was not an essential item to the activity then it may be deleted. This is a vetting stage mentioned in chapters 7 and 8, to avoid too stringent requirements being given to the solution procedure.

IX. PART-TIME TEACHERS

A teacher that is not available for assignment, initially, for every time-period of a daily time-span is called a part-time teacher.

Circumstances existing within the South Australian education structure demands the employment of teachers with restricted availabilities for assignment. Situations such as part-time university courses, family commitments, and the sharing of teacher resources between schools are contained within this part-time structure.

The number of time-periods that are available for assignment, for a part-time teacher, is usually expressed as a fraction of the total daily time-span. It should be noted that other resources, beside teachers may have limited availabilities, (e.g. workshops, that are shared with other schools). In general, all other resources are fully available for all time-periods of a daily time-span.

EXAMPLE 3.6

A 6/8 part-time teacher is available for 6 time-periods of an 8 time-period daily time-span.

Part-time teachers with limited availabilities, impose heavy restrictions on the school timetable problem. The restrictions compound when the part-time teacher is utilized for all available time-periods. This special type of restriction is said to be a "tight condition" and in general will constitute a block (as defined in section 2.6, chapter 2). This will be discussed more fully in later chapters.

X. FIXED TIME-PERIODS

Some activities must occur during specified time-periods. The time-period for the activity involved cannot be changed, and is called a fixed time-period.

The activity may involve several resources. A fixed time-period may only involve one activity and hence can only have a block-period size of one.

EXAMPLE 3.7

(a) A teacher, A. Jones, and class 301 must meet for a science lesson during the activity (2, 3) every Monday (the 3rd lesson). A television set is required for the purpose of viewing a science program during this activity.

The fixed time-period is 3 associated with activity (2, 3) on a Monday and involves the resources :-

A. Jones, 301, T.V.

(b) All senior classes, 4th and 5th year levels, must meet during the activities (6, 7), (7, 8), related to the 7th and 8th time-periods on each Wednesday with teachers A. Jones, B. Brown, C. Smith, D. Black for inter-schools sports.

In this second example the fixed time-periods are the 7th and 8th on Wednesday with resources : A. Jones, B. Brown, C. Smith, D. Black, 401, 402, 403, 411, 501, 502, 503.

For clarity, each class is treated separately for fixed time-period requirements, when they are assigned by manual methods.

XI. COURSE

A course is a body of subject matter to be studied by classes of students. A course may encompass several academic levels, or may involve single classes only.

English, Mathematics I, Physics and Geography are examples of course names. English is a course offered to all levels within a school while Biology is only offered to 4th and 5th year level students.

The course name is the same for each of the academic levels, but the subject content differs at each level. The course structure and method of presentation of the subject matter may differ for each involved class.

XII. COURSE STRUCTURE

A course structure is the daily activity organization, for the purpose of instruction in the subject matter of the course. Students and school resources must be arranged into meetings, that satisfy the requirements of the course. Course structures for courses offered in secondary schools in South Australia, consist of one or several of the following phases :-

1. INSTRUCTIONAL PHASE

Four types of instructional phases are defined :-

Lecture phase - usually consist of lessons of block-period size one, and are activities involving a teacher in a lecturing situation with a class.

Workshop, Laboratory phases - consist of activities requiring more than one consecutive time-period. Block-period sizes of 3 or 4 are quite usual for this phase type.

Group Discussion phases - consist of block-period size one activities. These phases are similar to lecture phases except students take an active role in the activity.

Independent Study phases - consist of single time-period activities for the purpose of private study by the students of the class. This session usually requires the use of the school library.

2. A MEETING PATTERN FOR EACH COURSE

This pattern indicates the activities and the time-periods involved. Block-period sizes are specified if required, for each academic level and classes involved.

3. COURSE DEPENDENCIES

A course dependency is a relationship between the subjects of different courses. For example, the two courses of History and Geography are offered as an alternative choice to students. This relationship is stated in the course-dependency section of the course structure for both courses, History and Geography.

EXAMPLE 3.8

An example of a course structure for one day could be

Course Name : English

Instructional phases :

1st year level	1 lecture, 1 group discussion
2nd year level	2 lectures
3rd year level	1 lecture
4th and 5 year levels	none

Meeting Pattern :

one time-period, in a block-size one for each phase of the course.

Course Dependency :

the 3rd year course of ENGLISH is to be offered with options of French or Latin, to be assigned to a common time-period. (Will be a teacher-class set.)

XIII. CLASS-COURSE REQUEST

A class-course request details resources requested for an activity for the course (one time-period), for a particular class, during a daily time-span. If an activity requires more than one time-period, more than one class-course request is needed with some means of relating the two requests.

EXAMPLE 3.9

Consider the course ENGLISH described in the previous example. Assume that the 3rd year level classes involved are classes 301, 302, 303. The 3rd year level classes have the option of English or Latin or French, requiring one teacher for each course. No other resources are required.

The following class-course request details the resources requested for the English course :-

Resources : Jones, Smith, Brown
301, 302, 303.

where Jones teaches English
Smith teaches Latin
Brown teaches French

The class issuing the request is always one of the requested resources. The following details are extracted from examples 3.8, 3.9.

(a) Since more than one teacher and one class resource is involved, in the class-course request (and inter-course dependencies), the allocation is of a type described earlier as a teacher-class set.

(b) If the courses of French and Latin were to be taught separately (not as options to English), the 3 courses would require separate class-course requests.

XIV. CLASS REQUIREMENT

A class requirement is a complete collection of daily activities for the class. The collection describes all meetings, and resources required for each.

Associated with the class requirement is the block-period indicator, that defines relationships between the activities of a class requirement where the block-period sizes of two or more time-periods are required.

The resources requested in the class-course request are required when they are included in the class requirement. This transition from request to required has been discussed previously in this chapter.

The total requirement (stored in the form of a requirement matrix as discussed in chapter 4), is a collection of all class requirements for the school. It describes all activities for a daily time-span for all courses.

XV. RESOURCE LOAD

A resource load is the required number of time-periods necessary to satisfy the activities requiring the resource, for a single daily time-span. This number is expressed as a ratio (similar to the part-time teacher description) of the following form.

time-periods required : available time-periods for the resource

A class is always fully accepted and will have the maximum load. They must always be involved in some activity for each time-period of the daily time-span.

EXAMPLE 3.10

The teacher resource of A. Jones is required for 6 of 8 available time-periods. This is expressed as :-

A. Jones 6 : 8

Notice that all class resources have the ratio p : p where there are p time-periods in a daily time-span.

XVI. RESOURCE AVAILABILITY ARRAY

A resource availability array is a matrix representation of the availability of each resource in the school, for assignment to each time-period of a daily time-span. This array is a binary matrix where 1 indicates that a resource is available for assignment in the period concerned and 0 the contrary (see chapters 4 and 5).

Individual resource type availabilities will be required in the thesis. These will be referred by the following names :-

teacher availability array

class availability array

room availability array

equipment availability array

laboratory-workshops availability array

EXAMPLE 3.11

Consider the following resources :-

teachers : Jones, Smith, Brown

classes : 101, 102

rooms : R1, R2, R3

equipment : T.V.

The activity time-periods are labelled 1, 2, 3 for a 3 time-period daily time-span. The resource availability array is given by :-

Resource Name

Resource Period	Jones	Smith	Brown	101	102	R1	R2	R3	T.V.
1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	0
3	1	0	1	1	1	1	1	1	0

The array indicates :-

(a) teacher Smith is not available for time-period 3.

(b) the T.V. is not available for time-periods 2, 3.

(c) the remaining resources are available in each time-period.

The teacher availability array is :-

Teacher Resources

Period \ Teacher	Jones	Smith	Brown
1	1	1	1
2	1	1	1
3	1	0	1

In chapter 5, the availability array will be treated as a set of column vectors. Each column indicates the resource availability of the resource associated with the column. The availability arrays have an important role in the feasibility tests in the solution method.

3.3 THE SOUTH AUSTRALIAN SCHOOL TIMETABLE PROBLEM

Three school types are present in this state, namely High Schools, Technical High Schools and Area Schools. Each offer a variety of courses. They differ in size, physical structure, academic structure and number of available resources. The differences have a marked effect on the structure of their individual timetables, and for this reason are described below. The relevant similarities are compiled into a set of basic constraints for the computer method described in chapters 5 and 6.

A school timetable is a table indicating the activities for each time-period, such that no resource is assigned to more than one unrelated activity during any one time-period. The timetable must also satisfy the requirements of both internal school administrators and external Education Department policies. The relevant requirements existing in each school type are now discussed.

High Schools tend to be academically oriented. Courses require many lecture phases to be assigned within the timetable. Extensive use of teacher-class sets is prevalent, adding severe constraints on their timetable solutions.

Technical High Schools are oriented toward trade subjects with many activities involving practical work included. In contrast to High Schools, they have fewer lecture phases. Extensive use of block-periods and teacher-class sets are included for workshop and laboratory exercises. An added constraint involving the sharing of workshop resources with neighbouring schools is present in some schools.

Area Schools cater for both academic and trade courses. They are found in country areas and usually encompass first to fourth year courses. Class sizes are usually smaller than in High or Technical High Schools and resources are limited.

The current trend in education is toward more comprehensive schools. These schools will have timetables, that combine the features of High Schools, Technical High Schools and Area Schools. This trend, together with the stated objectives of chapter 7, required the design of a generalised timetable solution method, capable of solving timetables of all school types. At present, this new comprehensive school is being incorporated into High and Technical High Schools, by broadening the subject fields offered.

Table 3.1 gives a breakdown of schools into the 3 main secondary types for 1972.

High Schools	70
Technical High Schools	28
Area Schools	43
	141

TABLE 3.1

A breakdown of S.A. secondary schools into the 3 main school types.

A lack of sufficient resources, mainly teachers, is a major contributing factor to the timetable difficulties. The implications of these resource deficiencies are :-

(a) the need to utilize part-time teachers to off-set this short fall. This procedure imposes severe restrictions on the timetable structure, a direct result of the limited resource availabilities.

(b) the use of extensive teacher-class sets to cope with the diversity of subjects required by the students.

(c) the school facilities, including workshop and laboratory rooms are limited in some schools. An arrangement between schools, involving the common use of workshop facilities, in some circumstances, can overcome this deficiency of facilities. In this respect, the timetables of two neighbouring schools can be tied together.

EXAMPLE 3.12

Two neighbouring schools A and B require the use of workshop facilities. School A has the facilities on site, but school B does not. The two schools A and B must share the facilities in some manner., e.g. available to school A in the morning sessions and to school B in the afternoons.

In an effort to make clear the features found common to the above school types, the following list is included. These features have an important role, as they are used as a basis for the construction of the general secondary school timetable solution method described in chapters 4, 5 and 6. The features are :-

1. Teaching loads for each school day should be evenly distributed for each teacher. For example, a load of 7 : 8 on one day and 2 : 8 on another would not be desirable. A better distribution would be 5 : 8 and 4 : 8.

2. Course meetings should be evenly distributed throughout the weekly time-span. It would be undesirable to have 6 English lessons on one day and none for the remaining 4 days.

3. Single lecture phases for the same course should not, unless specifically required, be assigned to consecutive time-periods of a daily time-span.

4. Block-periods should not, in general, be assigned to time-periods that span lunch or recess breaks. In some circumstances this restriction may be omitted, e.g. block-period size 4 where recess breaks may occur between the two sets of 2 time-periods. Craft teachers do not oppose this break.

5. Each class of the school has an assigned <u>class teacher</u>. This teacher is responsible for the administrative functions related to the class, such as roll marking, handling notices and term reports. A class teacher usually takes the class for at least one activity per day, and are assigned to the first time-period for the class teacher functions.

6. Special requirements for fixed time-periods for radio and T.V. programs, inter-school sports, religious instruction lessons must be met.

7. Senior teachers, responsible for the course structures used within the school for each subject taught, should be available for at least one common time-period during a school week for the purpose of a senior staff meeting with the school principal.

8. Course structures must be arranged with the use of teacher-class sets to permit the wide variety of subject options for senior students in the 3rd, 4th and 5th year levels.

3.4 AIMS OF THE MANUAL TIMETABLE METHODS

Before proceeding with the formulation of the mathematical model of the timetable problem, the aims of the manual timetabling methods will be discussed. These are used in the formulation of objectives in chapter 7, for the computer method discussed in this thesis. They also indicate the method of approach for the solution procedure and this is noted in chapters 5 and 6.

Many manual methods are in present use within South Australian schools. These range from pencil-paper methods to sophisticated coloured magnetic systems. Various publications such as those of C. Lewis (28) and N. Lawrie (27) summarise the stages in these respective manual methods, in an effort to increase the efficiency and adaptability of these techniques.

The basic aims of all manual methods may be summarised as follows :-

1. To produce a workable solution, acceptable to the school concerned, within a determined limit of time.

2. To produce such a solution without much manual labour by the personal at the school.

3. To incorporate as many desirable features (described above), into the solution as possible. Some features are ignored if too much time is spent in the production of a solution.

4. To attempt to solve the problem on a daily basis, so that a timetable may be introduced into the school system as soon as possible. This aims at producing a timetable for Monday say, with the view that Mondays' timetable may be used while Tuesday, Wednesday, Thursday and Friday timetables are produced. A school could temporarily use the same day's timetable for several days, and in fact do, until the complete timetable is produced.

5. To attempt to assign the teacher resources and activities as described by the requirements of the school. Changes of resources may be necessary when too much time is spent satisfying some class-course request. The interchange of resources may solve the problem, by easing the conflicts that occurred during the solution method.

6. It is important that the senior level timetable is solved first. This section of the timetable is most difficult since it involves extensive teacher-class sets. The extent of these sets are defined by the variety of subjects offered to senior students, and the resources available to cope with the subjects. The timetable should be completed for the senior students with a minimum delay since they are involved with external examinations and heavily loaded courses with respect to the course content. The general approach by the manual methods is to solve the 5th year timetable first, on a daily basis, then work through the

4th, 3rd, 2nd and finally the 1st year timetables respectively. Many of the resources required in one level are required in another, and for this reason the timetable solution becomes progressively more difficult as the various levels are completed.

7. To efficiently use laboratory and workshop facilities is a secondary aim of the manual methods. For example, to have a 1st year laboratory class followed by a 5th year and then another 1st year class would be inefficient. The apparatus required by the classes would not be the same and it would be more convenient to have the two 1st year classes in consecutive activities. The manual timetable should attempt to group the levels together in an effort to increase the efficiency of these types of facilities.

The above aims are the more important ones, determined by consulting a variety of people involved with the manual production of school timetables. Other factors such as having principals free for specified time-periods have been considered and to a large extent have been included when possible. Any factor that is not important to the timetable solution is usually not considered during the construction stage, but may be included later, when possible.

CHAPTER 4

THE MATHEMATICAL MODEL FORMULATION FOR THE SCHOOL TIMETABLE PROBLEM

4.1 INTRODUCTION

This chapter contains the mathematical formulation of the school timetable problem. The model characterizes a resource allocation problem with constraints, and caters for all secondary school types within the state of South Australia. The simple tight timetable problem is discussed. This particular problem serves as a basis for the formulation of the generalised model described in the final section of this chapter, but has little practical significance. It does indicate however, a common problem for comparison purposes with other timetable methods. Therefore some results of chapter 7 have been quoted for the tight problems.

The model is produced from the theory of sets, combinatorics and graph theory, a review of which has been presented in chapter 2. The problem is first described in the form of a set of disjoint activity paths, that are transformed into a resource requirement array discussed in this chapter. The availability array for resources, as presented in chapter 3, section 3.2, is formulated together with the block-periods, fixed time-periods and teacher-class sets. Details of mappings for describing timetable features, and 0-1 matrices for the availability and resource requirements are given.

The chapter also shows in outline, how the model is used to solve the problem. The algorithms for the problem solution are described in detail in chapters 5 and 6.

4.2 THE MATHEMATICAL MODEL

The large weekly problem is formulated as 5 separate daily problems. This division of the timetable problem is not unique to this thesis, and has previously been used by C. C. Gotlieb (18). The 5 daily timetable problem has the following advantages.

First, the problem size is reduced into a set of 5 smaller sub-problems. Second, the requirement of an even spread of course and teacher loads may be incorporated into the problem solution more easily. It may be noted from the manual aims of section 3.4 of chapter 3, that this even distribution with respect to daily loads is mentioned as a desirable feature for the timetable solution.

The advantage of the weekly approach comes from the optimality of the overall solution. The daily problem approach achieves sub-optimal results, but optimality with respect to the weekly problem is not guaranteed. However, this is not a large disadvantage, as the sub-optimal solution is acceptable to the school administrators, and there is some doubt as to what the best solution contains.

Thus, although the breakdown is preferable for the algorithms of chapters 5 and 6, the sub-division may be excluded at the expense of :-

(a) a reduction in administrative control over the solution distribution of courses, and

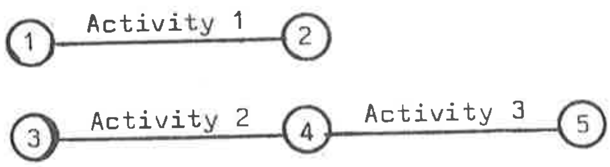
(b) an increase in computer time to produce a solution.

School administrators have emphasised the two following matters. First, the preference for a day by day structure, to give greater control over the course layout in the resultant timetable. Second, when problems do occur in the production of a solution, some partial solution may be available for use. This recovery stage is important to the administrators, since a partial solution could operate temporarily within a school, until such a time as the faults were rectified. A failure to obtain a solution to the weekly problem would exclude any possibility of a temporary solution.

Accordingly, the problem, irrespective of any computer time saving or efficiency, has been prepared on a daily basis to meet the specified recommendations of the departmental officers concerned.

It is convenient to represent the structure of the school timetable problem, by a set of disjoint undirected paths of nodes joined by lines. The nodes represent events and the lines, commonly called links, indicate the timetable activities. An event is the beginning or end of a timetable activity, and a timetable activity is the interaction of a given set of resources for a single time-period, i.e. ... a specific lesson. Example 4.1 below, is given to clarify this formulation. The sets of paths describe every activity within the school timetable.

EXAMPLE 4.1



In the graphs, the events represented by uniquely numbered nodes and the activities by lines (chapter 2, section 2.3). Some nodes are both start and end events, e.g. node 4. Others are either start events only (1 and 3) or end events (2 and 5).

The set of timetable paths are constructed from the course structures, class-course request and class requirements, of the school. The manner in which this is done is explained in detail in Example 4.2. Each course structure uses resources described by the school administrators, for each activity. For any activity of a path, the number of involved resources is at least two (one class and one teacher). Each activity has a duration of one time-period and will constitute a specific lesson for the resources involved.

As defined in chapter 3, section 3.2, a course structure describes activities for all academic levels, covered by the course, within the school. From these structures the set of paths is constructed in the following manner.

EXAMPLE 4.2

For the course named English the following structure is defined :-

Course Structure

3rd year level 1 lecture

2nd year level 1 lecture, 1 group discussion

1st year level no meetings

Class-Course Request

3rd year level

class 301 to meet teacher A. Jones

class 302 to meet teacher A. Jones

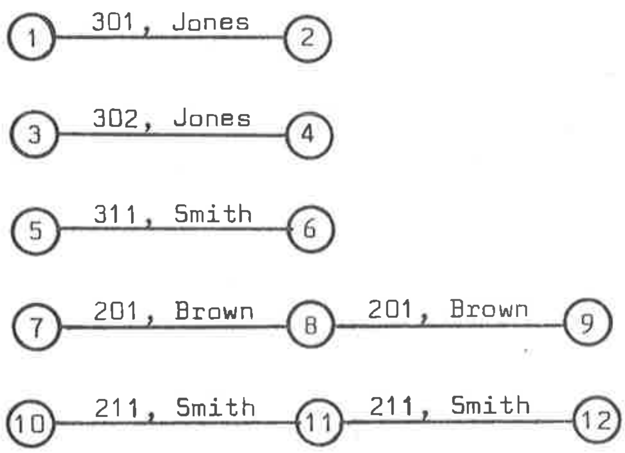
class 311 to meet teacher B. Smith

2nd year level

class 201 to meet teacher C. Brown

class 211 to meet teacher B. Smith

From the above description the following activity paths are constructed :-



The set of paths is composed of consecutive and parallel activities. The two terms, consecutive and parallel, relate to the resources involved in the activities concerned, and no indication of where the activities can be assigned in the timetable solution is implied by their use.

Consider two activities that involve the subsets X and Y of resources, requested from the resource set E of the school.

Then, the two activities are <u>consecutive</u> if the difference sets (section 2.2, chapter 2) X-Y and Y-X are both empty.

i.e. every resource $\beta \in X$ implies $\beta \in Y$, and $\beta \in Y$ implies $\beta \in X$.

The two activities are <u>parallel</u> if either or both of the difference sets X-Y and Y-X are non-empty.

i.e. there exists a resource $\beta \in X$ such that $\beta \notin Y$ or/and a resource $\gamma \in Y$ such that $\gamma \notin X$.

When all class-course structures are considered in this manner, a total daily resource path structure may be constructed. Each activity of the paths is a lesson that must be associated with a time-period in the timetable solution. However, this structure does not fully specify the timetable problem. It only indicates the lessons, resources and number of time-periods required. It does not specify resource availability, fixed time-period requirements, block-periods, nor the structure of teacher-class sets, described in section 3.2, chapter 3.

The resource availability array, as described in section 3.2 of chapter 3 will now be formulated.

Let E be an α-set E = $\{1, 2, 3, \ldots, \alpha\}$ of resources β where $\beta = 1, 2, 3, \ldots, \alpha$ and $\alpha \geq 2$. A school must have at least one teacher and one class resource, and hence the lower limit on α.

Let a daily time-span consist of p time-periods j, where $j = 1, 2, 3, \ldots, p$.

Then an $\alpha \times p$ array A, defined by :-

$$A = (a_{\beta j}) = \begin{cases} 1 & \text{if resource } \beta \text{ is available for} \\ 0 & \text{assignment to time-period } j \end{cases}$$

where $\beta = 1, 2, \ldots, \alpha$ and $j = 1, 2, \ldots, p$ is called the <u>resource availability array</u>.

The model defined above is now used to present an introductory formulation of the timetable problem as follows.

4.3 THE SIMPLE TIMETABLE PROBLEM

The simple timetable problem is defined in other publications on timetables, e.g. J. Lions (33). It has no block periods, teacher-class sets, or fixed time-periods and the resources are available for assignment to every time-period of the daily time-span. This problem, since it can be clearly defined, is used as a basis for comparison between various timetable procedures. It is presented at this stage to give firstly, a basis for the comparison of results with other solution procedures to the simple problem, and secondly, to formulate relevant conditions and constraints for the practical problem that follows.

In the following fomulation the subsets of resources, associated with the activities of the timetable problem are considered to be ordered samples of the school resource set E. The ordered samples are denoted by $\bar{r}_{i'}$ and are called resource vectors. $\bar{r}_{i'}$ defines the resources requested for the i'-th activity $\bar{a}_{i'}$ of the set of activity paths.

Denote the set of resource vectors by $\bar{R}$ and let $|\bar{R}| = n$ be the number of elements of $\bar{R}$ (see section 2.2, chapter 2). Then there are n activities in the timetable problem.

Let $|\bar{r}_{i'}|$ denote the number of resources in the i'-th resource vector, and since every activity must involve at least one teacher and one class, then $|\bar{r}_{i'}| \geqslant 2$. Further, each resource of $\bar{r}_{i'}$ is distinct since any resource may be requested at most once for any activity.

The notation of $\bar{R}$ and $\bar{r}_{i'}$ was chosen since the set of resource vectors will be reformulated into the resource requirement array R with resource vectors r_{ij} later in this section (see section 3.2, chapter 3). This transition, places the elements of $\bar{R}$ into the mathematical model, in preparation for the algorithms of the solution method. No direct relationship exists between the indices i', i and j but every member of $\bar{R}$ is an element in R.

Within the set E, there exists an m-subset C of classes. Let

$$C = \{ C_1, C_2, \ldots, C_m \}$$

denote the class subset, where $C_i \in C$ implies that $C_i \in E$ for all

$i = 1, 2, 3, \ldots, m$.

Since an activity involves at least one teacher and one class, then C is a proper subset of E, denoted

$$C \subset E, \quad \text{and} \quad |E| > m.$$

There also exists a proper k-subset of T of teacher resources. Let

$$T = \{t_1, t_2, \ldots, t_k\}$$

denote the set of teachers where $t_l \in T$ implies $t_l \in E$ for all $l = 1, 2, 3, \ldots, k$.

The two subsets C and T are disjoint

$$T \cap C = \phi, \quad \text{and} \quad |E| > m+k$$

For the simple timetable problem let E be the α-set such that

(a) $\alpha = m+k$

(b) $E = T \cup C$ (4.1)

i.e. E consists of only the class and teacher resources.

Let $R(\beta)$ denote the set of all activities $\bar{a}_i$, with resource vector $\bar{r}_i$, that involve the resource $\beta \in E$ within the set of paths.

Then $|R(\beta)|$ denotes the number of activities requiring the resource β.

i.e. the number of times β is required in the daily time-span.

Before continuing with the discussion of the general simple timetable problem, a special case of this problem will be described. This problem, called the <u>Simple Tight Timetable Problem</u> satisfies the following conditions :-

(a) each class and teacher must meet for some activity in the daily time-span,

(b) the number of teachers, classes and time-periods in a daily time-span are the same

$$m = k = p$$

(c) every resource is fully utilized in each time-period of the timetable (no slack).

The mathematical formulation follows :-

Since every resource is fully utilized in every time-period, then,

$$\left| R(\beta) \right| = p \quad \text{for every } \beta \in E \qquad (4.2)$$

By the definition of the simple timetable problem, all resources are available for every time-period, hence

$$\sum_{j=1}^{p} a_{\beta j} = p \quad \text{for every } \beta \in E \qquad (4.3)$$

The following necessary constraints for the timetable solution are stated.

<u>CONSTRAINT 1</u>

No resource $\beta \in E$ shall be assigned to more than one unrelated[1] lesson for any time-period of the timetable solution.

---

(1) A lesson is related to some given lesson when both must be assigned to a common time-period, and both are associated with an activity that involves more than one class and more than one teacher, i.e. a teacher-class set.

CONSTRAINT 2

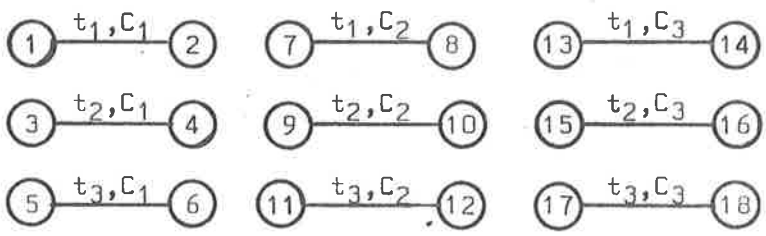
All activities of the timetable requirements must be assigned to a time-period in the timetable solution.

The following example 4.2, defines a simple tight timetable problem.

EXAMPLE 4.2

The problem involves 3 time-periods and 6 resources (3 classes, 3 teachers).

The set of activity paths for the problem are as follows :-



All simple tight timetable activity structures are of similar lay-out to the above example. All paths are parallel, and no consecutive activities exist.

i.e. the resource sample for each activity is unique within the simple tight timetable problem.

The activity paths are expressed in the form of an m x p class resource requirement array R in the following manner.

The class C_i, i = 1, 2, ... , m is always associated with row i of the class requirement array. The bijective mapping Δ', maps each of the classes C_i of the set C onto the integer row numbers i of the set I = {1, 2, 3, ... , m} of rows. Similarly, the set of time-periods j are mapped onto the column indices $J = \{1, 2, \ldots, p\}$, associating column j with time-period j by the bijection Δ''.

i.e. $\Delta'(C_i) = i$

$\Delta''(j) = j$ for $i = 1, 2, 3, \ldots, m$

$j = 1, 2, 3, \ldots, p$

We may assume, without loss of generality, that the activities $\bar{a}_i$, are ordered in the following manner.

Form the sequence :-

$$R(C_1),\ R(C_2),\ \ldots,\ R(C_m)$$

of activities with resources $C_1, C_2, \ldots, C_m$ respectively, where

$$R(C_i) = \bar{a}_{(i-1)p+1},\ \bar{a}_{(i-1)p+2},\ \ldots,\ \bar{a}_{ip}$$

This ordering may be performed, since for any class $C_i \in C$ $|R(C_i)| = p$.

For the row i, of the class requirement array, there are p resource vector elements, associated with the p activities involving class C_i in $R(C_i)$. The resource C_i may be omitted from the resource vectors of row i since it may be assumed that the class resource will always be involved.

EXAMPLE 4.3

Consider the activity paths of example 4.2. The following sets are constructed

$R(C_1) = \{(1, 2), (3, 4), (5, 6)\}$

$R(C_2) = \{(7, 8), (9, 10), (11, 12)\}$

$R(C_3) = \{(13, 14), (15, 16), (17, 18)\}$

From these ordered sets the following class resource requirements matrix is compiled.

$$R = \begin{bmatrix} (t_1,C_1) & (t_2,C_1) & (t_3,C_1) \\ (t_1,C_2) & (t_2,C_2) & (t_3,C_2) \\ (t_1,C_3) & (t_2,C_3) & (t_3,C_3) \end{bmatrix} = \begin{bmatrix} (t_1) & (t_2) & (t_3) \\ (t_1) & (t_2) & (t_3) \\ (t_1) & (t_2) & (t_3) \end{bmatrix}$$

In the latter array, the resources C_1, C_2, C_3 are omitted, since they may be assumed to be represented by the row number.

Each row of the array is a <u>class requirement</u> (see section 3.2 of chapter 3), listing all resources required in the p meetings of the class. The m class requirement rows are denoted by R_1, R_2, ... , R_m.

A solution array S, is an m x p array satisfying the following conditions :-

(a) each column j consists of m resource vectors, associated with the m class activities to be assigned to the time-period j designated by the column number.

(b) each row S_i of S contains the same resource vectors as each respective row R_i of R. i = 1, 2, ... , m.

(c) the order of the elements of row S_i is determined under a bijective mapping Δ_i (see below) associated with R (i.e.

... the order of the resource vectors in S_i is a permutation of the order of the vectors in R_i).

(d) S satisfies the defined constraints.

CONSTRAINT 3

Each element position of S must contain one and only one resource vector of R.

The solution method will be shown to consist of the generation of the bijective mappings Δ_i, i = 1, 2, ... , m to give the arrangements of the resource vectors in the solution rows S_i.

THEOREM 4.2.1

The simple tight timetable problem will always have a solution.

PROOF

By definition, each row R_i of R involves each teacher resource $t_1 \in T$ in a resource vector. We may assume without loss of generality that the resource vectors in column j or R all involve the teacher resource $t_j \in T$, e.g. see the form of the array R in example 4.3.

Consider the bijection Δ , that is the left cyclic permutation (LCP) defined in section 2.5 of chapter 2. The LCP is denoted by :-

$$\Delta = \begin{pmatrix} 1 & 2 & 3 & \cdots & p \\ 2 & 3 & 4 & \cdots & 1 \end{pmatrix}$$

and maps the element r_{ij} in position (i, j) of R_i, into the position (i, j+1) in S_i for j = 1, 2, 3, ... , p-1. The element r_{ip} is mapped into the position (i, 1) in S_i.

Using the theory of chapter 2, section 2.2, define the mappings

$$\Delta_{i+1} = \Delta(\Delta_i) \qquad i = 1, 2, \ldots, p-1$$

where

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 & \cdots & p \\ 1 & 2 & 3 & \cdots & p \end{pmatrix}$$

is the identity mapping.

i.e. in the case i = p = 3 the following mappings are produced.

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \Delta_2 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix},$$

$$\Delta_3 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

Then the mapping Δ_{i+1} maps the j-th element of row R_{i+1} into the $(i+1, \Delta_{i+1}(j))$ position in S_{i+1}.

By applying the mappings to their associated rows of R, the solution array is produced, that satisfied all stated conditions and constraints. Hence the simple tight problem has at least one solution.

Q.E.D.

The following example is included, to clarify the application of theorem 4.2.1 to the simple tight timetable problem.

EXAMPLE 4.4

Consider the resource requirement matrix of example 4.3, namely :-

$$R = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_1 & t_2 & t_3 \\ t_1 & t_2 & t_3 \end{bmatrix}$$

The mappings Δ_1, Δ_2 and Δ_3 are defined by :-

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

$$\Delta_2 = \Delta(\Delta_1) = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$$

$$\Delta_3 = \Delta(\Delta_2) = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

i.e. the LCP, Δ cycles the image elements of Δ_i to the left by one position, each time Δ is applied, to give Δ_{i+1}.

Then

$$\Delta_1(R_1) = (t_1 \quad t_2 \quad t_3) = S_1$$

$$\Delta_2(R_2) = (t_3 \quad t_1 \quad t_2) = S_2$$

$$\Delta_3(R_3) = (t_2 \quad t_3 \quad t_1) = S_3$$

e.g. The mapping Δ_3 for R_3 takes the elements of row R_3 of R in positions (3, 1), (3, 2), (3, 3) and places them into positions (3, 3), (3, 1), (3, 2) respectively.

Thus the solution

$$S = \begin{bmatrix} t_1 & t_2 & t_3 \\ t_3 & t_1 & t_2 \\ t_2 & t_3 & t_1 \end{bmatrix}$$

The simple tight timetable problem is associated with the problems of Latin Square, see H. Ryser (47). A Latin Square is a p x p array of p distinct elements such that :-

1. each row contains the p distinct elements
2. each column contains the p distinct elements
3. no row or column contains any element more than once.

The general simple timetable problem will now be discussed.

The general simple timetable problem does not require that $m = k = p$. However, every class must be utilized in every time-period of a daily time-span.

$$|R(C_i)| = p \qquad \text{for every } C_i \in C \qquad (4.4)$$
$$i = 1, 2, \ldots, m.$$

The following necessary conditions must be satisfied for the simple problem to have a solution.

(a) Any resource $\beta \in E$ can not be required in more than p unrelated activities.

$$|R(\beta)| \leq p \qquad \text{for every } \beta \in E \qquad (4.5)$$

This is a generalisation of equation 4.2 for the tight problem. The classes by definition satisfy the equation 4.5.

The total number of activities n for the simple problem is :-

$$n = m \times p$$

All activities are unrelated since no teacher-class sets exist within this problem type.

The contrary of equation 4.5 is assumed for some resource $\beta \varepsilon E$. Then the resource β is required in R for more than p time-periods. However, there are only p available time-periods for assignment within the timetable problem and no solution could be determined without violating the constraints.

i.e. all requirements must be satisfied, and no resource shall be assigned to more than one unrelated activity in a time-period.

Hence, the equation 4.5 must hold.

(b) Any resource $\beta \varepsilon E$, shall not be required in more unrelated activities, than the total number of time-periods available to that resource.

$$\left| R(\beta) \right| \leq \sum_{j=1}^{p} a_{\beta j} \qquad (4.6)$$

for all $\beta \varepsilon E$

by definition, this condition will always be satisfied for the simple problem. The proof follows that of equation 4.5.

(c) To restrict the problem to the hours within the timetable (i.e. ... no lessons can occur outside school hours) the following condition is included. Each resource $\beta \varepsilon E$ shall be available for at most p time-periods for a daily time-span.

$$\sum_{j=1}^{p} a_{\beta j} \leqslant p \qquad (4.7)$$

for every resource $\beta \varepsilon E$

(d) There must be at least as many teacher resources as there are classes.

$$|T| \geqslant |C| \qquad (4.8)$$

for any time-period with unrelated activities.

If the contrary were assumed, then a class would have no teacher. This violates the conditions that an activity must involve a teacher and class, and that a class must be associated with an activity for every time-period.

The above conditions are related to the computer algorithms in chapter 5. However, a mathematical formulation is given at this stage to demonstrate the association of the bijective mappings and systems of distinct representatives in the solution method. The simple problem will be used to demonstrate these connections.

Let Δ_i denote the non-empty set of all bijective mappings associated with row R_i of R for i = 1, 2, ... , m.

i.e. Δ_i defines the p! permutations of size p, of the elements 1, 2, ... , p.

The members of the set Γ_i are denoted by $\Delta_{i\delta}$, where $\delta = 1, 2, 3, \ldots, p!$

Denote the elements (resource vectors) of row R_i by r_{ij} where $j = 1, 2, \ldots, p$.

Two resource vectors $r_{i_1 j_1}$ and $r_{i_2 j_2}$ are <u>assignably equal</u> when there exists at least one resource $\beta \epsilon E$ such that $\beta \epsilon r_{i_1 j_1}$ and $\beta \epsilon r_{i_2 j_2}$.

i.e. the two resource vectors involve at least one common resource.

Consider two rows R_{i_1} and R_{i_2} of R, $i_1 \neq i_2$, with associated bijective mappings $\Delta_{i_1 \delta_1}$ and $\Delta_{i_2 \delta_2}$, that map the resource vectors of R_{i_1} and R_{i_2} respectively, into new positions within rows S_{i_1} and S_{i_2} of the solution array S.

Then the two bijective mappings $\Delta_{i_1 \delta_1}$ and $\Delta_{i_2 \delta_2}$ are <u>distinct</u> if for each $j = 1, 2, \ldots, p$ the two resource vectors in positions (i_1, j) and (i_2, j) of S are not assignably equal.

i.e. the two mappings do not map any resource into more than one lesson during anyone time-period.

<u>EXAMPLE 4.5</u>

Consider the bijection sets that consist of all mappings of 3 elements $j = 1, 2, 3$ (the time-periods that are equivalent to the column numbers in R and S)

Each set will have 3! = 6 elements (see chapter 2, section 2.5)

The sets Γ_1 and Γ_2 are defined by $\Gamma_1 = \{\Delta_{11}, \Delta_{12}, \ldots, \Delta_{16}\}$ and $\Gamma_2 = \{\Delta_{21}, \Delta_{22}, \ldots, \Delta_{26}\}$.

Each set Γ_1 and Γ_2 contains the 6 bijections

$\binom{1\ 2\ 3}{1\ 2\ 3}$, $\binom{1\ 2\ 3}{1\ 3\ 2}$, $\binom{1\ 2\ 3}{3\ 1\ 2}$, $\binom{1\ 2\ 3}{3\ 2\ 1}$, $\binom{1\ 2\ 3}{2\ 3\ 1}$, $\binom{1\ 2\ 3}{2\ 1\ 3}$

The order of the elements within Γ_2 and Γ_1 may be considered to be

$$\Delta_{11} = \Delta_{21} = \begin{pmatrix}1 & 2 & 3\\ 1 & 2 & 3\end{pmatrix}$$

$$\Delta_{12} = \Delta_{22} = \begin{pmatrix}1 & 2 & 3\\ 1 & 3 & 2\end{pmatrix}$$

$$\Delta_{13} = \Delta_{23} = \begin{pmatrix}1 & 2 & 3\\ 3 & 1 & 2\end{pmatrix}$$

$$\Delta_{14} = \Delta_{24} = \begin{pmatrix}1 & 2 & 3\\ 3 & 2 & 1\end{pmatrix}$$

$$\Delta_{15} = \Delta_{25} = \begin{pmatrix}1 & 2 & 3\\ 2 & 3 & 1\end{pmatrix}$$

$$\Delta_{16} = \Delta_{26} = \begin{pmatrix}1 & 2 & 3\\ 2 & 1 & 3\end{pmatrix}$$

It should be noted that for convenience the order of the elements has been chosen in the above manner, but that any other order could have been taken. An example of distinct mappings from Γ_1 and Γ_2 based on R_1 and R_2 of

$$R = \begin{bmatrix} t_1 & t_3 & t_5 \\ t_3 & t_7 & t_5 \end{bmatrix}$$

is

$$\Delta_{11} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

and

$$\Delta_{22} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}$$

since $\Delta_{11}(R_1)$ maps the elements in R_1 from positions (1, 1), (1, 2), (1, 3) to positions (1, 1), (1, 2), (1, 3) respectively, and, $\Delta_{22}(R_2)$ maps the elements in R_2 from positions (2, 1), (2, 2), (2, 3), to positions (2, 1), (2, 3), (2, 2) respectively.

Thus the resultant rows

$$S_1 = (t_1 \quad t_3 \quad t_5)$$

$$S_2 = (t_3 \quad t_5 \quad t_7)$$

do not have a resource occuring more than once in any column position j, j = 1, 2, 3.

The selection of distinct mappings associated with the rows R_1, R_2, ... , R_m of R, to produce the solution rows S_1, S_2, ... , S_m of S is equivalent to the selection of a set of distinct representatives from the sets Γ_1, Γ_2, ... , Γ_m.

The number of elements in each set Γ_i will be calculated through the number of available positions (time-periods) for each resource set of row R_i, i = 1, 2, ... , m. This calculation has important implications in the computer algorithms and is discussed fully in section 6.3, chapter 6.

A necessary and sufficient condition for the existence of a system of distinct bijective mappings for rows R_1, R_2, ..., R_m of the time-table problem resource requirement array R, from the sets Γ_1, Γ_2, ... , Γ_m of bijections associated with the rows of R, to produce solution rows S_1, S_2, ... , S_m is that, for every integral i = 1, 2, 3, ... , m and indices k(1), k(2), ... , k(i) such that $1 \leq k(1) < k(2) < \ldots < k(i) \leq m$ there exists at least i distinct bijections

$$\Delta_{k(1)}, \quad \Delta_{k(2)}, \quad \ldots, \quad \Delta_{k(i)}$$

where

$$\Delta_{k(i)} \varepsilon \; \Gamma_{k(i)}$$

for each index.

This is an application of Theorem 2.5.1. The implication of the theorem on the solution method is important. The above statement indicates that if any subset of the rows of R do not have distinct bijections, then no solution to the problem can exist. (i.e. ... it is no possible to assign the resource vectors into time-periods without violating the defined conditions.) The theorem is applied to the solution method within the Implication Algorithm at each stage of the solution. (see section 6.2 of chapter 6).

Before proceeding with a discussion of the solution method for the practical problem, the extent of the solution space will be briefly discussed.

4.4 THE SOLUTION SPACE FOR THE TIMETABLE PROBLEM

Consider the timetable problem with $|C| = m$, $|T| = k$, $|E| = \alpha$ and p time-periods in a daily time-span.

The following is an application of earlier work of V. Portugal (42) to the solution method of this thesis.

If the elements (resource vectors) of the solution array S are considered as co-ordinates, then each timetable solution, irrespective of feasibility with respect to the defined conditions and constraints, can be represented by a point in an m x p dimensional space of timetable solutions, i.e. ... there are mp co-ordinates per timetable. The search for a solution, is a search for a point in this solution space.

If any point can assume any of the k teacher resources, then the number of points in the space will be of order

$$k^{mp}$$

When constraint 2 is imposed (all the activities that are included in the resource vectors of the requirement array must be assigned), the number of solution points reduces to

$$(p!)^m$$

i.e. ... m rows (classes) of p resource vectors, (time-periods) each row having p! arrangements.

When the simple problem is considered with respect to constraint 1, the first row of S may be arranged in p! ways. Portugal shows that the solution space is restricted to $C^m_p p!$ points, since for each of the m rows, p resource vectors are selected.

For the practical situation, the solution space is reduced further. Repetitions of resources within the resource vectors within rows of R have an effect.

It will be shown that the algorithms of chapters 5 and 6 will consider only feasible bijections with respect to the class-requirement rows of R, at each stage of the solution method. The number of feasible solutions is dependent upon the resource vector availabilities, block-period requirements, teacher-class set requirements involving inter-row dependencies, resource repetitions within the resource vectors, and the effects of previously assigned resource vectors. The solution method determines, that every un-assigned row of R may still be assigned in S, by the calculation of the feasibility of the remaining solution images through the resource vector availabilities. If the number of images for an unassigned vector is reduced to zero an immediate indication is given through the Implication Algorithm. (See chapter 6, section 6.2).

4.5 THE PRACTICAL PROBLEM

The constraints imposed on the simple problems apply to the practical situation. Teacher-class sets are included, and define the related activities of constraint 1. They involve several teacher and class resources, that must be assigned to a common time-period. In this respect, the requirement involves more than one row of R for assignment. This is the only case, where common resources in difference resource vectors are assigned to the same time-period of the timetable solution.

For the practical problem, a resource vector may involve more than 2 resources (class and teacher). Equipment, room, other teacher and class resources may also be required for one activity. All requirements are expressed within the resource vectors and an example of a more comprehensive resource description follows :-

e.g. $R_2 = ((t_1, q_4), (t_1, t_2, C_1), (t_3, O_2, q_5))$

where t = teacher, q = equipment, C = class,

O = room resources .

The number of elements of the resource vectors does not necessarily have to be the same for every activity. It is noted that the resource vectors described the resources for every activity of the timetable problem.

The resource set E contains other subsets, namely room and equipment resource subsets.

These are defined by

$$Q = \{q_1, q_2, \dots, q_\varepsilon\}$$

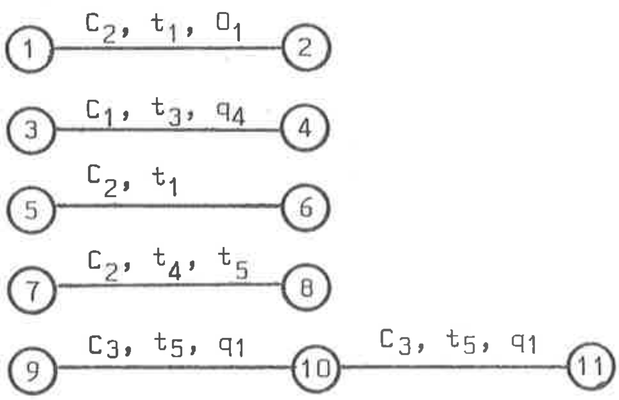
$$O = \{O_1, O_2, \dots, O_{\varepsilon'}\}$$

for equipment and room facilities respectively, and

$$E = T \cup C \cup O \cup Q$$

EXAMPLE 4.6

The following resource activity paths have been assumed to have been constructed from given course structures.



The associated resource requirement array is constructed in a similar manner to that of the simple problem, with resource vectors occurring in the class rows of R.

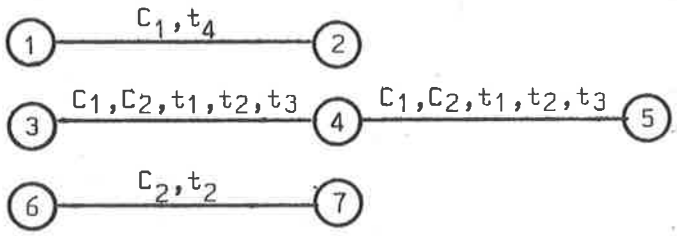
$$R = \begin{bmatrix} (t_1, O_1) & (t_3, q_4) \\ (t_1) & (t_4, t_3) \\ (t_5, q_1) & (t_5, q_1) \end{bmatrix}$$

The order of the elements within the resource vectors is not important. Note that the resources (t_5, q_1) are involved in two consecutive activities and appear as two activity descriptions for row R_3 of R.

For clarity, an example of a teacher-class set resource vector is also given. It can be seen that the activity path description is adequate to indicate teacher-class sets, but further description is needed for block-periods and fixed time-period requirements.

EXAMPLE 4.7

There are 2 classes in the problem, and they are required to meet together with teachers t_1, t_2, t_3 for 2 activities. The following set of activity paths describe the meetings.



The order of the activities is not important in the activity paths, and the node numbers (events) are only displayed for the convenience of defining a particular activity.

Once the resources are defined within the resource requirement array, the order of the resource vectors become fixed. The resource requirement array for this example is :-

$$R = \begin{bmatrix} (t_4) & (C_2, t_1, t_2, t_3) & (C_2, t_1, t_2, t_3) \\ (t_2) & (C_1, t_1, t_2, t_3) & (C_1, t_1, t_2, t_3) \end{bmatrix}$$

The teacher-class sets are described in rows R_1 and R_2 of R since they involve classes C_1 and C_2 associated with these rows. No indication has been given at this stage of block-periods or fixed time-periods.

It will be shown in chapter 5, section 5.2, that availability vectors for the resource vectors of R when calculated, define all time-periods available for each resource vector. By approaching the problem in this manner, and reducing the availability vectors after each row of R is assigned in S, the infeasible situations (those that reduce availability vectors to zero) are easily determined.

The basic special requirements of block-periods and fixed time-periods will now be discussed.

Consider firstly the fixed time-period requirement. Associated with each resource vector r_{ij} defined from the activity $\bar{a}_i$, with resource vector r_i, that involves class C_i, is a non-negative integer f, such that :-

$$0 \leq f \leq p$$

A mapping ρ_i associates with each resource vector r_{ij} of R_i, a member of the set 0, 1, 2, ... , p such that :-

$\rho_i(r_{ij}) = 0$ implies complete freedom of assignment for r_{ij} within the available time-periods.

$\rho_i(r_{ij}) = f$ implies that r_{ij} must be assigned to time-period f of the daily time-span.

Hence the family F = $(\rho_1, \rho_2, \ldots, \rho_m)$ associates with each required resource vector, a fixed time-period indicator. It will be shown that when a resource vector has a fixed time-period f, that the associated availability vector is reduced to a 1 in position f and zero elsewhere. The resource vector is thus assigned to time-period f by restricting the image portions for this activity to the one period. This will be discussed fully in chapters 5 and 6.

Block-periods will now be formulated. A surjective mapping ψ_i, associates with each resource vector r_{ij} of R_i, a member of the set 1, 2, 3, 4, 5 of integers, that indicate the required block-period sizes.

$\psi_i(r_{ij}) = b$ where $1 \leq b \leq 5$ indicate the size of the block-period required for the resource vector r_{ij}.

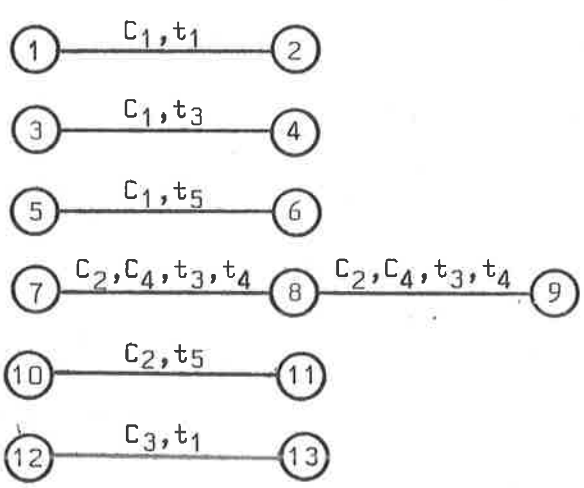
Since, by definition every activity is included in R, and each activity has a duration of one time-period, then there must be b occurrences of resource vector r_{ij} in row R_i. Each will have an associated block-period indicator of size b.

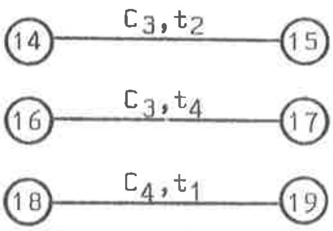
i.e. $\psi_i(r_{ij(1)}) = b, \ \psi_i(r_{ij(2)}) = b, \ \ldots, \ \psi_i(r_{ij(b)}) = b$

Then $B = (\psi_1, \psi_2, \ldots, \psi_m)$ define all block-periods within the resource requirement array R.

EXAMPLE 4.8

Consider the following activity paths





Special Requirements :-

1. activity (1, 2) must be fixed in time-period 1 of the solution.
2. activities (7, 8), (8, 9) must occur as a block-period of size 2 in the solution.
3. activity (14, 15) must be fixed in time-period 1 of the solution.

The following resource requirement array is constructed from the activity paths

$$R = \begin{bmatrix} (t_1) & (t_3) & (t_5) \\ (C_2,t_3,t_4) & (C_4,t_3,t_4) & (t_5) \\ (t_1) & (t_2) & (t_4) \\ (C_2,t_3,t_4) & (C_2,t_3,t_4) & (t_1) \end{bmatrix}$$

Note that R can not be a solution since activity (14, 15) is not in the first time-period (column 1), resource t_5 is in column 3 twice and is not a teacher-class set requirement and resource t_1 occurs twice in column 1.

Associated with the rows of R are the following fixed time-period mappings

$F = (\rho_1, \rho_2, \rho_3, \rho_4)$ where

$$\rho_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\rho_2 = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\rho_3 = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\rho_4 = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \end{pmatrix}$$

i.e. ρ_1 indicates that the resource vectors in columns 1, 2 and 3 respectively, of row R_1 of R are to be assigned such that :-

resource vector r_{11} is fixed in time-period (column) 1 of the solution.

resource vectors r_{12}, r_{13} are assigned freely in time-periods (columns) 2 and 3.

In chapter 5, section 5.3, the mappings F are stored more conveniently as the images

$$F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where the time-periods (column numbers related to R) have been omitted. This association of the resource vectors with the fixed time-perio indicators of F can be assumed since the portions of the vectors within R are fixed.

Similarly the block-period mappings $B = (\psi_1, \psi_2, \psi_3, \psi_4)$ are defined by

$$\psi_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\psi_2 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 1 \end{pmatrix}$$

$$\psi_3 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\psi_4 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 1 \end{pmatrix}$$

and are stored as

$$B = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{bmatrix}$$

where once again the association with the resource vectors of R are assumed.

i.e. resource vectors r_{21} and r_{22} must occur as a block-period size 2 in the solution.

A feasible solution to the above problem is given at this stage without derivation. Further examples in chapters 5 and 6 will indicate the solution method.

$$S = \begin{bmatrix} (t_1) & (t_5) & (t_3) \\ (C_4,t_3,t_4) & (C_4,t_3,t_4) & (t_5) \\ (t_2) & (t_1) & (t_4) \\ (C_2,t_3,t_4) & (C_2,t_3,t_4) & (t_1) \end{bmatrix}$$

Note that the teacher-class set in rows S_2 and S_4 of S involve common resources t_3, t_4, C_2, C_4. The activities involved occur in a block-period size 2 that has been assigned in the solution.

In the above problem the position of the block-period was not specified within the timetable solution. As mentioned in chapter 3, section 3.2 start-periods defined where the block-periods may occur in the practical case. These are now formally described.

Start-periods for each block-period size b are defined by the surjective mappings $_b$ that maps the elements 1, 2, 3, ... , p relating to the time-periods of the school day, into the binary numbers 0, 1 in the following way :-

$$\tau_b(j) = \begin{cases} 1 & \text{if block period size b } \underline{\text{can}} \text{ be started in time-period j} \\ 0 & \text{otherwise} \end{cases}$$

EXAMPLE 4.9

For block-period size 2 the following surjection

$$\tau_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

defines start-periods 1, 3 and 4 as the only legal positions for the beginning of any block-periods of size 2.

Hence the following double periods are allowable

1-2, 3-4, 4-5

The block start mappings are stored in an array BS in image form as are the fixed and block requirements.

All other special requirements of the South Australian secondary schools can be incorporated within the above formulation, together with the solution method algorithms of chapters 5 and 6. It will be shown that the availabilities (image positions for the assignment mappings) have an important role in the solution procedure.

CHAPTER 5

THE BIJECTIVE MAPPING GENERATOR

5.1 INTRODUCTION

The solution method described in this chapter is an individual approach to the solution of the school timetable problem. The notions of required resource vectors (section 4.3, chapter 4) and feasible bijective mappings (section 5.4) for the determination of the solution timetable are discussed. The terms resource vector and required-resource vector, are used to mean the vector of resources that are required for an activity. These requirements are obtained from the resource activity paths of chapter 4, section 4.2. Class requirements, each consisting of p required-resource vectors, are considered as complete assignable units. i.e., all activities relating to a class are assigned together. Hence, in this solution method there are m permanent assignment stages for the m classes of a school. This is in contrast to the m x p assignment stages of many other methods (see section 1.1, chapter 1).

The advantages of the class unit approach are, firstly, that the number of assignment stages has been reduced by a factor p. Secondly, the inter-relationships between resource vectors of the unassigned classes can be considered with respect to a larger group of assigned vectors (section 6.3 of chapter 6). It will be shown that this concept has important implications as discussed in chapter 6. Thirdly, the teacher-class sets (section 3.2, chapter 3) establish a relation between classes through their resource vectors. Thus the class approach is particularly relevant to these requirements.

The main disadvantage comes from the amount of testing required to determine the implications of a class assignment. Although extra time was involved in testing the more complex class requirements, the method was nevertheless adopted for the following reasons. First, it permitted a significant reduction in the number of possible solutions to be investigated after each assignment stage before a final solution was produced (see section 5.3). Second, the method was exhaustive in its approach, permitting the early recognition of unfeasible situations. Third, the method was directly applicable to the South Australian secondary school situation, where classes are considered as units within a school.

At each stage of the solution method the required-resource vectors of a row of the resource-requirement array (section 4.3, chapter 4) must be mapped from their existing positions in that row, into new positions in the solution row. This translation, or assignment, is subject to the conditions and constraints stated in chapter 4, together with school policies included in the algorithms of chapters 5 and 6. The available positions in the solution row, for each resource vector of the requirement row are calculated in the form of <u>composite availability vectors</u> (CAV), described in section 5.2

The CAV reduction algorithm (section 5.3) has an important role in identifying and rejecting unfeasible mappings. The implication algorithm of section 6.3, chapter 6, applies the CAV reduction algorithm extensively, to investigate many of the implications of a class assignment.

An assignment is determined from the CAV by generating a mapping of the column numbers, related to time-periods, onto themselves such that the images are members of the associated CAV (section 5.4) of the requirement row.

This chapter describes both the CAV and their reduction, together with the bijection generator. The reduction algorithm and generator when combined with the algorithms of chapter 6, provide the solution method to the school timetable problem.

5.2 THE COMPOSITE AVAILABILITY VECTORS

From chapter 4, section 4.3, a required-resource vector is a subset of the school resources, that are required for a timetable activity. Associated with each required-resource vector is a composite availability vector (CAV), constructed from the availability vectors (chapter 4, section 4.2) of each resource member. It will be shown in this section that the CAV define completely, all feasible bijective mappings for each row of the requirement array, at each stage of the solution method.

The resource availability array A, (section 4.3, chapter 4) defines the availability for assignment of each resource, for each time-period of a daily time-span. $A(\beta)$ denotes the column availability for the resource $\beta \varepsilon E$ of the school resource set $E = \{1, 2, 3, \ldots, \alpha\}$. This daily availability is given by

$$A(\beta) = \begin{bmatrix} \theta_1(\beta) \\ \theta_2(\beta) \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \theta_p(\beta) \end{bmatrix}$$

where

$$\theta_j(\beta) = \begin{cases} 1 & \text{if resource } \beta \text{ is available to be assigned to time-period } j \text{ in the solution} \\ 0 & \text{otherwise} \end{cases}$$

The rules governing the logical union ($\cup$), logical not ($\sim$), and logical intersection ($\cap$) of the binary numbers 0, 1 are given below :-

logical union : $0 \cup 0 = 0$

$1 \cup 1 = 1 \cup 0 = 0 \cup 1 = 1$

logical intersection : $1 \cap 0 = 0 \cap 1 = 0 \cap 0 = 0$

$1 \cap 1 = 1$

logical not : $\sim 1 = 0 \qquad \sim 0 = 1$

Consider two resources $\beta_1 \varepsilon E$, $\beta_2 \varepsilon E$ and the availability vectors associated with them, namely $A(\beta_1)$ and $A(\beta_2)$. When the above logical operations are applied to the availability vectors, it is implied that the operation occurs between the p corresponding elements $\theta_j(\beta_1)$ and $\theta_j(\beta_2)$ for j = 1, 2, ... , p.

$$\text{e.g.} \quad A(\beta_1) \cap A(\beta_2) = \begin{bmatrix} \theta_1(\beta_1) \cap \theta_1(\beta_2) \\ \theta_2(\beta_1) \cap \theta_2(\beta_2) \\ \vdots \\ \theta_p(\beta_1) \cap \theta_p(\beta_2) \end{bmatrix}$$

Thus, for example, if the third element in the column vector $A(\beta_1) \cap A(\beta_2)$ is unity then we are being told that both β_1 and β_2 are available in period 3. If, by contrast, the third element is zero, we are being told that either β_1 or β_2 or both β_1 and β_2 are not available in period 3.

The <u>composite availability vector</u> for the resource vector r_{ij}, is denoted by $A^*(r_{ij})$. The resource vector is a q-sample $(\beta_1, \beta_2, \ldots, \beta_q)$ of the school resource set $E = \{1, 2, 3, \ldots, \alpha\}$, and describes all the resources required for an activity for the class C_i (see section 4.2, chapter 4). The CAV defines, by entry of unity, the time periods in which all the elements of r_{ij} are simultaneously available, and, by an entry zero, defines the time periods in which at least one of the r_{ij} is not available.

$$A^*(r_{ij}) = A(\beta_1) \cap A(\beta_2) \cap \ldots \cap A(\beta_q)$$

The CAV has the same structure as the resource availability vector $A(\beta)$, in that it is a column vector of p binary elements indicating the availability or non-availability of each time-period of the solution timetable, for the resource vector r_{ij}. Every resource vector of the resource requirement array R (section 4.3, chapter 4) has an associated CAV.

The <u>composite availability array</u>, denoted by A_i^* combines the CAV, p in number, associated with the p required-resource vectors r_{i1}, r_{i2}, ... , r_{ip} of row R_i of R. The array is given by :-

$$A_i^* = A^*(R_i) = (A^*(r_{i1}) \quad A^*(r_{i2}) \ldots A^*(r_{ip}))$$

The CAA is a square, binary, p x p array, indicating the availability or non-availability for assignment, of the p activities associated with class C_i of a school, to each time-period of the solution timetable.

An example, to demonstrate the construction of the CAV and CAA is now given.

<u>EXAMPLE 5.2</u>

Consider the resource availability array A given by :-

		resources					
		β_1	β_2	β_3	β_4	β_5	β_6
	1	1	1	1	1	1	1
time-	2	1	1	1	1	1	1
periods	3	0	1	0	1	1	1
	4	1	0	0	1	1	1

Assume that the following resource requirement array R has been given.

		time-periods (unallocated)			
	1	(β_1,β_4)	(β_2,β_5)	(β_3)	(β_4,β_6)
Classes	2	(β_1,β_2)	(β_3)	(β_4,β_6)	(β_2)
	3	(β_5,β_6)	(β_6)	(β_1)	(β_4)

Consider the resource vector $r_{23} = (\beta_4,\beta_6)$ involving resources β_4,β_6.

The CAV for r_{23} is given by :-

$$A^*(r_{23}) = A(\beta_4) \cap A(\beta_6)$$

$$= \begin{bmatrix}1\\1\\1\\1\end{bmatrix} \cap \begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}1\\1\\1\\1\end{bmatrix}$$

Showing that the class 2 requirement of resources (β_4,β_6) may be satisfied by use of any one of the 4 time-periods.

Similarly the CAV for resource vectors r_{21}, r_{22}, r_{24} are

$$A^*(r_{21}) = A(\beta_1) \cap A(\beta_2) = \begin{bmatrix}1\\1\\0\\1\end{bmatrix} \cap \begin{bmatrix}1\\1\\1\\0\end{bmatrix} = \begin{bmatrix}1\\1\\0\\0\end{bmatrix}$$

$$A^*(r_{22}) = A(\beta_3) = \begin{bmatrix}1\\1\\0\\0\end{bmatrix}$$

$$A^*(r_{24}) = A(\beta_2) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

The composite availability array for row R_2 associated with class C_2 is given by :-

$$A_2^* = (A^*(r_{21})\ A^*(r_{22})\ A^*(r_{23})\ A^*(r_{24})) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Similarly

$$A_1^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad A_3^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

It has been shown in chapter 4, section 4.3 that the images $_j$, $j = 1, 2, \ldots , p$ for the mapping

$$\Delta_i = \begin{pmatrix} 1 & 2 & 3 & \cdots & j & \cdots & p \\ \delta_1 & \delta_2 & \delta_3 & \cdots & \delta_j & \cdots & \delta_p \end{pmatrix}$$

are the new positions within the solution row S_i of S (the solution array, section 4.2, chapter 4) for the required resource vectors r_{ij} of R_i. Recall that the resource vectors within the resource requirement array were not allocated to time-periods even though they occupied specific column positions within the rows of R. The mapping Δ_i <u>allocates</u> (assigns) column positions that are related to the time-periods for the timetable solution to the required resource vectors. The resource vectors of a row R_i are considered as the elements to be allocated to time-periods in the timetable solution, for a class C . A resource occuring within a resource vector is not considered alone, but rather in relation to the other resources also required for that activity described by the resource vector. For a mapping to be feasible the following conditions must be satisfied.

For each required resource vector r_{ij} of row R_i, the composite availability vector $A^*(r_{ij})$ must satisfy the condition that

$$\theta_{\delta j}(r_{ij}) = 1 , \quad j = 1, 2, \ldots , p. \qquad - (5.1)$$

i.e., for the mapping to be feasible, each required resource vector must be available for allocation to time-period δ_j, indicated by the CAV.

The CAV therefore define all feasible mappings for each row of R through the indication of permitted allocation positions within the solution S, for each required resource vector.

For any feasible mapping to exist for a requirement row of R, the following condition, known as Hall's condition (section 2.6, chapter 2) must be satisfied : for the association CAA of a row, the logical union of any k-combination of the column vectors of the CAA, k = 1, 2, 3, ... , p must be such, that the resultant vector contains at least k available allocation positions, for the assignment of the k required resource vectors.

The condition states, that if there exists a k-combination of

CAV such that their union contains at most k-1 available allocation positions, then it is not possible to generate a bijection that assigns the k required resource vectors, since by definition, all images of a bijection must be distinct (see section 2.6, chapter 2).

For the practical problem, Hall's condition is not sufficient to indicate complete feasibility for a given requirement row. Added practicalities such as block-periods that have restricted positions within the solution, require extra considerations for feasibility with respect to the assignable row requirements. It will also be shown that a bijection can be feasible with respect to a requirement row of R but when other rows of R are considered in relation to this mapping it becomes unfeasible (section 5.4, and section 6.2, chapter 6).

It will suffice at this stage, to understand the importance of the CAV in the solution method. They not only define all feasible mappings for each row of R, but also indicate unfeasibilities in the manner described above.

Before the bijection generator is discussed, the CAV reduction algorithm will be given. This algorithm will be shown to reject inadmissible elements from the CAV and is used extensively in the algorithms of chapter 6.

5.3 CAV REDUCTION ALGORITHM

The CAV reduction algorithm considers the CAA associated with a row of R and reduces inadmissible elements (see below) of the CAV to zero. This algorithm resembles a similar reduction algorithm of J. Cisma (9) in an earlier publication on school timetable investigations.

DEFINITION 5.1

An admissible element $\theta^*_{\delta j}(r_{ij})$ of the CAA A^*_i, is an element that can be included in a feasible mapping Δ_i with respect to row R_i of R.

Otherwise the element is said to be inadmissible.

The rejection of inadmissible elements is accomplished in the following steps. Consider the k-combination of CAV for row R_i to be $A^*(r_{ij_1})$, $A^*(r_{ij_2})$ ... $A^*(r_{ij_k})$ and the logical union of the CAV to be given by :-

$$A^*(r_{ij_1}) \cup A^*(r_{ij_2}) \cup \ldots \cup A^*(r_{ij_k})$$

$$= \begin{bmatrix} \theta^*_1 \\ \theta^*_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \theta^*_p \end{bmatrix}$$

where

$$\theta^*_j = \begin{cases} 1 & \text{when at least one required resource vector of the k-combination is available for assignment to time-period } j \\ 0 & \text{otherwise} \end{cases}$$

The reduction algorithm is presented in 3 steps. An example is given after the 3rd step to clarify the reduction techniques.

Step 1

The location of teacher-class set requirements for row R_i. The following algorithm is applied for the rejection of in-admissible elements.

TC.1 Set $j = 1$ (the column number)

TC.2 If $r_{ij} \cap C = \phi$ goto TC.3
else goto TC.4 (where C is the set of classes at the school. This locates a teacher-class set)

TC.3 $j = j+1$; if $j > p$ exit from algorithm ;
else goto TC.2

TC.4 Set $\overline{A''} = A^*(r_{ij})$ the CAV of the required resource vector.

TC.5 Set $j' = 1$ now locate all other involved teacher-class sets of row R_i that are the same as r_{ij}.

TC.6 if $j' > p$ goto TC.8 ;
else goto TC.7

TC.7 if $r_{ij'} = r_{ij}$ set $\overline{A''} = \overline{A''} \cup A^*(r_{ij'})$, $j' = j+1$
goto TC.6 ;
else goto TC.6 with $j' = j'+1$

TC.8 Include class C_i in the resource vector

$r'_{ij} = C_i \cup r_{ij}$

TC.9 Locate other rows related to this teacher-class set requirement

$i' = 1$

TC.10 if $i' = i$ goto TC.12

else if $C_{i'} \varepsilon (r'_{ij} \cap C)$ goto TC.13

TC.12 $i' = i'+1$; if $i' > m$ goto TC.19

else goto TC.10

TC.13 Set $\overline{A'''} = 0$ i.e. consists of p zero elements

TC.14 Set $j' = 1$

TC.15 Determine all resource vectors of row R that are involved in the teacher-class set

$r'_{i'j'} = r_{i'j'} \cup C_{i'}$

TC.16 If $r'_{i'j'} = r'_{ij}$, set $\overline{A'''} = \overline{A'''} \cup A^*(r_{i'j'})$; goto TC.17

else goto TC.17

TC.17 $j' = j'+1$

TC.18 If $j' > p$, set $\overline{A''} = \overline{A''} \cap \overline{A'''}$; goto TC.12

else goto TC.15

i.e., determine the common available time-periods for rows R_i and $R_{i'}$ for the teacher-class set.

TC.19 Reduce all CAV of R_i that are associated with the teacher-class set vector r_{ij}

$A^*(r_{ij}) = \overline{A''}$

TC.20 Set $j' = 1$

TC.21 If $r_{ij'} = r_{ij}$ set $A^*(r_{ij'}) = \overline{A''}$

else goto TC.22

TC.22 $j' = j'+1$, if $j' > p$ goto TC.3

else goto TC.21

The above step locates a teacher-class set requirement in row R_i of R (stage TC.2). It then determines all related resource requirement rows (classes) and calculates the common available time-periods, storing them in $\overline{A''}$ in stages TC.5 to TC.18. In stages TC.19 to TC.22 the CAV of all required resource vectors of row R_i that involve the teacher-class set are reduced. Once teacher-class set requirements for row R_i have been considered block-periods must be investigated (step 2). An application of the complete reduction algorithm will be given later in this section.

Step 2

The investigation of block-period requirements for row R_i.

The nomenclature BR will be used for the algorithm relating to block-period requirements.

BR.1 Set the block-period size

$b = 1$

BR.2 Set column indicator $j = 1$

BR.3 If required resource vector r_{ij} is in a block-period size b goto BR.5 ; else goto BR.4

BR.4 $j = j+1$, if $j > p$ goto BR.13

else goto BR.3

BR.5 Locate all resource vectors $r_{i\pi'}$, $r_{i\pi''}$ such that r_{ij}, $r_{i\pi'}$, $r_{i\pi''}$, ... are in the block-period requirement.

BR.6 Form the logical union of their associated CAV to give the time-periods available for assignment

$\overline{A}' = A^*(r_{ij}) \cup A^*(r_{i\pi'}) \cup A^*(r_{i\pi''}) \quad \ldots.$

(note that there will be b vectors in this union)

where

$$\overline{A}' = \begin{bmatrix} \overline{\theta}_1' \\ \overline{\theta}_2' \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \overline{\theta}_p' \end{bmatrix}$$

BR.7 locate block-period size b start positions in column b of BS array (see section 4.5, chapter 4)

$$BS_b \cap \overline{A}' = \begin{bmatrix} \overline{\theta}_1'' \\ \overline{\theta}_2'' \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \overline{\theta}_p'' \end{bmatrix}$$

where $\overline{\theta}_{j'}'' = 1$ indicates j' is included in the available time-periods for the block-period.

BR.8 Now determine if the periods j'+1, j'+2, ... , j'+b-1 are also available

If $\overline{\theta}_{j'}'' = 1$ form $\overline{\theta}_{j'}'' \cap \overline{\theta}_{j'+1}'' \cap \ldots \cap \overline{\theta}_{j'+b-1}''$

BR.9 If the logical intersection = 1 then all time-periods of the block-period are available and hence $\overline{\theta}_{j'}''$

is a feasible start-period for block-size b.

If zero, then $\overline{\theta}_{j'}''$ not feasible and is reset to zero. This step is carried out for each $\overline{\theta}_{j'}'' = 1$.

BR.10 finally only feasible start positions $\overline{\theta}_{j'}'' = 1$ will remain. If only one remains then the time-periods must be reserved for this block-period.

$$\text{if } \sum_{j''=1}^{p} \overline{\theta}_{j''}'' = 1 \quad \text{then goto BR.11}$$

else goto BR.4

BR.11 Define $A^*(r_{ij}) = A^*(r_{i\pi'}) = A^*(_{i\pi''}) = \ldots\ldots$

such that $\theta_1 = \theta_2 = \ldots\ldots = \theta_{j'-1} = 0$

$$\theta_{j'} = \theta_{j'+1} = \ldots\ldots = \theta_{j'+b-1} = 1$$

$$\theta_{j'+b} = \theta_{j'+b+1} = \ldots\ldots = \theta_p = 0$$

i.e., reserve the only b feasible time-periods for that block-period

BR.12 Then goto BR.4

BR.13 Set b = b+1 ; if b > 5 exit from the algorithm

else goto BR.2

The above section of the reduction algorithm locates the block-period requirements in R_i (at stage BR.3). If a block-requirement is located then an investigation on feasible positions within the solution are considered in stages BR.5 to BR.10. If it is found that the block-period can only be assigned in one position (BR.10) the time-periods involved are reserved for the requirement (stage BR.11). All block-period sizes are considered for b = 1 to b = 5.

Step 3

The next step involves the location of critical blocks (section 2.6, chapter 2).

A critical block is a k-combination of CAV such that their union contains exactly k, one digits. They are not related to block-periods, that define consecutive time-periods for a requirement of the timetable problem. During this 3rd step Hall's condition is also checked, to ensure that a feasible mapping exists for row R_i. (see section 2.6, chapter 2).

The nomenclature HC will be used for the algorithm relating to Hall's condition.

HC.1 Set k = 1 (the size of the k-combination)

HC.2 Have all k-combinations been considered?

If yes, goto HC.9 ; else, continue.

HC.3 Form the logical union of the k CAV in the combination,

i.e., $$A^*(r_{ij_1}) \cup A^*(r_{ij_2}) \cup \ldots\ldots \cup A^*(r_{ij_k}) = \begin{bmatrix} \theta_1^* \\ \theta_2^* \\ \cdot \\ \cdot \\ \cdot \\ \theta_p^* \end{bmatrix}$$

HC.4 If $\sum_{j=1}^{p} \theta_j^* < k$ Hall's condition is violated and no solution for the row R_i can be determined.

Thus the back-track algorithm of section 6.4, chapter 6 is called. (see later)

HC.5 else if $\sum_{j=1}^{p} \theta_j^* = k$ a critical block has been found and the k available positions for allocation must be reserved for these k resource vectors of the defined combination.

HC.6 From lemma 2.6.3, the critical block elements may be deleted from elements of the CAV not in the critical block without violating feasibility conditions for the problem.

Assume the 1's occur in positions

$$\theta^*_{j_1'} , \theta^*_{j_2'} , \ldots , \theta^*_{j_k'}$$

indicating time-periods $j_1', j_2', \ldots, j_k'$ for each required resource vector $r_{i\pi}$ such that

$$r_{i\pi} \not\in \{r_{ij_1}, r_{ij_2}, \ldots, r_{ij_k}\}$$

$$\text{set } A^*(r_{i\pi}) = A^*(r_{i\pi}) \cap \sim \begin{bmatrix} \theta_1^* \\ \theta_2^* \\ \cdot \\ \cdot \\ \cdot \\ \theta_p^* \end{bmatrix}$$

HC.7 If any $A^*(r_{i\pi})$ is such that

$$|A^*(r_{i\pi})| = 0$$

i.e., no assignment positions remain for the vector $r_{i\pi}$ after the reduction then no feasible mapping can exist for R_i. Once again the back-track algorithm is required and will be discussed in the next chapter.

HC.8 else goto HC.2

HC.9 $k = k+1$; if $k > p$, exit from CAV reduction algorithm

else goto HC.2

EXAMPLE 5.2

Consider the following timetable problem with resources C_1, C_2, C_3, t_1, t_2, t_3, t_4 and resource availability array

Resource

		t_1	t_2	t_3	t_4	C_1	C_2	C_3
A =	1	1	1	1	1	1	1	1
time-	2	1	1	1	1	1	1	1
period	3	1	1	1	1	1	1	1

The resource requirement array is defined by

$$R = \begin{bmatrix} (C_2, t_1, t_2) & (C_2, t_1, t_2) & (t_3) \\ (C_1, t_1, t_2) & (C_1, t_1, t_2) & (t_4) \\ (t_3) & (t_4) & (t_4) \end{bmatrix}$$

The block-period array associated with R is given by

$$B = \begin{bmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 2 & 2 \end{bmatrix}$$

where b_{ij} indicates the block-period size of activity r_{ij} of R.

i.e., activities r_{32}, r_{33} must be assigned in a block-period size of 2. in the solution, whilst activity r_{13} is a block size 1.

The following CAA are calculated for each row of R.

$$A_1^* = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix} \quad A_2^* = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix} \quad A_3^* = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

The row R_2 of R will be considered to be the present row for assignment. The CAV reduction steps now begin for i = 2 (row 2).

1. $j = 1$, $r_{21} \cap \{C_1, C_2, C_3\} \neq \phi$

   hence a teacher-class set requirement exists in this position

2. set $\overline{A''} = A(r_{21}) = \begin{vmatrix} 1 \\ 1 \\ 0 \end{vmatrix}$

3. set $j' = 1$ $\quad r_{21} = r_{21}$ hence $\overline{A''} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} \cup \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix}$

   $j' = 2$ $\quad r_{21} = r_{22}$ hence $\overline{A''} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} \cup \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix}$

   $j' = 3$ $\quad r_{21} = r_{23}$ no action

4. set $r'_{21} = C_2 \cup \{C_1, t_1, t_2\} = \{C_1, C_2, t_1, t_2\}$

5. $i' = 1$ $\quad C_1 \in \{C_1, C_2\}$ hence row 1 (class C_1) is related to the teacher-class set

   so set $\overline{A'''} = 0$

   $j' = 1$ $\quad \overline{A'''} = \begin{vmatrix} 0 \\ 0 \\ 0 \end{vmatrix} \cup \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 1 \end{vmatrix}$

$j' = 2 \qquad \overline{A'''} = \begin{bmatrix}1\\1\\1\end{bmatrix} \cup \begin{bmatrix}1\\1\\1\end{bmatrix} = \begin{bmatrix}1\\1\\1\end{bmatrix}$

the available time-periods for row 1 teacher-class set has been calculated.

$i' = 2$ not considered since that is where we found the teacher-class set

$i' = 3$ is not involved.

6. set $\overline{A''} = \overline{A''} \cap \overline{A'''}$

$$= \begin{bmatrix}1\\1\\1\end{bmatrix} \cap \begin{bmatrix}1\\1\\1\end{bmatrix} = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

this gives the resultant common availability for all the teacher-class set CAV. In this case no reduction has taken place since they were all available for the same time-periods.

7. Set $A^*(r_{21}) = A^*(r_{22}) = \overline{A''} = \begin{bmatrix}1\\1\\0\end{bmatrix}$

8. Now consider block-periods (remember only row R_2 is being considered).

$b = 1$

$j = 1$ no block size 1

$j = 2$ no block size 1

$j = 3$ r_{23} has block size 1

9. no logical union of resource CAV is required since r_{23} is the only required vector involved.

$$\overline{A'} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Define

$$BS = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

where a 1 in position (i', j') implies that a block size j' may begin in time-period i'.

Thus

$$BS = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

the first column of BS indicating all feasible start periods for a block-period size 1 (namely all 3 periods are feasible)

Hence

$$BS_{b} \cdot \cap \overline{A'} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

and

$$\sum_{j''=1}^{3} \overline{\theta}''_{j''} = 3$$

and no reservation is necessary.

10. set b = 2

r_{21} has an associated block-size 2 from B, (2, 1)

The other required resource vector in this block-period is r_{22}.

$$\text{form } \overline{A'} = A^*(r_{21}) \cup A^*(r_{22})$$

$$= \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$BS_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

hence

$$BS_2 \cap \overline{A'} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\sum_{j''=1}^{3} \overline{\theta}''_{j''} = 1$$

and hence a reserve situation exists since no other position for the block-size 2 can be found.

Thus periods 1 and 2 must be reserved since $\overline{\theta}''_1 = 1$

$\overline{\theta}''_{1+2-1} = \overline{\theta}''_2 = 1$

Hence $\overline{\theta}'_3 = 1$ is reduced to $\overline{\theta}'_3 = 0$

The outcome is thus $A^*(r_{21}) = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = A^*(r_{22})$

After steps 1 and 2 are completed the CAA for row R_2 is now

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Step 3 is now applied

11. set $k = 1$

There is no k-combination of size $k = 1$ such that $\Sigma\ \theta_j^* = 1$ or $\Sigma\ \theta_j^* < 1$.

12. set $k = 2$

There is a 2-combination of r_{21}, r_{22} such that

$$\sum_{j=1}^{3} \theta_j^* = 2$$

i.e., the two column CAV have only two available time-periods when considered together, namely time-periods 1 and 2.

Thus there must be reserved for the required resource vectors r_{21}, r_{22} and deleted from r_{23}.

Hence $A^*(r_{23})$ becomes $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

The CAV reduction algorithm is now completed for R_2 of R and the resulting CAA is

$$A_2^* = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

From the above simple example it is evident that the CAV reduction algorithm has an important place in the timetable problem in the reduction of unfeasible solution positions for the required vectors. The bijection generator, discussed in the next section of this chapter, uses the reduced CAA to generate an assignment mapping for R_2.

5.4 THE BIJECTIVE MAPPING GENERATOR

The CAV have been reduced by the CAV reduction algorithm (section 5.3), and it is the purpose of the bijection generator to generate a feasible mapping for the assignment row. It is divided into two stages.

(a) The arrangement of the required resource vectors into a descending order of block-period sizes.

(b) The generation of allocation positions for each of the required activities.

The arrangement stage is included so that the more difficult allocations are investigated early in the generation stage. i.e., a block-size 4 is more difficult to place in the solution array than a block-size 2 since there is less flexibility for larger block-period requirements.

For the row R to be assigned the following arrangement stage is applied.

AR.1 Set $k = 1$ (sequence order indicator)

AR.2 Set $b = 5$ (the block-size indicator)

AR.3 if $b = 0$ exit from this stage of the procedure
else goto AR.8

AR.4 For $j = 1$ to p
if r_{ij} is not a block size b or has been considered,
set $j = j+1$ and consider next r_{ij}
else goto AR.5

AR.5 Set requirement resource vector order indicator to k

AR.6 For resource vectors $r_{i\pi'}$, $r_{i\pi''}$, ... included in the block-size b also set order indicator to be k.

AR.7 $k = k+1$, goto AR.4

AR.8 $b = b-1$, goto AR.3

Now each required activity has been assigned an order number. The image allocation positions for each required resource vector are now determined. The required resource vectors are investigated for assignment beginning with the vectors flagged with order 1 and working through until all have been allocated a solution position.

G.1 Set $l = 1$ (the generator resource vector indicator)

G.2 Set ICT(1) = ICT(2) = 0
to be resource availability used by the generator

G.3 If $l = k$; exit from generator

G.4 Set start period image $STS(l) = 1$

G.5 Set End period image $ETS(l) = 0$

G.6 Set $IL = STS(l)$

G.7 If $IL > p$, goto G.17

G.8 If $l = 1$, $ICT(1) = ICT(2) = 0$, goto G.10

G.9 $ICT(l) = ICT(l-1)$

G.10 Determine remaining start periods for resource vector l.

$IST = (\text{starts for } l) \cap (\sim ICT(l))$

G.11 Determine if IL is a legal start

If $IL \notin IST$ goto G.16

G.12 Set $ETS(l) = IL + b - 1$

G.13 For each resource vector in block determine image in range $STS(l)$ to $ETS(l)$. If no image goto G.16

G.14 For $J_1 = STS(l)$ to $ETS(l)$

$ICT(l) = ICT(l) \cup J_1$

G.15 $l \leftarrow l + 1$, goto G.3

G.16 $IL \leftarrow IL + 1$, goto G.7

G.17 $l \leftarrow l - 1$

G.18 If $l = 0$, exit ; no mapping generated.

G.19 $STS(l) = STS(l) + 1$

G.20 If $STS(l) > p$ goto G.17

G.21 goto G.6

<u>EXAMPLE 5.3</u>

Consider the problem quoted in example 5.2 where

$$R = \begin{bmatrix} (C_2, t_1, t_2) & (C_2, t_1, t_2) & (t_3) \\ (C_1, t_1, t_2) & (C_1, t_1, t_2) & (t_4) \\ (t_3) & (t_4) & (t_4) \end{bmatrix}$$

where (C_2, t_1, t_2) is a teacher-class set and must be allocated in a block-size 2 that may only begin in start-period 1.

The CAA for row R_2 has been reduced in example 5.2 to

$$A_2^* = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This composite availability array indicates r_{21} may be allocated to time-periods 1 or 2, r_{22} to time-periods 1 or 2 and r_{23} to time-period 3 only.

Thus the only feasible mappings for row R_2 are defined by this CAA.

The generator now orders the requirement vectors for row R_2. Using algorithm specified by the nomenclature AR,

Set $b = 5$ - no block-size 5 requirements

$b = 4$ - no block-size 4 requirements

$b = 3$ - no block-size 3 requirements

$b = 2$ - have block-size 2 in positions r_{21}, r_{22}

Thus r_{21}, r_{22} are both flagged with $k = 1$

- no other block-size 2

$b = 1$ – have block-size 1 in position r_{23}

Thus r_{23} is flagged with $k = 2$

Hence the order of generating the mapping is that r_{21}, r_{22} must be allocated first, followed by r_{23} second.

Pictorially, an assignment stack (queue) exists as follows

1	r_{21}, r_{22}
2	r_{23}

The start-period for stack-position is $\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = STS(1)$

Hence allocation positions for r_{21}, r_{22} is produced as

1, 2 respectively, i.e., $\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

thus using time-periods indicated as available by A_2^* for r_{21}, r_{22}. We note that the mapping could feasibly have been defined such that r_{21} was mapped to time-period 2 and r_{22} to time-period 1.

Thus time-period 3 remains for r_{23}, i.e., $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

Hence the mapping

$$\Delta_2 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

(Note the mapping

$$\Delta_2 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}$$

would also be feasible.)

For each block-period size, in descending order, the generator determines available image positions for the required resource vectors. Figure 5.1 is a tree structure, used to indicate the procedure of the CAV Reduction Algorithm and the Bijection Generator.

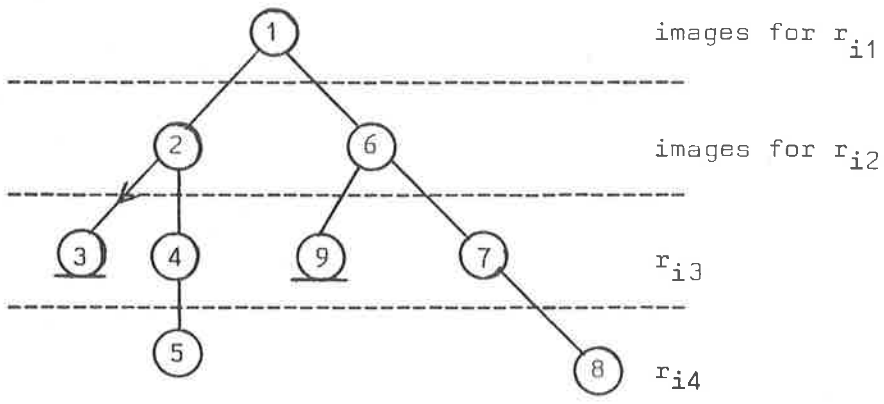


Figure 5.1

Tree structure of Bijection Generation

The CAV reduction algorithm 'prunes' the branches of the tree by reducing images for the associated resource vectors. Each node of the tree is associated with an available time-period for each of the resource vectors of row R_i of R.

i.e. node 1 is for resource vector r_{i1}

node 2 and 6 is for resource vector r_{i2}

node 3, 4, 9, 7 is for resource vector r_{i3}

node 5, 8 is for resource vector r_{i4}

$\underline{\Omega}$ indicates an unfinished mapping.

In this example of Figure 5.1, only two successful mappings exist (through nodes 1-2-4-5, 1-6-7-8). The CAV reduction algorithm has reduced all infeasible images for each resource vector and the bijection generator attempts to determine, from the remaining images, a feasible mapping.

Steps G.6 to G.11 are the selection of images for the resource vectors from available allocation positions. Step G.17 to G.21 indicate a route like 1-2-3 (an unfinished mapping) with no image available for r_{i4}. Hence a new selection r_{i3} is made (node 4) and hence the feasible mapping (with respect to R) of 1-2-4-5. The selection principle is equivalent to the selection of an SDR, described in chapter 2, section 2.6.

The original bijection generator was a permutation algorithm described by Wells (55) called the Johnson-Trotter algorithm. All permutations of images were produced and checked against the CAV's for feasibility. It was found that the time involved with the production of infeasible mappings (discussed in chapter 7) increased as the method proceeded. The new generator just described is more efficient, and includes an important time saving, since the production of infeasible mappings has been reduced for each row.

The following chapter discusses the relationship between the Implication Algorithm, Back-track procedure and the two algorithms of this chapter.

CHAPTER 6

THE IMPLICATION AND BACK-TRACK ALGORITHMS

6.1 INTRODUCTION

The CAV reduction algorithm rejected inadmissible elements (section 5.3, chapter 5) from the CAV associated with an assignable row of the resource requirement array. From the remaining admissible elements, a feasible mapping was generated by the bijection mapping generator (section 5.4, chapter 5). It is the purpose of the implication algorithm (section 6.2) to consider the effect of this mapping on the unassigned required resource vectors of the requirement array. The common resources that are required in both the unassigned activities and the assignable activities of the assignment row are the cause of first order implications. Second order implications are related to the unassigned vectors themselves, and will be discussed in section 6.2.

The reduction stage of the implication algorithm is more extensive than that of the CAV reduction algorithm. Factors such as teacher-class sets, block-periods,fixed-time-periods, and critical blocks (section 6.2, chapter 6) must be considered in relation to the remaining activities and available assignment positions. It will be shown that new critical blocks are produced by the implication algorithm, and must be considered in relation to the remaining allocation positions. Hall's condition (section 2.6) has important applications in this algorithm.

To consider every implication caused by a row assignment would not be economically feasible even with the use of a high speed computer. Hence the problem was considered in two stages. First the row that present the most difficulties for assignment were considered early. To determine the measure of difficulty a heuristic precedence algorithm was designed and has been discussed in section 6.4. The algorithm attempts to determine the most difficult row, not yet assigned, and supplies details to the bijection generator (section 5.4, chapter 5). Two important error detection devices were produced for the heuristic algorithm. They are the clash matrix (section 6.5) and resource load matrix (section 6.6). Both have a practical application in the school timetable problem solution and extensive use has been made of them in the Craigmore High School problem in chapter 8. Second a back-track algorithm (section 6.3) was incorporated, to retrace to previous assignment stages when an unfeasible situation was reached. This algorithm can be forced to consider every feasible bijective mapping at any stage by rejecting them through the implication algorithm. It will be shown later (section 6.3) that the solution method is exhaustive, and is capable of producing every arrangement of activities.

Finally, the assignment algorithm (section 6.7) is discussed. If the implication algorithm determines that no infeasibility is caused by a generated mapping, then the row is assigned, and all data relevent to the assignment is stored. The function of the algorithm is, in general, to store data in order that the back-track algorithm may later retrace to any previous stage as required.

The relationships that exist between the algorithms discussed, are indicated in Figure 6.1 below :-

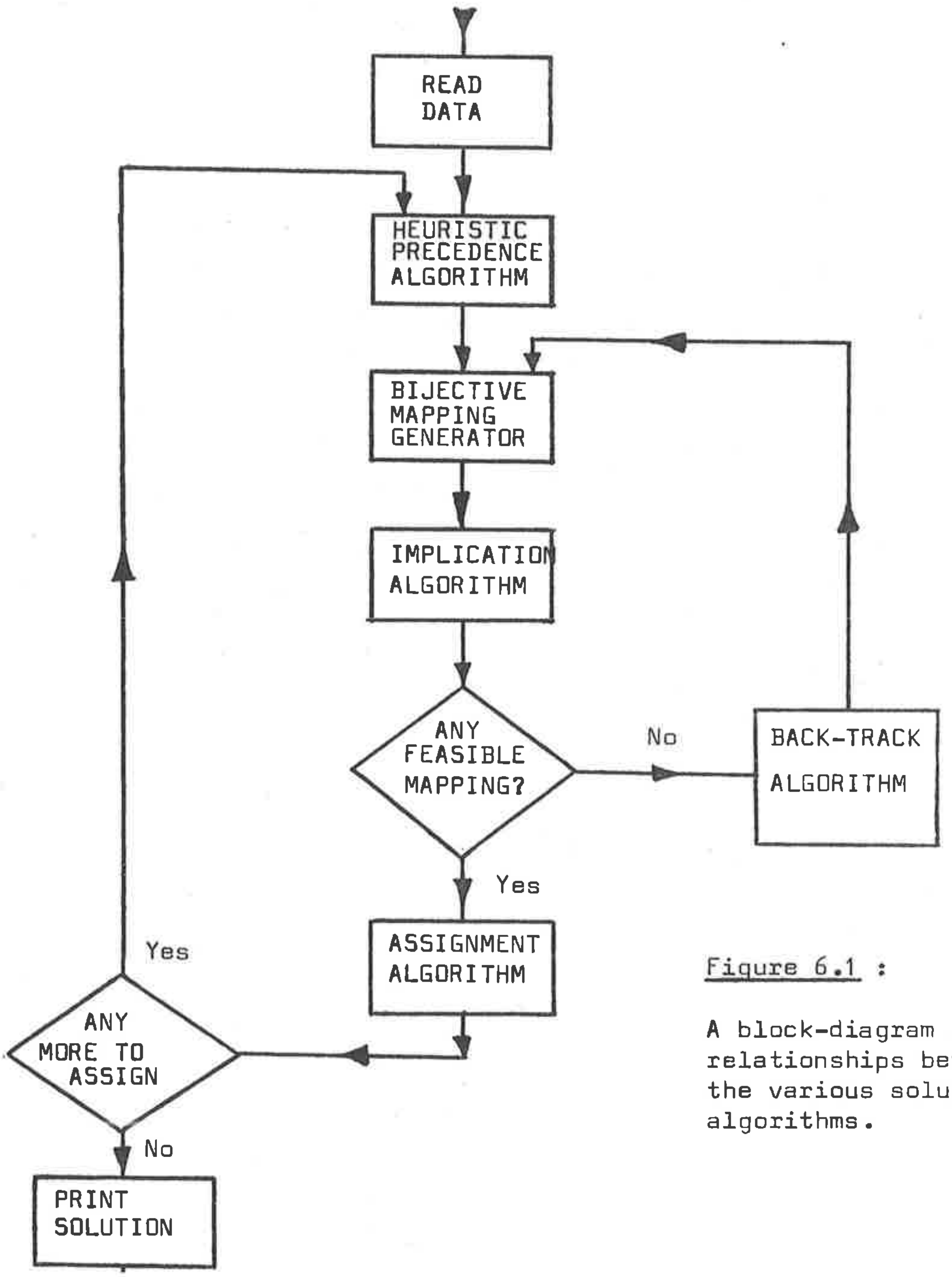


Figure 6.1 :

A block-diagram of relationships between the various solution algorithms.

6.2 THE IMPLICATION ALGORITHM

Consider that the rows R_1, R_2, ... , R_{i-1} have been assigned, using bijective mappings Δ_1, Δ_2, ... , Δ_{i-1} produced by the bijective mapping generator of section 5.4, chapter 5. The next row R_i of the resource requirement array is next to be assigned. The bijective mapping generator generates a mapping Δ_i that is feasible with respect to row R_i (i.e., maps the resource vectors of R_i into available positions in the corresponding solution row S_i). The function of the implication algorithm is to determine the implications of this assignment of row R_i on the remaining unassigned resource vectors of rows R_{i+1}, R_{i+2}, ... , R_m. This is accomplished through the reduction of the CAV (composite availability vectors) associated with each unassigned resource vector.

There are two main stages with the implication algorithm's reduction of CAV. The first stage considers the CAV of resource vectors that involve any of the resources that are to be assigned in row R_i. The period assigned to a resource within the solution row S_i must be rejected from these CAV when common resources are located within unassigned resource vectors. For example, suppose teacher Smith is involved in an activity of row R_i and was allocated to time-period 2 of S_i by the mapping Δ_i (i.e. the resource vector containing Smith is allocated to the second position in row S_i). Then Smith is not available for time-period two in any future assignments, and the time-period 2 is rejected from each CAV of the unassigned rows that involve the resource Smith. This reduction may be complex when teacher-class sets are involved.

The second stage of reduction occurs when considering the unassigned requirements. For example a resource vector with only one remaining assignable time-period in its CAV must be assumed to be temporarily allocated to that time-period. Hence further reductions may occur amongst the unassigned CAV.

If any CAV is reduced to zero (no remaining allocatable time-periods for its associated resource vector), then the mapping Δ_i is said to be infeasible with respect to the unassigned rows of R. Hence it is possible for a mapping Δ_i to be feasible with respect to row R_i but yet be ineligible for use because it is found to be infeasible with respect to the other rows of R.

The second reduction stage of the algorithm is equivalent to locating critical blocks (chapter 2, section 2.6) within the CAA (composite availability arrays) of the unallocated rows of R, and investigating the implications of the critical blocks. This process is demonstrated in the following example 6.1.

EXAMPLE 6.1

Consider the CAA associated with R_2 of a timetable problem where

$$A^*(R_2) = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This demonstrates a critical block of size 3 occurring for resource vectors r_{21}, r_{22} and r_{23} since there exist only 3 available positions for assignment, determined from

$$A^*(r_{21}) \cup A^*(r_{22}) \cup A^*(r_{23})$$

$$= \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Thus the 3 time-periods must be reserved for these resource vectors. Hence they must be eliminated from the CAV of r_{24} to leave

$$A^*(r_{24}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

which itself is a critical block.

From Lemma 2.6.3 the above reduction is justified since it does not affect the feasibility of the CAA (reduction of inadmissible elements for r_{24}). J. Cisma (9) has shown that the rejection of inadmissible elements does not cause Hall's condition to be violated.

Beside critical blocks, sometimes called 'tight situations', the practical requirements of the school timetable problem must be considered. e.g. teacher-class sets. For simplicity, the implication algorithm will be discussed in stages with examples given at each stage to demonstrate the reduction carried out.

Stage 1

Reduction of CAV's that are related to resource vectors (unassigned), that involve common resources to the assignable row R_i of R.

Assume the mapping generated for R_i is Δ_i, described by

$$\Delta_i = \begin{pmatrix} 1 & 2 & 3 & \cdots & p \\ \delta_1 & \delta_2 & \delta_3 & \cdots & \delta_p \end{pmatrix}$$

From the theory of section 2.4, chapter 2, the mapping is characterized by the permutation

$$\Delta_i = (\delta_1, \delta_2, \delta_3, \ldots, \delta_p)$$

The resource vectors of R_i are r_{i1}, r_{i2}, ... , r_{ip}. Hence the resource vector r_{ij} is mapped into position δ_j of the solution row S_i. The resources involved in r_{ij} must be made unavailable for position δ_j for all unassigned resource vectors that also require these resources. The following algorithm rejects allocation positions for involved required resource vectors.

S1.1 For each resource $\beta \in r_{ij}$ do the following steps

S1.2 Set $i_1 = 1$ to be row counter.

S1.3 If row i_1 assigned set $i_1 = i_1 + 1$

else : goto S1.5

S1.4 If $i_1 > m$ exit from stage 1

else : goto S1.3

S1.5 If $i = i_1$ set $i_1 = i_1 + 1$, goto S1.4

else : set $j_1 = 1$ to be column counter

S1.6 If $r_{i_1 j_1}$ is related to r_{ij} through a teacher-class set, set all elements of $A^*(r_{i_1 j_1})$ to zero except the δ_j-element set to 1. goto S1.7

else : goto S1.9

S1.7 $j_1 = j_1 + 1$

S1.8 If $j_1 > p$ $i_1 = i_1 + 1$, goto S1.4

else : goto S1.6

S1.9 Delete element δ_j from $A^*(r_{i_1 j_1})$ if $\beta \in r_{ij}$ and $\beta \in r_{i_1 j_1}$.

else : goto S1.7

S1.10 If $A^*(r_{i_1 j_1}) = 0$; the mapping Δ_i is not feasible with respect to row i_1 of R.

else : goto S1.7

The above algorithm stage 1 is applied for each resource $\beta \in r_{ij}$, for $j = 1, 2, \ldots, p$ thus deleting the allocation positions from future mappings (since constraint 1 states that a resource must not be allocated to two unrelated activities during the same time-period).

If the activity is related through a teacher-class set to an activity in row i_1 of R then the activity $r_{i_1 j_1}$ must also be allocated to time-period δ_j (see teacher-class set requirements, section 3.2, chapter 3). This step is contained in step S1.6 of the algorithm. If the activities are not related then steps S1.9, S1.10 are applied.

In step S1.10 if a CAV is reduced to all zeros then the mapping Δ_i is not feasible with respect to this row i_1 of R and another mapping for row R_i must be generated.

Stage 2

When all first order implications have been considered in stage 1, the second order implications are investigated. These are only in relation to the unassigned resource vectors and the effect that the mapping Δ_i has on them.

Stage 2 is treated in 3 steps. First the CAV reduction algorithm is applied to each unassigned row of R to reject any inadmissible elements brought about by the mapping Δ_i through stage 1. Second, the single available CAV are treated and the implications of these considered.

It will suffice, to briefly mention this first step since the CAV reduction algorithm has been extensively discussed in section 5.3, of chapter 5. The reduction is applied to each unassigned row of R in turn to ensure that each row of R is still feasible with respect to their required resource vectors. The special requirements such as teacher-class sets, and block-periods are also considered in this algorithm. If some requirement can not be met the implication algorithm causes another mapping to be generated for row R_i and the implication algorithm begins again at stage 1. Otherwise the next step in stage 2 is considered.

Any CAV, not assigned, with a single non-zero element must have the indicated time-period reserved for the associated resource vector.

i.e. if there exists a CAV, $A^*(r_{i_1 j_1})$, such that

$$\sum_{j=1}^{p} \theta^*_j = 1$$

where $\theta^*_{j'} = 1$

is the only non-zero element, then time-period j' must be reserved for $r_{i_1 j_1}$.

In essence, this is the same as temporarily assigning the resource vector $r_{i_1 j_1}$ to the time-period j' using a mapping Δ_{i_1} with $\delta_{j'} = j'$. Hence, for this single required resource vector $r_{i_1 j_1}$ we can use the algorithm of stage 1 with $\delta_{j'} = j'$. This algorithm will delete the element j' from each unrelated resource vector CAV, that involves any of the resources $\beta \in r_{i_1 j_1}$.

A simple example will now be given to demonstrate the functions of the implication algorithm.

EXAMPLE 6.2

Consider the problem where

$$R = \begin{bmatrix} (C_2, t_1, t_2) & (t_1, e_4) & (t_4, e_4) & (t_3, e_2) \\ (C_1, t_1, t_2) & (t_1, e_4) & (t_5, e_2) & (t_6, e_2) \\ (t_2) & (t_3, e_2) & (t_3, O_5) & (t_4) \end{bmatrix}$$

where the school resource set E consists of

classes $C = \{C_1, C_2, C_3\}$

teachers $T = \{t_1, t_2, t_3, t_4, t_5, t_6\}$

others $O = \{e_2, e_4, O_5\}$

The composite availability arrays (CAA), associated with each row R_i of R we recall, indicate the availability of the time-periods for assignment of each resource vector of R_i. The columns of A^*_i are associated with the corresponding resource vectors of R_i where the 1st column indicates the availability of the time-periods for r_{i_1}, column 2 for r_{i_2}, etc.

Assume the three CAA for the above requirement array R are given as follows :-

$$A^*_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

indicating complete availability of every time-period for all resource vectors of R_1.

$$A_2^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

indicating for row R_2 that the resource vector r_{23} is not available for time-period 4, and r_{24} is not available for time-periods 3 and 4.

i.e. columns 3 and 4 of A_2^* have zero elements in positions (3, 4) and (4, 3), (4, 4) respectively.

$$A_3^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

indicating resource vector r_{31} is not available for time-periods 3 or 4.

Since the fixed-period requirement allocates time-periods 2 to resource vector r_{31}, this time-period is not available to any of the other resource vectors of row R_3 (since no two activities of a class can be allocated to the same time-period). Hence it is in order, to remove time-period 2 from the other resource vector availabilities. This is done by rejecting the ones in positions (2, 2), (2, 3), (2, 4) of A_3^* associated with the row R_3 resource vectors. Further assume that resource vector r_{31} must be allocated to the 2nd time-period.

i.e. the fixed time-period mapping of section 4.5, chapter 4 is such that

$$F = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \end{bmatrix}$$

Any fixed time-period can be included immediately into the CAA by reduction of the associated CAV. In the case above, we reduce $A^*(r_{31})$ from

$$\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \text{to} \quad \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

thus making all time-periods inadmissible except the second. Note that the CAA also include any restricted time-period constraints for any resources of the school in the same way. (i.e. by the reduction of CAV).

Thus A_3^* becomes

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

In the practical problem, the first algorithm used is the implication algorithm. This action is taken since all problem requirements can be chekced for feasibility before the generation of mappings begin. e.g. do the fixed time-periods requirements cause any infeasibility?

The only CAV causing any reduction is $A^*(r_{31})$ and time-period 2 must be reserved for the required resource vector r_{31}, to leave

$$A_3^* = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

No further reduction of R_3 CAV is possible. Now the effect of $A^*(r_{31})$ is considered with respect to the other CAA.

r_{31} involve resource t_2, hence any other resource vector involving t_2 can not be assigned to time-period 2. (from constraint 1, that states that no resource may be allocated to more than one unrelated activity during the same time-period 2). Hence $A^*(r_{11})$, $A^*(r_{12})$ both involve t_2 and must be reduced by rejecting time-period 2.

Hence

$$A_1^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \text{ becomes } \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

and

$$A_2^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \text{ becomes } \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

No further implications of the problem special requirements cause reductions. Thus at this stage the problem is feasible with respect to the problem special requirements.

Now the generating algorithm is called for row R_1. We will assume that the order of rows to be assigned is R_1, R_2, R_3.

All requirements for row R are single block-periods and thus the bijection order is $r_{11} = 1$, $r_{12} = 2$, $r_{13} = 3$, $r_{14} = 4$.

The allocation positions are indicated by A_1^*.

The first mapping generated for R_1 is

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

That gives a solution row

$$S_1 = ((C_2, t_1, t_2), \ (t_1, e_4), \ (t_4, e_4), \ (t_3, e_2))$$

The implication algorithm checks the effect of the mapping.

The teacher-class set in r_{11} is investigated. This involves r_{21} since (C_2, t_1, t_2) indicates row R_2 through the resource C_2. Remembering that a teacher-class set involves an assignment to the same time-period (in this case period 1) the CAV of r_{21} must be reduced.

Thus

$$A_2^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \text{ becomes } \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Next consider the resources that have been assigned in row R_1. Resources t_1, t_2 can not be assigned elsewhere (except in the teacher-class set requirement of row 2) to time-period 1. Thus any resource vectors involving t_1 and t_2 must be reduced to exclude time-period 1. in rows R_2, R_3.

This involves $A^*(r_{22})$, $A^*(r_{31})$

Thus

$$A^*(r_{22}) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \text{ becomes } \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

and

$$A^*(r_{31}) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \text{ becomes } \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

i.e. stays as it is since time-period 1 is already excluded.

Similarly the 2nd assignment in row R_1 involved resources t_1, e_4 and the 2nd time-period. Hence exclude this time-period from all CAV in rows R_2 and R_3 that involve resources t_1, e_4.

Thus

$$A^*(r_{22}) = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \text{ becomes } \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Similarly for (t_4, e_4) in time-period 3

and (t_3, e_2) in time-period 4

to give

$$A_2^* = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \text{ and } A_3^* = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now the CAV reduction algorithm considers the CAA of R_2 and R_3 to consider unassigned vector implications.

Consider A_2^*.

The effect of a singular in $A^*(r_{21})$ involves the deletion of time-period 1 from columns 2, 3, 4 to give

$$A_2^* = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The singular in column 2 has no effect but the singular in column 4 reduces column 3.

Finally

$$A_2^* = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Thus an extensive reduction has occurred on A_2^* resulting in all singular CAV.

A_3^* is considered in the same way by the CAV reduction algorithm and it is left to the reader to determine that A_3^* becomes

$$A_3^* = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The solution row R_1 is adopted since no infeasibility is located with respect to the mapping Δ_1 on the unassigned resource vectors.

The next mapping for R_1 is the singular mapping, i.e. only one exists.

Namely

$$\Delta_2 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 3 & 2 \end{pmatrix}$$

The mapping has the effect of reducing

$$A_3^* = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{to} \quad \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and hence

$$\Delta_3 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 2 & 4 \end{pmatrix}$$

Thus the solution

$$S = \begin{bmatrix} (C_2, t_1, t_2) & (t_1, e_4) & (t_4, e_4) & (t_3, e_2) \\ (C_1, t_1, t_2) & (t_6, e_2) & (t_5, e_2) & (t_1, e_4) \\ (t_3, e_2) & (t_2) & (t_3, O_5) & (t_4) \end{bmatrix}$$

From the above relatively simple example, the extent of the implication algorithm can be seen. However, in general not every implication has been fully considered, since reductions caused by the second stage could cause new implications for the CAV. This is not reconsidered since it was found that the majority had been sufficiently considered, and the back-track algorithm and heuristic precedence algorithms reduce the probability that unfeasible situations will be caused by unconsidered possibilities. The back-track algorithm will now be discussed.

6.3 THE BACK-TRACK ALGORITHM

With the practical timetables that have features with extensive implications, the time for testing would be considerable. This method of solution disregards any further implications of the rows at the expense of the possibility of an infeasible situation arising. To minimise the possibility of infeasibility occurring a heuristic precedence algorithm was incorporated. This will be discussed in the next section.

Consider that the i-th stage has been reached and that the implication routine indicates that no feasible mapping with respect to the unassigned resources of R can be determined for R . The algorithm recreates the situation before the previous stage mapping was assigned and a new mapping for stage (i-1) is produced. If no more mappings remain for (i-1) then a revision of stage (i-2) is made. If stage 1 is reached by the back-tracking algorithm, and no mapping can be generated for this stage 1, then all feasible mappings for stages 1, 2, ... , i have been considered with no result. Hence no solution can be found for a subset of rows of R and no solution exists to the problem.

It can be seen from this outline of the back-track algorithm that the solution method is exhaustive. A flow chart of the back-track algorithm is given in Figure 6.2.

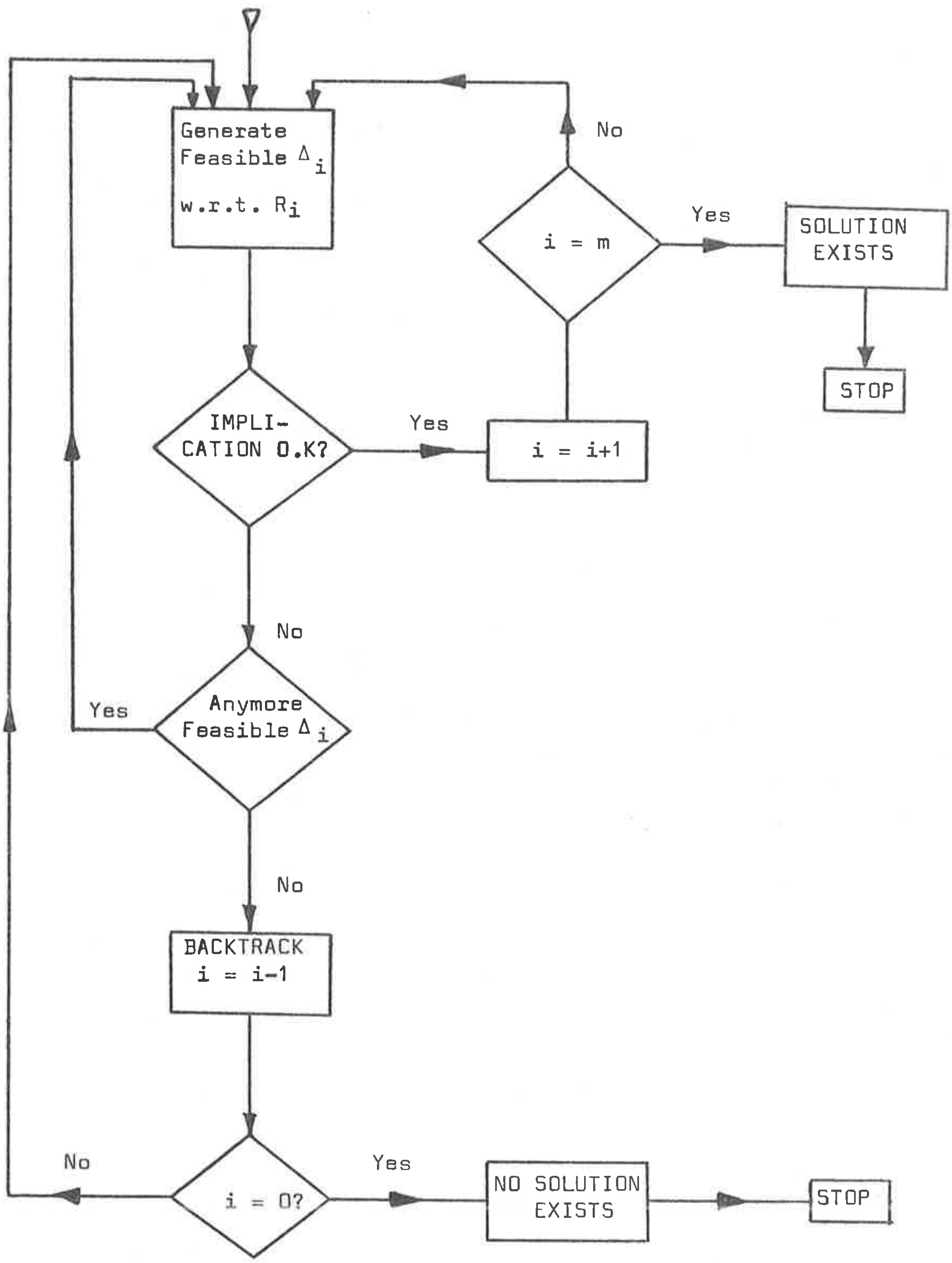


Figure 6.2 : Flow Chart of Back-track procedure.

To minimise the work of the back-tracking algorithm it was found desirable that good mappings were produced early. Rows of R that would have severe effects on other rows of R should be considered first in an effort to reduce the length of back-tracking when required. This led to the construction of the heuristic precedence algorithm.

6.4 THE HEURISTIC PRECEDENCE ALGORITHM

Manual investigations have shown that different degrees of difficulty are associated with various special requirements in the timetable problem. The heuristic precedence algorithm makes use of these investigations by listing the special requirements in order of difficulty. Also taken into consideration was an objective stated in chapter 3, section 3.4. This objective detailed the desirability of creating the timetable solution considering the upper (4th and 5th) year level classes at a school first, and working back to the 1st year level classes last. Using this technique has two advantages. First, if a complete solution can not be determined due to some infeasibility in the 1st year or 2nd year levels a partial result will have been produced for the upper levels of the school while corrections are made. Second, the complex teacher-class sets and block-period requirements are usually located in the upper level requirements and are more easily considered early in the assignment procedure.

The manual techniques indicate the following order of difficulty. Teacher-class sets, block-periods, fixed time-periods, part-time staff in that order. The following computer precedences were calculated on this basis. It should be noted at this stage, that precedences are recalculated **after** each assignment stage to faciliate the changing order of difficulty that arises due to the availability reduction stages of the implication algorithm.

LEVELS

L1 Teacher-class sets with block-periods involving these sets, block-periods outside the sets, fixed time-period (singular availability vectors) and the extent of teacher-class set row involvement.
e.g. a set involving 3 rows of R would be considered before a 2 row involvement.

L2 Teacher-class sets, block-periods, without singular availabilities.

L3 Teacher-class sets, with block-periods not involving the sets and with singular availabilities.

L4 Teacher-class sets with block-periods outside the sets, without singular availabilities.

L5 Teacher-class sets and no block-periods with singular availabilities.

L6 Teacher-class sets.

L7 Block-periods with singular availabilities.

L8 Block-periods - no singulars.

L9 Singular availability resources.

L10 Any remaining rows.

The level of precedence decreases from L1 to L10. In the production of the precedence algorithm two important matrices were used. These were the Resource Load Matrix and Clash Matrix.

6.5 RESOURCE LOAD MATRIX

The resource load matrix summarizes the total number of time-periods required by each resource, to meet the requirements defined in the resource requirement array. This summary is presented in the form of a table or matrix where entries in column 1 indicate the number of time-periods required and column 2 the number of available time-periods for the resource. Rows are indexed by the resource code, discussed in later chapters. In essence, each row of the matrix is associated with a particular distinct resource, thus giving a complete picture of the total involvement of every resource in the timetable pattern.

The matrix is formed for each daily timetable problem, and by scanning each daily requirement for a resource a tally of the weekly load can be determined. It should be noted that teacher-class set requirements involve the same resources for an activity and although several classes are involved the number of time-periods required is still only one. (The only situation where common resources may be allocated to the same time-period for different classes, see chapter 3, section 3.2)

The resource load matrix has two major uses. First it is used to determine precedence when several requirement rows have the same precedence level as discussed in the previous section 6.4. Rows that have heavily loaded resources will then be given a higher precedence within that level. For example, if two rows R_1 and R_2 are on the same precedence level, R_1 involves resources that have loads of 6, 7 and 8 time-periods while row R_2 involves resources with loads of 5 and 6, then row R_1 will be given a higher priority.

The resource load matrix changes as the assignments are made. The matrix in effect keeps a tally of the number of remaining time-periods required by each resource after each assignment stage.

EXAMPLE 6.3

Consider resources involved in example 6.1. From the resource requirement array, and knowing that resource t_2 is not available for time-periods 3 and 4, t_5 is not available for time-period 4, t_6 is not available for time-periods 3 and 4 the following resource load matrix is compiled.

Resource	Load	Availability
t_1	3	4
t_2	2	2
t_3	3	4
t_4	2	4

Resource	Load	Availability
t_5	1	3
t_6	1	2
e_2	4	4
e_4	3	4
0_5	1	4

Note : (1) the class resources can be omitted since they are always fully loaded as they are involved in every daily time-period.

(2) resources t_1, t_3, e_2, e_4 are heavily committed.

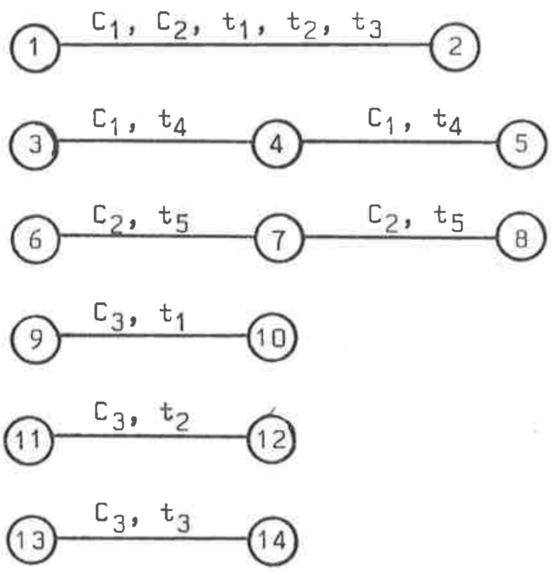
Secondly, the resource load matrix is used when no solution can be determined for a given problem. This is useful for the fault location of over-committed resources (i.e. over-committed for 5 out of 4 available time-periods) and for altering loadings when errors must be corrected in 'no solution' situations. School administrators use the load matrix in conjunction with the clash matrix of section 6.6 for re-allocating requirements (see chapter 8, section 8.4). The clash matrix will now be discussed.

6.6 THE CLASH MATRIX

The clash matrix is an important error detection and correction aid, (described in detail later), designed in conjunction with the resource load matrix (section 6.5). It is constructed by the computer program from the required activities of the timetable problem (see example 6.4). The clash matrix is generated for two main purposes. Firstly, it is used to indicate activities of the timetable problem that cause an infeasibility within the problem definition. This situation will arise when a combination of resources required for a class activity involves at least one resource of each activity related to another class.

EXAMPLE 6.4

Consider a timetable problem defined by the activity paths



The activity (1, 2) deletes any possibility of an allocation of the class C_3 activities since (1, 2) involves all the teacher resources required by C_3 activities. Recall that no unrelated activities involving the same resource may be allocated to the same time-period (chapter 4, section 4.2). The clash matrix indicates this type of infeasibility, and can be used to consider the effects of various combinations of activities allocated to the same time-period (see later in section 8.3, chapter 8).

Secondly, the clash matrix can be useful for redefining the resource combinations identified by the computer program as infeasible. Since the manual interaction during the timetable construction has been minimized through producing the solution by computer, it is necessary to indicate the existing combinations of resources for activities so that manual alterations can be made with little difficulty. A practical example of the use of the clash matrix is given in section 8.3, chapter 8. The construction of the clash matrix will now be discussed.

The clash matrix is a binary array with each column representing a distinct activity of the timetable problem. The rows of the matrix represent the same activities that are identified by the columns, and in addition the resource elements of the school resource set are also associated with rows of the clash matrix. To clarify the description consider example 6.3. The resource set $E = \{C_1, C_2, C_3, t_1, t_2, t_3, t_4, t_5\}$ and the distinct activities are (1, 2), (3, 4), (6, 7), (9, 10), (11, 12), (13, 14). These 6 activities are represented in the clash matrix by the first six rows and columns while rows 7 to 14 represent the elements of the resource set E. Thus in general, the activities $\bar{a}_1, \bar{a}_2, \bar{a}_3, \ldots, \bar{a}_r$ are associated with the first r rows and columns of the matrix while the α elements of the resource set E are associated with rows $r+1, r+2, \ldots, r+\alpha$, as shown in diagram 6.1

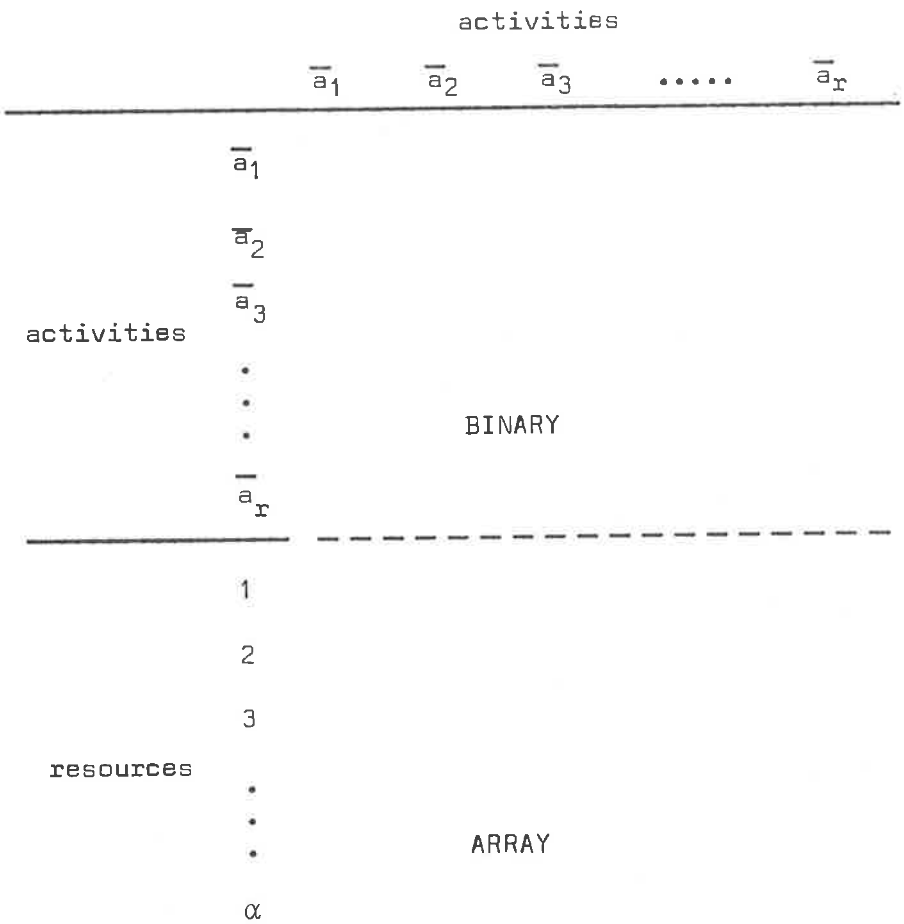


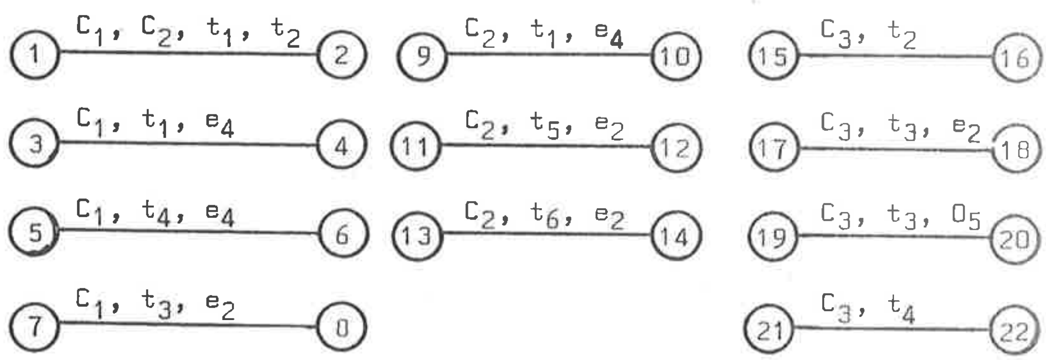
Diagram 6.1 : The layout of the clash matrix

The element (i_1, j_1) of the clash matrix is set to 1 if the activity (resource) of row i_1 does not involve any of the resources (is not a resource) of the activity of column j_1. Otherwise the element (i_1, j_1) is set to zero. Thus a zero entry indicates that activities of row i_1, column j_1 can not be allocated to the same time-period because they involve at least one common resource. The resource section of the clash matrix is included to define each resource combination for the activities $\bar{a}_1, \bar{a}_2, \ldots, \bar{a}_r$, and is useful in assisting manual corrections to infeasible problems, such as those described above in example 6.3. Distinct activities are considered to delete the duplication of information when several activities require the same resource combinations. (i.e. the consecutive activities described in section 4.3, chapter 4).

The most probable area of infeasibility when a problem does not have a solution is in the combinations of resources chosen for the teacher-class set activities of chapter 3, section 3.2. For convenience a clash sub-matrix was produced, and includes only activities that involve 3 or more resources. This sub-matrix does not include the basic single teacher class activities that do not impose far reaching restrictions on the timetable solution. The sub-matrix is merely an option and is mentioned without further discussion. An example of a clash matrix is given in example 6.4 to demonstrate the construction involved.

<u>EXAMPLE 6.5</u>

Consider the activity paths of example 6.1, namely



		(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(11,12)	(13,14)	(15,16)	(17,18)	(19,20)	(21,22)	Row Sum
Activities	(1,2)	0	0	0	0	0	0	0	0	1	1	1	3
	(3,4)	0	0	0	0	0	1	1	1	1	1	1	6
	(5,6)	0	0	0	0	0	1	1	1	1	1	0	5
	(7,8)	0	0	0	0	1	0	0	1	0	0	1	3
	(9,10)	0	0	0	1	0	0	0	1	1	1	1	5
	(11,12)	0	1	1	0	0	0	0	1	0	1	1	5
	(13,14)	0	1	1	0	0	0	0	1	0	1	1	5
	(15,16)	0	1	1	1	1	1	1	0	0	0	0	6
	(17,18)	1	1	1	0	1	0	0	0	0	0	0	4
	(19,20)	1	1	1	0	1	1	1	0	0	0	0	6
	(21,22)	1	1	0	1	1	1	1	0	0	0	0	6
Resources	C_1	0	0	0	0	1	1	1	1	1	1	1	7
	C_2	0	1	1	1	0	0	0	1	1	1	1	7
	C_3	1	1	1	1	1	1	1	0	0	0	0	7
	t_1	0	0	1	1	0	1	1	1	1	1	1	8
	t_2	0	1	1	1	1	1	1	0	1	1	1	9
	t_3	1	1	1	0	1	1	1	1	0	0	1	8
	t_4	1	1	0	1	1	1	1	1	1	1	0	9
	t_5	1	1	1	1	1	0	1	1	1	1	1	10
	t_6	1	1	1	1	1	1	0	1	1	1	1	10
	O_1	1	1	1	1	1	1	1	1	1	1	1	11
	O_2	1	1	1	1	1	1	1	1	1	1	1	11
	O_3	1	1	1	1	1	1	1	1	1	1	1	11
	O_4	1	1	1	1	1	1	1	1	1	1	1	11

Activities

	(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(11,12)	(13,14)	(15,16)	(17,18)	(19,20)	(21,22)	Row Sum
O_5	1	1	1	1	1	1	1	1	1	0	1	10
e_1	1	1	1	1	1	1	1	1	1	1	1	11
e_2	1	1	1	0	1	0	0	1	0	1	1	7
e_3	1	1	1	1	1	1	1	1	1	1	1	11
e_4	1	0	0	1	0	1	1	1	1	1	1	8

Clash Matrix

Note that (a) activity (17,18) is available to be assigned to a common time-period with any of 4 other activities while (1,2) can only be allocated with 3. (shown by row sum).

(b) For the resource section of the matrix, by calculating (r - row sum) = load of resource a column of the resource load matrix can be formed.

EXAMPLE 6.6

Consider the previous example 6.5. From the resource section the following resource load column can be determined.

Resource	Load
C_1	(11-7) = 4 : 4
C_2	(11-7) = 4 : 4
C_3	(11-7) = 4 : 4
t_1	(11-8) = 3 : 4
t_2	(11-9) = 2 : 4
t_3	(11-8) = 3 : 4
t_4	(11-9) = 2 : 4
t_5	(11-10) = 1 : 4
t_6	(11-10) = 1 : 4
O_1	(11-11) = 0 : 4
O_2	(11-11) = 0 : 4
O_3	(11-11) = 0 : 4
O_4	(11-11) = 0 : 4
O_5	(11-10) = 1 : 4
e_1	(11-11) = 0 : 4
e_2	(11-7) = 4 : 4
e_3	(11-11) = 0 : 4
e_4	(11-8) = 3 : 4

Note : the ratio 1 : 4 indicates that the resource is required for 1 of the 4 daily time-periods.

The clash matrix is produced after all the program data has been interpreted by the computer. It is generated, along with the resource load matrix, during the vetting stage of the computer program. Thus anyinfeasibilities for 'over-loaded' resources can be immediately determined. The principles of their use are indicated in the following situations.

Situation 1

A resource has been located that is overloaded. The clash-matrix can be interrogated to locate any teacher-class sets that involve such a resource. The resource may then be deleted from the set, and from the load matrix a suitable replacement may be chosen, thus reducing the load on the 'over-loaded' resource.

Situation 2

A problem has no solution. (see chapter 8, section 8.3). The program prints the clash-matrix which is interrogated to locate the activities causing infeasibility. Activities or combinations of activities are located from the clash matrix such that they cannot be allocated to the same time-periods, and thus the cause of the infeasibility is determined. Therefore a new combination of resources in such activities must be sought. By conferring with the load and clash matrices the re-organization of the activities can be more easily accomplished. A detailed example is contained in chapter 8 to indicate such situations.

The clash matrix presents a clear indication of the extent and composition of the school activities. It indicates activity pairs that are compatible in that they may be allocated to common time-periods. The matrix is therefore of considerable assistance in the construction of new combinations of activity resources when an infeasibility is attributed to a faulty grouping of resources. Both the clash matrix and resource load matrix are of considerable benefit in the error detection and correction techniques for practical problems and a direct application is presented in chapter 8. The connection between the two matrices has been demonstrated in examples 6.5 and 6.6.

6.7 THE ASSIGNMENT ALGORITHM

This algorithm is the final stage of the assignment procedure. The algorithm simply stores the solution rows determined by the generated mappings and stores relevant computer information required to reinstate previous stages together with the reduced CAA's.

A flow chart combining all algorithms is given in the next chapter when the computer program and results are discussed.

CHAPTER 7

THE COMPUTER PROGRAM AND GENERAL TIMETABLE RESULTS

7.1 INTRODUCTION

The mathematical model of the timetable problem has been presented in chapter 4, and a formal description of the solution algorithms is contained in chapters 5 and 6. The establishment of the solution method in the form of a computer program, and its application to various school timetable problems is discussed in this chapter.

The speed of the logical operations of the computer are utilized in the investigation of the implications of an assignment (see chapter 6, section 6.2). The advantage that this approach has over previous methods is in the implication and assignment techniques. From chapter 4, section 4.2, an assignment involves all daily activities of a class. Thus for a p time-period school day, an assignment is the allocation of p activities of a class to the p time-periods. The implication algorithm considers the effects of the assignment on other unassigned classes. In previous methods such as those quoted in chapter 1, an assignment involved only one teacher-class activity. Thus the inter-relationships between activities and their subsequent implications on the timetable solution were not quickly established when generating the solution. In this method a class is treated as an assignment unit, and all relationships for a class assignment are considered together with individual activity implications involved in an assignment. Thus the solution method firstly detects any infeasibilities in a problem more quickly, and secondly is directed toward a solution, when one exists, without a large amount of time being wasted on undetected assignment difficulties.

This approach, through the recursive nature of the back-track algorithm of chapter 6, has the added advantage of being exhaustive. The program is capable of producing every solution to a given time-table problem by rejecting the last assignment of each successive solution produced, and thereby forcing the program to back-track and try again.

The program will be used to solve many school timetable problems in South Australia, and indeed has already been used with success at Craigmore High School (chapter 8). Therefore it was important that running costs of the program should be kept within acceptable economic bounds. In the final outcome, results that have exceeded expectations have been achieved without excessive expense. A discussion of the method of solution is given. Input data for the program are detailed and binary word patterns, which are used extensively, are discussed. An important feature of the program is the packing of data within words in the computer primary storage of the Control Data 6400 machine.

7.2 INPUT DATA

For convenience in data storage and manipulation every school resource is given a distinct positive integer code (non-zero). Since classes already have such a code within the schools (see section 3.2, chapter 3) these remain unchanged, but teachers, rooms and equipment must be considered. An example of teacher codes for the Craigmore High School Timetable problem is given in Appendix B, table B.1.

Every activity is presented to the program in the following form. Thus for the i-th activity the data string is

$$(\beta_1^i, \beta_2^i, \ldots, \beta_x^i, \overline{m}^i, \overline{b}^i, \overline{f}^i)$$

where

$\beta_j^i \in E$ is a resource of the school and is a member of the total resource set E of the school for each j = 1, 2, ... , x. for the i-th activity.

The variable notation x is used since the number of resources required by each activity need not be constant. The lower bound on x is 2, since an activity must involve at least one teacher and one class resource, (chapter 3, section 3.2), and the upper bound is α, the total number of resources in the school.

i.e., $2 \leqslant x \leqslant \alpha$ for all activities of the timetable problem.

$\overline{m}^i$ is the number of times the activity is required in the timetable solution. In a graphical representation of activity paths (see chapter 4, section 4.2) for the school timetable problem, $\overline{m}^i$ would represent the number of links in the path requiring the resources $\beta_1^i, \beta_2^i, \ldots, \beta_x^i$.

$\overline{b^i}$ is the block-period size indicator for activity i. If $\overline{m^i}$ is greater than one, the activities may either be required to occur in consecutive time-periods (a block-period) or as single time-period activities, separated by other activities in the timetable solution. The indicator $\overline{b^i}$ defines the number of consecutive time-periods required.

$\overline{f^i}$ is the fixed time-period indicator. If $\overline{f^i} = q$ then the activity i must be assigned to time-period q in the solution. If $q = 0$ the activity i may be assigned to any time-period.

The assumptions and contraints that relate to the problems to be solved are now repeated briefly for the convenience of the reader.

1. Every activity is represented in the resource requirement array in the form of a resource vector (see chapter 4, section 4.3). Hence every resource vector will contain the resources listed in the associated data-string for an activity given by the input activity data.

2. Each row of the resource requirement array R is associated with a particular class resource, and each element of the row is a resource vector, listing all resources required for a class activity. Thus for each class resource activity i, there will be $\overline{m^i}$ resource vectors of the associated class row of R with the same resource elements.

EXAMPLE 7.1

Consider an input activity data-string to comprise the following :-

(301, 302, 10, 11 ; 2, 2, 0)

which indicates that resources 301, 302, 10, 11 are required twice in a block-period size 2 and the activities are not fixed to any particular time-period. The classes are 301 and 302.

Let class 301 be associated with row R_1 of R. Then row R_1 will have two resource vectors of the form (302, 10, 11) (recall that the class resource associated with the row is omitted from the resource vectors of that row).

3. For a p period day, any resource may be required in at most p activities. Hence for any resource $\beta \varepsilon E$.

$$\sum_{\substack{\beta \varepsilon \text{ activity } i \\ i = 1, 2, \ldots}} \overline{m}^i \leq p$$

Indeed for the class resources, this inequality becomes an equality since every class must be occupied for every time-period of a daily time-span. (chapter 3, section 3.2).

4. The block-period indicator of the data-string defines only one block-period of size ≥ 2. If $\overline{m}^i > \overline{b}^i$ then all activities, $(\overline{m}^i - \overline{b}^i)$ in number, are assumed to have a block-size of one. A practical limitation on block-sizes within schools is that :-

$1 \leqslant \overline{b}^i \leqslant 5$ (section 3.2, chapter 3).

and for the data-string :-

$\overline{m}^i \geqslant \overline{b}^i$

5. The fixed period indicator must be within the range of the daily time-span and hence :-

$0 \leqslant \overline{f}^i \leqslant p$ (section 3.2, chapter 3)

6. In order to reduce ambiguity in the details of fixed time-period requirements, a condition was included such that whenever $\overline{f}^i > 0$ then $\overline{b}^i = \overline{m}^i = 1$ within the activity data-string. Thus every fixed time-period activity had to be separately detailed for the input data.

The activity data-strings are interpreted by the computer and a verification routine checks that resources are not over committed (see resource load matrix, chapter 6, section 6.5), that fixed time-period requirements can be allocated without causing infeasibility, and that the block-period requirements satisfy the constraints listed above. From each activity data-string three inter-related arrays are formed. These are :-

(a) resource requirement array which defines all resources required for each activity of the timetable solution

(b) block-period array which indicates which of the resource vectors of the resource requirement array are required in block-periods

(c) fixed time-periods array which indicates any resource vector of the resource requirement array that is required to be allocated to a specific time-period.

All details of these 3 arrays may be obtained by referring to chapter 4. To demonstrate the manner in which these arrays are formed a brief example will be given.

EXAMPLE 7.2

Consider the following data-string :-

(101, 102, 11, 12 ; 2 ; 2 ; 0)

where 101, 102, 11, 12 are the resources required for the activities

3 = number of activities = number of time-periods required since each activity has a duration of 1 time-period

2 = block-period size

0 = no fixed time-period required.

The codes 101, 102 are class codes (section 3.2, chapter 3) and will be associated with rows R_1 and R_2 respectively of the resource requirement array. Since each activity has an associated resource vector in R for the classes involved, then :-

Row R_1 will have 3 resource vectors of resources (102, 11, 12), and

Row R_2 will have 3 resource vectors of resources (101, 11, 12),

the class codes 101, 102 being omitted since they are implied for all resource vectors of rows R_1 and R_2 respectively.

Hence the resource vectors

$$r_{ij_1} = r_{ij_2} = r_{ij_3} = (102, 11, 12) \quad \text{for } R_1$$

and

$$r_{2j_1} = r_{2j_2} = r_{2j_3} = (101, 11, 12) \quad \text{for } R_2$$

The block-period indicator must be associated with these activities and since $\overline{b}^i < \overline{m}^i$ there are $(\overline{m}^i - \overline{b}^i)$ activities of block-size 1. Hence there are

2 activities in a block-size 2, and

1 activity in a block-size 1.

Thus the block-period array associated with R (section 4.2, chapter 4) becomes :-

$$B = \begin{bmatrix} 2 & 2 & 1 & \cdots\cdots \\ 2 & 2 & 1 & \cdots\cdots \\ \cdots & \cdots & \cdots & \cdots\cdots \\ \cdots & \cdots & \cdots & \cdots\cdots \end{bmatrix}$$

Similarly the fixed-period array associated with R becomes (section 4.2, chapter 4) :-

$$F = \begin{bmatrix} 0 & 0 & 0 & \cdots\cdots \\ 0 & 0 & 0 & \cdots\cdots \\ \cdots & \cdots & \cdots & \cdots\cdots \\ \cdots & \cdots & \cdots & \cdots\cdots \end{bmatrix}$$

Initially, it is assumed that every resource is available for every time-period of the daily time-span. Hence the resource availability array A has all elements set to 1 to indicate the complete availability of resources (see chapter 3, section 3.2). Some resources are not available for every time-period, e.g. part-time staff, and the availability array must be modified in the following manner.

The resource data-string indicating periods that are unavailable is presented to the computer program in the following form :-

$$(\beta_{i'} \; ; \; j_1, \; j_2, \; \ldots)$$

where $\beta_{i'}$ is the resource code, and j_1, j_2, ... are the time-periods that are not available for resource $\beta_{i'}$.

Thus the column vector associated with the resource $\beta_{i'}$ must be modified in the resource availability array such that row elements j_1, j_2, ... are reduced to zero.

As mentioned in chapter 4, section 4.2, block-periods have defined start-periods τ_b where b is the block-size and τ_b is a mapping of time-periods 1, 2, ... p onto the binary numbers 0, 1 indicating permitted start periods if a period is mapped onto a 1, and not permitted otherwise.

EXAMPLE 7.3

The block-period mapping

$$\tau_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

indicates start-periods 1, 2, 4, 6, 7 are legal for a block-size 2.

The details of these start-periods for the various block-sizes of theprogram are presented in the form of a block-period data-string.

$$(b \; ; \; j_1', \; j_2', \; ....)$$

where b = block-period size, b = 1, 2, 3, 4, 5.

j_1', j_2', ... are the admitted start-periods for block size b.

Hence the mappings may be constructed with a time-period j_1', j_2', ... mapped onto 1 and the remaining time-periods are mapped onto 0.

From chapter 4, section 4.5, the images of the mappings are stored in an array BS, with rows representing block-sizes and columns the p time-periods.

EXAMPLE 7.4

The block-period data-string

(2 ; 1, 2, 4, 6, 7)

results in the mapping

$$\tau_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

for an 8-period daily time-span, and will be stored in the 2nd col of BS, the block-period start array.

All details have now been presented to the computer program, and verified to indicate any obvious infeasibility such as over-committed resources, undefined block-start-periods, block-periods out of the range 1 to 5, etc. The computer program will now be discussed in relation to the solution of problems.

7.3 THE COMPUTER PROGRAM

The computer program was written in the FORTRAN IV and COMPASS programming languages for the Control Data 6400 machine. The primary storage words within this machine are unusually large (60 bits) and this feature was exploited in the program for the compact retention of the various resource requirement and availability arrays. Various working areas were also needed to permit a return to any previously defined stage of the solution. The large word size was combined with the high speed logical operations of .AND., .OR., .NOT. to further increase the speed of the solution method.

The main objectives of the computer program may be stated as follows :-

(a) to provide a solution to a timetable problem if one exists, or to indicate "no solution" when such a problem is encountered

(b) to translate the algorithms of the solution method into a form that is understood by the computer

(c) to do (b) efficiently, within the terms of reference discussed in chapter 4, e.g. cost

(d) to provide suitable diagnostic facilities for problems with a "no solution" result, so that a minimum of time is required to correct faults.

7.3.1 Storage and Operations of Data

The method of storage of the arrays will be discussed. All arrays, with the exception of the block-period size indicator array, are stored as bit patterns within the computer words. To illustrate this method of storage, the resource requirement array (section 3.2, chapter 3) will be discussed.

As detailed in section 7.2 of this chapter, all resources of a school are given a coded number between 1 and 60. If more than 60 resources are required at a school this coded number range can be increased to a programmed limit of 240 in multiples of 60. Hence, if a school has 60 or less resources, then one word of storage will be sufficient to represent them, if 120 or less, then two words will be required, etc. Each resource is associated with a particular bit position in a computer word.

EXAMPLE 7.4

Storage of data within primary computer words.

Consider the resources :-

		Code = position in computer word
Classes	301	1
	201	2
	101	3
Teachers	Jones	4
	Smith	5
	Brown	6
Other resources	T.V.	7
	Room 1	8
	Room 2	9
	Room 3	10

Then a word to represent a requirement of the resources 301, Smith, T.V., Room 1 would be :-

(............... 8 7 . 5 ..... 1) position

(0 0 0 .... 0 1 1 0 1 0 0 0 1) primary computer word

with 1-bits in positions 1, 5, 7, 8 of a single computer word, to represent the resource codes.

7.3.2 Resource Vectors and Composite Availability Vector Operations

Resource vectors describe the resources required for the activities of the timetable problem, and are stored in the resource requirement array R (chapter 4, section 4.3). In the computer program, they are stored in the manner just described. Each resource vector is a list of required resources, and this list is represented by a 0-1 bit pattern. For a school of 60 at less resources there will be 1 word per resource vector, for a school with 61 to 120 resources, 2 words per resource vector, etc. A large school could have as many as 240 individual resources. Hence the size of the array R in the computer program where there are m class resources and p time-periods per school day is within the range

$m \times p$ to a maximum $m \times 4p$.

Associated with each resource vector, r_{ij}, is a composite availability vector $A^*(r_{ij})$, that indicates the availability of the resources required for the activity for each time-period of the daily time-span (chapter 5, section 5.2). It was previously shown that this information can be stored in a 0-1 array, and this is the case within the program for each CAV. Hence the link as follows :-

Resource Vector	Associated composite availability vector	
0 0 0 ... 1 0 0	0 0 ... 0 1 1 0 0 0	
0 0 0 ... 0 1 0	0 0 ... 0 1 1 0 0 1	(7.1)
0 0 0 ... 1 1 0	0 0 ... 0 1 1 0 0 1	

that shows that resource 3 in the first row is available for time-periods 4, 5 as indicated by the CAV.

Through simple logical .AND. operations it is possible to determine common available time-periods for any group of resource vectors.

e.g. the common available time-periods for rows 1 and 2 above are given by :-

(0 0 ... 0 1 1 0 0 0) .AND. (0 0 ... 0 1 1 0 0 1)

to give

(0 0 ... 0 1 1 0 0 0)

indicating time-periods 4, 5 are the only common time-periods available to both resource vectors.

To determine 'tight situations' or critical blocks as discussed in section 5.3, chapter 5 within the CAV, the logical .OR. operation is used as follows :-

Consider the three resource vectors of 7.1. By performing a logical .OR. operation on the three associated composite availability vectors

(0 0 ... 0 1 1 0 0 0) .OR. (0 0 ... 0 1 1 0 0 1) .OR.

(0 0 ... 0 1 1 0 0 1)

we get

(0 0 ... 0 1 1 0 0 1)

Hence three resource vector requirements together have only three available time-periods, and these time-periods must be used by these activities. Thus a tight situation has been located and the appropriate actions of reserving these periods can be applied.

e.g. consider a fourth CAV

(0 0 ... 0 1 1 1 1 0)

Then by a simple logical operation

(0 0 ... 0 1 1 1 1 0) .AND. .NOT. (0 0 ... 0 1 1 0 0 1)

= (0 0 ... 0 1 1 1 1 0) .AND. (1 1 ... 1 0 0 1 1 0)

= (0 0 ... 0 0 0 1 1 0)

Thus time-periods 2 and 3 are the only remaining available time-periods from the original 2, 3, 4, 5 time-periods since the periods 4, 5 are reserved for the other tight activities. This example demonstrates the mechanism of the location and subsequent reduction of availabilities associated with tight situations.

It can be seen from the above examples that the 0-1 patterns and logical operations have important applications in computer methods on timetable problems. e.g.Barraclough (3) has indicated the use of bit patterns and logical operations for timetable problems. Storage of the large amount of data required for timetable problems has been overcome by the method of compacting detail into word patterns. The methods just described have the advantage of reducing array sizes without loss of computational speed.

7.3.3 The Main Features of the Computer Program

The main features of the computer program are shown in the flow chart of Figure 7.1, and are as follows.

The first stage of the program involves the establishment and verification of data arrays. Any of the obvious inconsistencies as discussed in section 7.2, are detected immediately and a diagnostic printed. After all data have been considered, the program either continues to the next stage, or if errors have been located stops to allow corrections to be made manually. There are 2 stop conditions, stop 01, stop 02 in the flow chart. The first indicates errors in data, the second refers to incompatabilities between fixed time-period requirements and availability conditions of resources required in the activities.

The preparation of the arrays includes the construction of resource requirement arrays, resource availability arrays, composite availability arrays, block-size arrays, and the general work and storage arrays necessary for the back-track procedure discussed in chapter 6, section 6.3. As detailed in chapter 5, section 5.2, all binary reductions are involved with the composite availability vectors that are determined

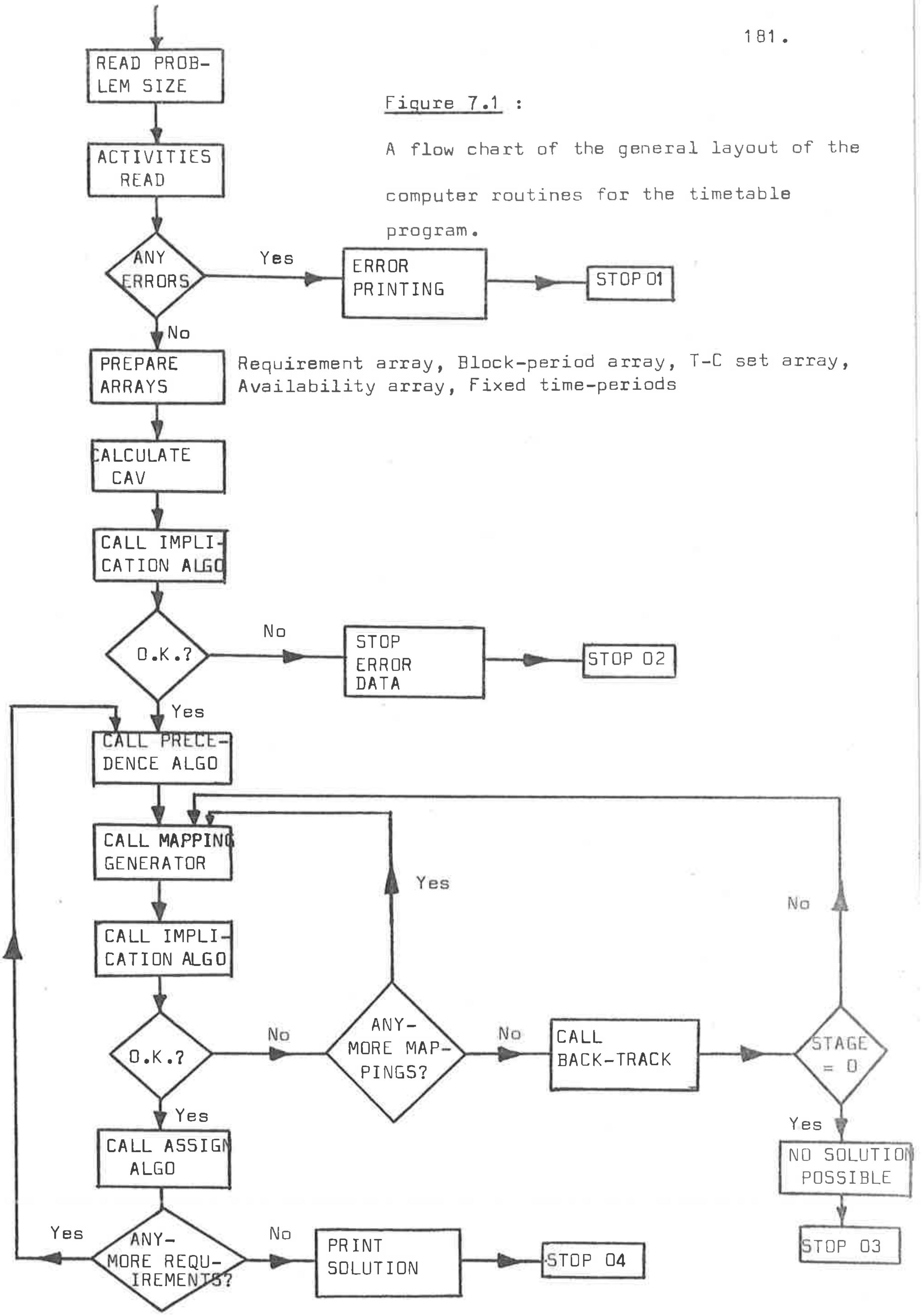


Figure 7.1 :

A flow chart of the general layout of the computer routines for the timetable program.

from the availability vectors of individual resources.

Any errors detected up to this stage of the program must be manually corrected. Errors are located by the program.

e.g. resource 12 is required for 8 time-periods but there are only 7 time-periods in the school day.

This error indicates that resource 12 is over-loaded. Hence all activities requiring resource 12 must be located and a suitable re-allocation of resources made. Once all errors have been corrected the program is restarted. The next stage is the assignment stage.

Figure 7.1 details the order of the algorithms as they occur within the computer program. A description of each algorithm has been given in chapters 5 and 6. The precedence algorithm determines the next class requirements (or row requirements since the two terms class and row are synonymous) to be attempted by the bijection generator. The levels of precedence have been discussed and the program locates the unassigned class with the highest precedence number.

Then the bijection algorithm generates a feasible mapping for the assignment of that class. It has previously been stated that although the mapping generated may be feasible for the class to be assigned, it may not be feasible when considered with respect to other unassigned class requirements. The implication algorithm considers many of the effects that the assignment would have on the unassigned class requirements. If no infeasibilities are fore-seen the mapping is accepted and the assignment made. Then the precedence algorithm determines the next assignable class. If an infeasibility is found by the implication algorithm, a new mapping must be generated for the class requirements.

When some difficulty arises, due to an unforeseen infeasibility, the back-track algorithm can be called to reinstate any previous assignment situation so that other solution paths may be considered. The back-track algorithm has been discussed extensively in chapter 6, section 6.3. It has also been shown that the method is exhaustive since every assignment can be generated for each class requirement of the timetable problem. If the program is forced to retrace to the first class requirements considered, and no alternative assignment can be generated for this class then no solution to the problem can exist. In such a situation a subset of class requirements has been found such that no suitable assignment can be made without violating the constraints of the problem, and thus no solution to the problem exists. This aspect has been discussed previously by Cisma (9). The maximum time so far encountered to locate a "no solution" result is approximately 7 minutes computer time, for a problem involving 40 teachers, 25 classes and a 7 time-period day. At present there does not appear to be any means whereby an exact estimate of time to locate these "no solution" results can be determined.

7.4 THEORETICAL AND PRACTICAL RESULTS

7.4.1 Development of Method of Solution

The first bijection generator used for the solution method presented in this thesis was a permutation routine, described as the Johnson-Trotter algorithm in Welsh (56). The algorithm is based on a translation technique for producing successive permutations by the interchange of two adjacent elements within the preceeding permutation.

e.g. by interchanging the numbers 1 and 3 in the permutation 1, 3, 2 the new permutation 3, 1, 2 is generated.

The method has been shown to be an efficient permutation generator, e.g. see reference (41). In the present application, each permutation produced was used in the bijection generator and tested for feasibility for the class in the school to be assigned. Successive permutations were generated until a feasible mapping for the class requirements was identified. Then the same procedure would be repeated for the next class requirements, and so on, until the timetable was completed.

As suggested by Appleby et al. (2), the computation times entailed in this approach were excessive, even with the use of a high speed computer. A table of computation times is presented in method 1 of Table 7.1. The long execution times were caused by the extensive testing of infeasible mappings associated with this method. To reduce these execution times, a subroutine was included in the program to locate requirements with only one available assignment position, and to ensure that the mapping produced would not involve any attempt to assign these requirements elsewhere. This modification resulted in a substantial reduction in execution times, as shown in method 2 of Table 7.1.

Number of Teachers	Number of Classes	Number of time-periods in a school day	Method 1 Based on Johnson-Trotter algorithm	Method 2 Modified Johnson-Trotter algorithm	Method 3 Bijection Generator with Implication algorithm
3	3	3	.57	.45	.99
4	4	4	.49	.52	1.05
5	5	5	.56	.64	1.28
6	6	6	.96	.89	1.55
7	7	7	3.85	2.54	1.93
8	8	8	31.98	15.44	2.26
9	9	9	333.77	132.95	3.05

Table 7.1 : A comparison of execution times in CP seconds, to solve the various simple tight timetable problems from 3 x 3 x 3 to 9 x 9 x 9 for the 3 methods indicated.

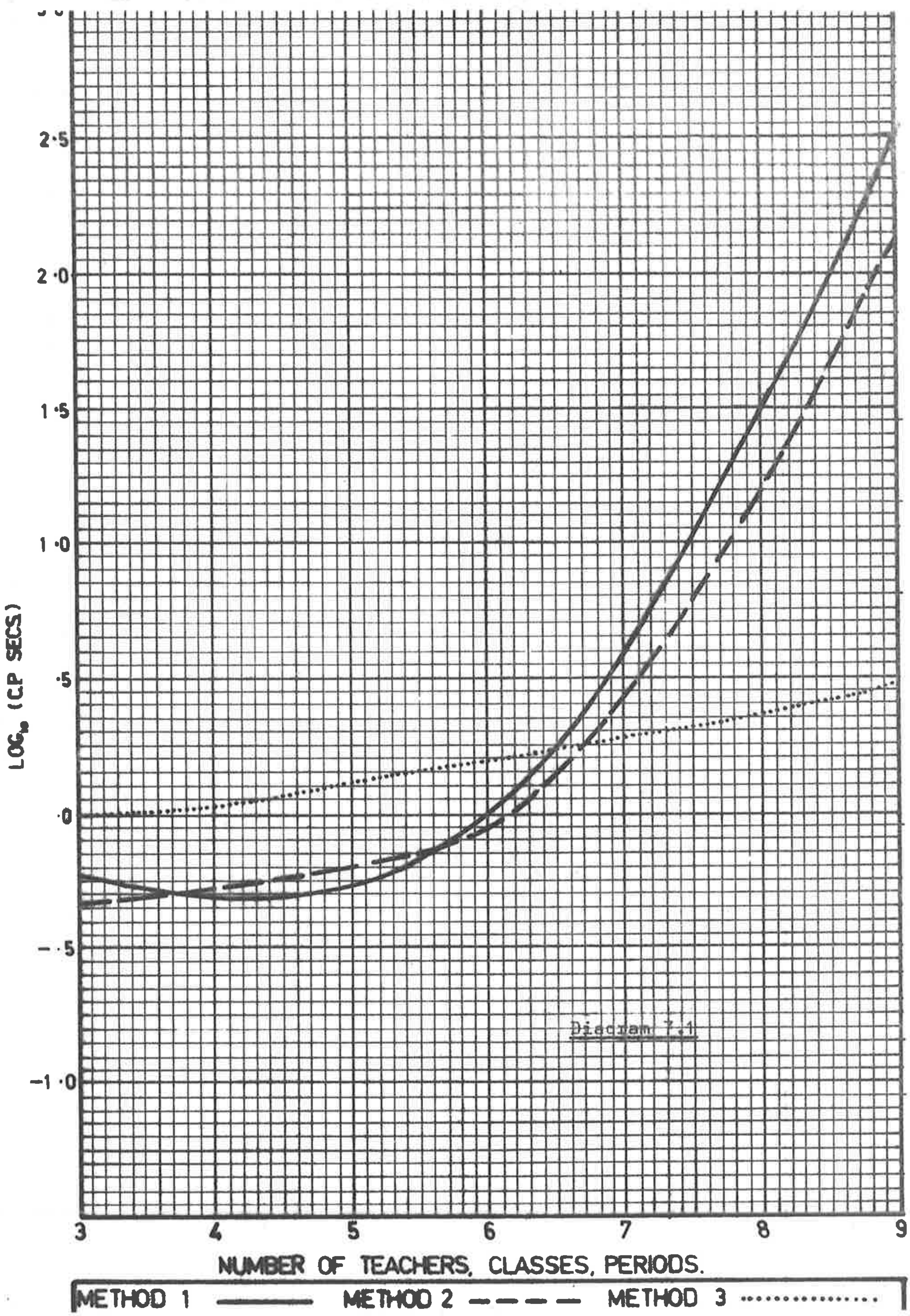
The simple tight timetable problems have been discussed in chapter 4, section 4.3, where each class must meet with the same teachers during the school-day. Hence each resource requirement row of the resource requirement array contains the same resource vectors, and the mapping generator described above, must investigate more and more unfeasible mappings as the method proceeds. This situation arises because

(a) the mapping generator based on the Johnson-Trotter algorithm produces the same mappings in the same order for each class assignment, and

(b) as the number of rows assigned increases, the number of feasible mappings remaining decreases.

e.g. by enumerating all permutations for a 4 × 4 × 4 simple tight timetable problem it can be shown that there are 24 = 4! feasible mappings available for the first assignment. However, after one of these has been accepted there remains only 2 feasible mappings of the original 24 feasible mappings for the second assignment.

This led to the third method of generating mappings. This method only generates the feasible mappings for any row of the resource requirement array. The rejection of unfeasible mappings was accomplished through the use of the composite availability vectors, that indicate the remaining available



3·0
2·5
2·0
1·5
1·0
·5
·0
−·5
−1·0
LOG10 (C.P SECS)
3
4
5
6
7
8
9
NUMBER OF TEACHERS, CLASSES, PERIODS.
METHOD 1
METHOD 2
METHOD 3

Diagram 7.1

positions (time-periods) for each unassigred resource vector. (Chapter 5, section 5.2, 5.3). The essential features that give the increased efficiency are first, the reduction in execution time due to the elimination of tests of feasibility for mappings associated with the class requirements. Second, the reduction of the composite availability vectors that eliminates infeasible mappings of the unassigned class requirements. Third, the application of the implication algorithm to look-ahead at future requirements to ensure that such requirements have not become infeasible because of the generated mapping at each assignment stage.

The computer execution times for this third method are given in Table 7.1. The presented graphically in diagram 7.1.

The solution method discussed in this thesis (method 3), has been extensively tested on both theoretical (simple, and simple tight timetable problems) and practical problems with a considerable reduction in execution times for the generation of solutions. Computation times to study the effects of the various practical complexities required by schools are tabulated and discussed below. In each case the results quoted are for the simple timetable problem of 9 teachers, 9 classes and 9 time-periods since :-

(a) this simple problem has no flexibility as all resources are fully utilized, and is therefore more difficult to solve, and

(b) the 9 x 9 x 9 problem gives a maximum solution time since it is the largest of the simple tight problems tested, and also is the maximum number of time-periods (9), occuring in schools within South Australia.

The application of this program to an existing school timetable problem at Craigmore High School will be described in detail in chapter 8. The solution produced is now in use at that school, and future applications are discussed in chapter 9.

7.4.2 Fixed Time-Period Requirements

As defined in chapter 3, section 3.2, a fixed time-period requirement forces an allocation of a particular activity to a specified time-period. Such a requirement increases the number of constraints on the timetable problem since such an allocation reduces the availability of the resources involved in the fixed requirements, i.e. the resources involved are no longer available for assignment to this time-period elsewhere in the timetable solution.

The program was tested on the 9 x 9 x 9 problem with increasing numbers of fixed requirements. Results are contained in Table 7.2 for 1 to 10 fixtures.

Number of Fixed Time-Period Requirements.	0	1	2	3	4	5	6	7	8	9	10
Solution Time	3.0	3.1	3.1	3.0	3.0	3.0	3.0	3.0	3.0	2.9	3.0

Table 7.2 : Solution time results for fixed time-period requirements 1 to 10 in the 9 x 9 x 9 simple tight timetable problem

It was evident from the solution times that the fixed time-period requirements had little effect on the speed of the solution method. This is understandable since the consequence of a fixed time-period requirement is that the associated composite availability vector is reduced, such that only the required time-period remains available for that activity. (see chapters 5 and 6 for a more detailed explanation of the composite availability vectors and the effect of the fixed time-period requirement). This stage is indicated in Figure 7.1 of the computer flow-chart when the arrays and the CAV are calculated. The effect of the fixed requirements on other CAV is established through the implication algorithm, discussed in chapter 6, section 6.2.

Upon extending the number of fixed requirements to 30 no change was noted and result times were still between 2.9 and 3.1 seconds CP.

7.4.3 Block-Period Requirements

Block-period requirements involve allocating activities to consecutive time-periods within the timetable solution. The number of time-periods involved is the block-period size (see chapter 3, section 3.2). The most common practical block-sizes of 2 and 3 were extensively tested. Table 7.3 contains examples of solution times for six problems of block-period size 2, and six problems of block-period size 3.

No. Block Size 2	0	1	2	3	4	5	6
Solution Time	3.1	3.1	2.9	3.0	3.0	2.9	2.8

No. Block Size 3	0	1	2	3	4	5	6
Solution Time	3.1	3.0	3.1	2.9	3.4	6.3	3.8

Table 7.3 : Solution times for 9 x 9 x 9 problem with block-period size of 2 and 3.

The block-size 2 results remained relatively stable. This indicated that these requirements, being the most prevalent block-periods in practical problems, were marginally more difficult than single period requirements with respect to execution times. The block-size 3 requirements indicated similar tendencies, having a slight increase in execution time when compared to the block-size 2 results. An increase in computation time was noted for the problem involving 5 block-size 3 requirements and was associated with several back-tracks that were needed to produce a solution to this problem. Other problems were solved within increasing numbers of block-size 2 and block-size 3 requirements. The maximum execution time was still associated with the 5 block-size 3 problem. However, there may exist problems thar do have increased computation times, associated with back-tracking to produce a solution. Nevertheless, the execution times presented are economically acceptable, and are considerably less than expectations.

7.4.4 Teacher-Class Set Requirements

As detailed in section 3.2, chapter 3 the teacher-class sets involve several teacher and class resources for the one activity. Once again solution times were relatively stable implying that resource distribution was the main cause of increased solution times for this method of solution. The implication algorithm was sufficiently flexible to direct the problems to speedy solutions on each occassion that the timetable problem had a result. The loading of resources and problems with no solutions are discussed in chapter 8.

7.4.5 General Problems

Many problems were tested that were compiled from existing practical timetable problems. An example is given in appendix

A with a solution. A detailed discussion of the solution technique will be given in chapter 8. The computer program on all occassions produced solutions to problems when they existed, and indicated "no solution" results when such situations arose. In the no solution problems, the infeasibilities were manually corrected and results produced. An example of this is given in section 8.4, chapter 8. The practical problem of Craigmore High School will be discussed in the next chapter.

CHAPTER 8

THE SOLUTION OF THE CRAIGMORE HIGH SCHOOL TIMETABLE PROBLEM

8.1 INTRODUCTION

The solution method described in this thesis was tested by solving the Craigmore High School timetable problem, selected by the Education Department of South Australia. The problem contained the following special features :-

(a) the school was a comprehensive type (chapter 3, section 3.2) which was technically suitable since the required timetable involved complexities associated with both High and Technical High schools.

(b) the school was to have staff changes midway through the second term of the school year. These changes would significantly disrupt the previous timetable and a complete new solution would therefore be required.

(c) the new result was required quickly, to avoid extra administration burdens on both students and staff at the school.

Staff changes during the school year are not unique and occur for a variety of reasons, e.g. resignations. Replacement teachers are not often qualified in the disciplines of the existing teachers, thus necessitating a re-allocation of staff duties. Hence a new timetable must be constructed. The computer method has a direct application to such intra-year problems as well as the new year problems that arise at the start of each academic year.

The description of the Craigmore High school problem and its solution will now be given. Examples are included to illustrate different aspects of the problem. The problem associated with Tuesday's timetable is discussed in detail. All data associated with the Craigmore description is contained in appendix B. Solutions to the five daily problems are tabled - appendix C.

A problem with no solution is presented in section 8.4 and the relevant data given in appendix D.

8.2 DEFINITION OF THE CRAIGMORE HIGH SCHOOL TIMETABLE PROBLEM

8.2.1 General Discussion

The Craigmore problem involves some 410 students and 23 staff members consisting of a headmaster, deputy headmaster, 3 senior masters, 1 senior mistress and 17 teachers. One of the teachers is only available for the first 3 time-periods of any one school day (a part-time teacher). It will be seen later, that this teacher is fully utilized in every available time-period.

The students were assigned by school administrators to 13 classes that have the following numeric codes (described in chapter 3, section 3.2). For convenience these codes will be adopted for the remaining discussion in this chapter

101, 102, 103, 104, 105 ;
1st year level

201, 202, 203, 211 ;
2nd year level

301, 302, 303, 311
3rd year level

At present the school is at two thirds capacity with respect to student enrolments since it is a new school in a recent suburban area. Administrators expect enrolments to be at the capacity of approximately 610 in January, 1973. Classes were constructed from the previous academic achievement and I.Q. of each student together with personal interviews to determine the future course requirements of the student.

During 1972 the 4th and 5th year levels were not available at Craigmore. However extensive teacher-class sets occurred in the 2nd and 3rd year levels as will be seen later in this chapter. This gave rise to a very complex timetable problem that was time-consuming and difficult when solved by manual methods (see section 8.3). The problem contained a high percentage of features of both the Technical and High school timetables and was hence a demanding problem for the computer solution method.

The school facilities consists of 18 classrooms, but several are for specific purposes, e.g. typing room, history and geography model room, remedial teaching room, etc. These are used by specified classes for required time-periods within the timetable solution. On the whole however, the class-room situation did not cause any restrictions on the timetable solution procedure. Sufficient rooms were always available to satisfy all class requirements.

8.2.2 Resource Requirement Array Construction

For the purpose of this thesis the manual compilation of timetable data for the computer method has not been detailed. The teacher resource codes, teaching subjects and teacher status have been detailed in table B.1 of Appendix B. Class resource codes will be left unchanged in the text to avoid confusion. Thus code 301 will still be associated with a 3rd year class, being the first class in the 'O-track' (chapter 3, section 3.2). However, it should be remembered that classes are similarly coded as are the teachers in table B.1, such that no two resource of the school have the same code number. School administrators compiled the resource activity requirements for each class of the school on a daily basis. The details are presented in tables B.2 to B.6 in the form of activity requirements, that indicate the resources required, number of lessons involved, the block-period size and whether the lesson is to be allocated to a specific time-period in the school day. The details of this presentation has been given in section 7.2 of chapter 7. These requirements partially describe the five sub-problems of the weekly timetable. The other features such as block-period definitions, will be discussed later. Examples are given to illustrate various aspects for ease of understanding.

The resources required for each activity are placed in the resource requirement array by the computer program (see section 4.2, chapter 4).

<u>EXAMPLE 8.1</u>

Consider the resource data of table B.3 for Tuesday in Appendix B. Resource requirement vectors for classes 101, 102, ... , 311 are presented in the rows of that table, and are placed into the 13 rows of the resource requirement array in the following manner.

Consider class 101 to be placed in row one of R, namely R_1.

Then R_1 = ((15,18), (15,18), (6), (103, 16, 22), (12), (10), (3), (16)) Similarly R_2 is associated with class 102, R_3 with 103, etc. until R_{13} associated with 311.

8.2.3 Block-Period and Fixed Time-Periods

The block-period starts that are required for the Craigmore problem are defined in the following manner (see chapter 4, section 4.5).

$$\tau_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\tau_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Only block-period sizes 1 and 2 are required within this timetable problem since the administrators are of the opinion that two time-periods of 40 minutes are sufficient for craft practical periods. The block-periods starts are stored by the computer on the form in the array BS

$$BS = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}$$

where column 1 is related to block-periods size 1 and column 2 to block-period size 2. There are numerous block-period requirements indicated by tables B.2 to B.6 of Appendix B. For example table B.3 for Tuesday has a block-period size 2 for each classes 101, 102, 104, 105 as indicated by the column labelled <u>block</u>.

Fixed time-period requirements are also indicated in table B.3 for classes 301, 302, 303, 311 in activities (69,70) and (71,72) for time-periods 3 and 4. This is a complex fixed time-period requirement since it involves nine resources. A fixed time-period involving every class is given in table B.5 Appendix B, where Religious Instruction is given to every student in the 8th lesson on Thursday, activity (9,10). The persons taking these lessons are external to the staff of the school and are indicated as part-time teachers with teacher codes 24 and 25. The fixed time-period array F as described in chapter 4, section 4.5 is associated with the daily resource requirement array.

<u>EXAMPLE 8.2</u>

For Tuesday, (table B.3, Appendix B) the fixed time-periods must be indicated for classes 301, 302, 303, 311 thus involving rows 10, 11, 12 and 13 of F, the fixed time-period array.

The rows of the resource requirement array R are for example

R_{10} = ((302,303,311,2,3,9,21), (302,303,311,2,3,9,21), (302,303,311,2,3,4,17,23), (302,303,311,2,3, 4,17,23), (302,303,311,6,7,16,19), (302,303, 311,6,7,16,19), (302,303,311,6,12,20,21,22))

Similarly R_{11}, R_{12}, R_{13}.

Hence the 10th row of F will be

$$F_{10} = (0 \quad 0 \quad 0 \quad 3 \quad 4 \quad 0 \quad 0 \quad 0)$$

Similarly F_{11}, F_{12}, F_{13}.

For convenience, it was noted in chapter 4, section 4.5, that all fixed time-period requirements must be of multiplicity and block-period size 1. This is demonstrated in the two row requirements of table B.3 in activities (69,70) and (71,72) that involve exactly the same resources.

8.2.4 Resource Availability

Initially all resources are assumed available for every time-period. The resources that have reduced availabilities in this exercise are the part-time teachers, namely resource 8. The time-periods 4 to 8 are not available and hence all composite availability vectors involving this resource must be reduced to exclude these periods.

EXAMPLE 8.3

Consider table B.3, Appendix B where activities (53,54) and (67,68) involved resource 8 (the part-time teacher). These activities are contained in rows 6, 7, 8, ... , 13 of the resource requirement array since the resources involve classes 201, 202, ... , 311.

Consider the CAA (see chapter 5, section 5.2) for row R6 of R.

$$A_6^* = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

where periods 4 to 8 have been excluded due to resource 8.

8.2.5 Teacher-Class Sets and Timetable Structure

The structure of the second and third year levels indicates complete teacher-class setting for every time-period of the daily time span. This complexity was consistent for every day of the school week.

The 8 time-period pattern present at Craigmore is relatively common although latest trends favour the 7 period day. The daily activity pattern is :-

lessons 1 to 3 (each of 40 minutes duration)

Recess Break

lessons 4 and 5

Lunch Break

lessons 6 to 8.

No afternoon recess break was available at this school.

As mentioned in chapter 4, the course content is not considered by the assignment procedure. The activity is assigned to a time-period, and the function of the activity is not considered in detail. Therefore a printout procedure had to be written to relate the activity to a subject to make the solution useful for the schools. Also a decoding routine was included to convert the teacher codes to teacher names for output purposes.

Numbers of :	Monday	Tuesday	Wednesday	Thursday	Friday
Teachers	23	23	23	25**	23
Classes	13	13	13	13	13
Time-periods	8	8	8	8	8
Block-periods	3	4	3	6	5
Fixed-periods	0	2	3	13	0
Teacher-class sets	14	12	14	14	14
Back-tracks*	0	100	6	0	1
Solution Time (seconds)	15.2	102.3	16.7	15.0	15.1

Table 8.1

Comparisons of features of the 5 daily timetables of the Craigmore problem.

* the number of back-tracks the computer program went through to produce the daily solutions.

** the two extra staff were external to the school for religious instruction lessons.

The final timetable results have been presented in Appendix C, tables C.1 to C.5. The solution method and special features will now be discussed.

8.3 SOLUTION OF THE CRAIGMORE PROBLEM

8.3.1 General Discussion

A detailed comparison of the various important features of the 5 daily Craigmore sub-problems are given in table 8.1. Solution times are included and the number of back-tracks by the solution procedure presented. Solutions were determined for every day and are included in Appendix C.

Solution times include the 12 seconds (approx.) needed for the compilation of the timetable program. The Tuesday timetable was most difficult and the problem will be discussed to indicate the reasons for the difficulties. The solutions produced were readily acceptable to the Craigmore administrators and the solution was immediately incorporated into the school system. The staff member responsible for the manual production of their timetable in past years was enthusiastic at the speed of the solution. The Tuesday problem is now discussed in relation to the following sections.

8.3.2 Special Features

The description of the special features of the Tuesday timetable have been discussed in section 8.2. These include block-periods, fixed time-periods and resource requirements. The main problem area arises through the distribution of resources in the class activities. Many of the resources required in the complex teacher-class sets are involved with the 1st year level classes.

The clash matrix (chapter 6, section 6.6) is given in Appendix B, table B.8 and the resource load matrix in table B.7. From this matrix it is seen that the activity (75,76) of table B.3 clashes with 10 of the other teacher-class set requirements of Tuesday (indicated by the row sum by counting the number of zeros that occur). Recall that a zero entry in the clash matrix indicates that the resource vectors associate with the row and column of the clash matrix can not be allocated to the same time-period. Thus requirement of activity (75,76) may only be assigned in a common time-period with requirement of activity (61,62) since only activities (61,62) and (69,70) are shown by the clash matrix to be available for assignment with this set. ((69,70) may be neglected since it involves the same classes). We can see that the resources involved in such an assignment involves many of the 1st year level resource requirements, e.g. class 103, it clashes with 3 lessons.

Further difficulties arise due to the fixture of activities (69,70), (71,72) of table B.3 into time-periods 3 and 4. The clash matrix demonstrates that only activities (53,54), (55,56) and (57,58) may be assigned to the same time-periods for the 2nd year level classes. Resource 8, (see table B.9, Appendix B) the part-time teacher is involved in (53,54) and must be allocated in time-periods 1 to 3. That resource is also required in (67,68) and must be allocated in time-periods 1 or 2 since 3 already allocated. Hence, since activity (53,54) is required twice, indicated by the multiplicity column of (53,54) in table B.3, then (53,54) must be allocated in time-period 3 for at least one of the two required time-periods.

The remaining requirement of activity (53,54) and (67,68) may be such that they are either allocated to time-period 1 or 2.

Another complexity is the limited number of distinct teacher resources required by class 103. Three of the teacher resources are required twice in the day (once again indicated by the multiplicity of table B.3). Two of the required resources for class 103 are heavily involved in the extensive teacher-class sets of the 2nd and 3rd year levels, namely teacher resources 19 and 2.

The above is a brief description of the difficulties of the Tuesday timetable. However, the solution method although encountering some difficulty, solved the problem in some 90 secs., which was quite acceptable. This involved some 100 back-tracks to previous stages to avoid the no solution stages indicated by the implication algorithm, extensively treated in chapter 6, section 6.2. The solution is given in table C.2, of Appendix C.

8.3.3 The Precendences for Tuesday's Timetable

The classes 201, 202, 203, 211 are considered first since they involve the most complex allocations determined on the basis of priorities discussed in chapter 6, section 6.4. The factors involved in this priority are number of distinct teacher-class sets, block-periods, fixed time-periods, resource availabilities, etc. In the case of the 2nd year level classes there are, for example, 6 distinct teacher-class sets whilst the next nearest is the 3rd year level with 5. No block-periods are required by the 2nd year level activities.

When the 2nd year level classes have been allocated (all are allocated since each have the same priority and are treated in turn) the 3rd year level classes become the next highest. Hence each class 301, 302, 303, and 311 are allocated. Priorities are then calculated for the 1st year level classes and these are allocated according to the sequence 102, 104, 105, 101, 103 thus completing the problem. For example class 102 is allocated prior to 104 since 102 involves 2 distinct teacher-class sets and a block-period of size 2. (see activities (19,20), (21,22), (23,24) of table B.3, Appendix B).

The precedences are recalculated after each allocation stage to determine the next row for assignment.

8.3.4 Brief Description of the Allocations

It has been shown in chapter 5, section 5.2 that associated with each row of the resource requirement array R is the composite availability array. that indicates all time-periods available to the resources of each of the activities associated with the row of R.

EXAMPLE 8.3

Consider the classes 301, 302, 303, 311 that will be associated with rows 10, 11, 12 and 13 of R, from table B.3, Appendix B. The composite availability arrays for each of the rows R_{10}, R_{11}, R_{12}, R_{13} will be the same since they each involve the same resources (with the exception of class resources that are always available).

Hence

$$A^*_{10} = A^*_{11} = A^*_{12} = A^*_{13} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Similarly the CAA for rows 6 to 9 for 2nd year classes are calculated and

$$A^*_{6} = A^*_{7} = A^*_{8} = A^*_{9} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

where a zero in position (i, j) indicates the non-availability of the time-period i for the jth element of the associated row of R.

Before any precedences are calculated or allocations determined, the implications of the fixed-period and part-time features are investigated. This stage is accomplished by applying the implication algorithm to the CAA.

EXAMPLE 8.4

The implications of the fixed time-periods of activities (69,70), (71,72) of table 8.3 have the following reduction effect.

Teacher resources involved are 2, 3, 4, 17, 23.

All rows involving any one or more of these teachers will have their CAA reduced to omit time-periods 3 and 4 (make these time-periods unavailable).

Such a CAA is associated with 2nd year level classes where the new CAA are

$$A_6^* = A_7^* = A_8^* = A_9^* = \begin{bmatrix} 1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1 \\ 1&1&1&1&0&0&0&0 \\ 0&0&1&1&0&0&0&0 \\ 0&0&1&1&1&1&1&1 \\ 0&0&1&1&1&1&1&1 \\ 0&0&1&1&1&1&1&1 \\ 0&0&1&1&1&1&1&1 \end{bmatrix}$$

This reduction of the sets of resources is indicated by the clash matrix of table B.8 that indicates that activities (69,70), (71,72) clash with (59,60), (61,62), (63,64) of the 2nd year level classes.

Also the implications within the CAA of rows R_{10} to R_{13} of R that are directly effected by the fixed time-periods become

$$A_{10}^* = A_{11}^* = A_{12}^* = A_{13}^* = \begin{bmatrix} 1&1&1&1&0&0&1&1 \\ 1&1&1&1&0&0&1&1 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&0&0&0&1&1 \end{bmatrix}$$

Similar reductions are produced for the 1st year level CAA.

The precedence algorithm indicates 2nd year level classes are to be allocated first.

From the above reduced CAA a bijective mapping is generated to allocate the activities of rows R6 to R9 to time-periods as described in chapter 5, section 5.3.

The mapping

$$\Delta_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$$

is not feasible since this would reduce the CAA of A^*_{10} to A^*_{13} to zero in the 4th column (involving resource 8 the part-time teacher).

$$\Delta_6 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 3 & 2 & 4 & 6 & 7 & 5 & 8 \end{pmatrix}$$

The mapping Δ_6 gives an assignment for row R6 of R as the solution row S6 of S as :-

$$S6 = \Big[\begin{pmatrix} 7,\ 8,\ 16,\ 20, \\ 202,203,204,211 \end{pmatrix} \begin{pmatrix} 6,\ 11,\ 19,\ 21, \\ 202,203,204,211 \end{pmatrix}$$

$$\begin{pmatrix} 7,\ 8,\ 16,\ 20, \\ 202,203,204,211 \end{pmatrix} \begin{pmatrix} 6,\ 15,\ 20,\ 21, \\ 202,203,204,211 \end{pmatrix}$$

$$\begin{pmatrix} 4,5,10,12,13,15,18,23 \\ 202,203,204,211 \end{pmatrix} \begin{pmatrix} 1,\ 2,\ 3,\ 21, \\ 202,203,204,211 \end{pmatrix}$$

$$\begin{pmatrix} 2,\ 14,\ 17,\ 19 \\ 202,203,204,211 \end{pmatrix} \begin{pmatrix} 4,5,10,12,13,15,18 \\ 23,202,203,204,211 \end{pmatrix} \Big]$$

Since all other 2nd year level classes are related to this assignment of row R6 through the teacher-class sets then

S6 = S7 = S8 = S9

The above assignment indicates for example that resources 6, 15, 20, 21, 201, 202, 203, 204, 211 are allocated in the related activity to time-period 4 since this set of resources occurs in the fourth position of row S6.

The implication algorithm determines all implications of this assignment on rows R1 to R5, R10 to R13 of R for feasibility. Then a new precedence is calculated and the allocation begins for the new class, etc. The details of the solution produced are given in tables C.2.

In the above description the details of the reduction algorithm have been omitted, as have those related to the bijective mapping generator. These may be determined by refering to chapters 5 and 6, and have therefore not been stated again. A problem with no solution arising during the Craigmore investigations will now be discussed.

8.4 A PROBLEM WITH NO SOLUTION AT CRAIGMORE

An automated timetable method is of little practical value to school administrators when a problem has no solution, unless it also gives details that indicate the reasons for the infeasibility of such a problem. The timetable procedure described within this thesis contains error detecting devices that enable manual alterations to be made to the input requirement data once an infeasible problem has been discovered. The error detection is accomplished mainly through the clash matrix and resource load matrix (chapter 6, section 6.5) together with output messages indicating the trouble spots. A practical problem that had no solution is discussed. All details are contained in Appendix D. The problem arose during investigations at Craigmore High School when the administrators were varying teacher allocations to classes to arrive at different timetable patterns.

Table D.1 of Appendix D contains the details of resources required for each school activity. Upon entering the problem the computer returned a 'no solution' result and indicated that a problem area was located with the 2nd year level classes. The precedence list determined at this stage showed that the order of assignment was 301, 302, 303, 311, 201, and thus the 3rd year level classes had been successfully allocated. The printout further indicated that the problem area was caused by the teacher-class set of activity (53,54) of table D.1 involving the resources 201, 202, 203, 211, 3, 5, 13, 14, 19, 22 for a block-period size 2. Upon consulting the clash matrix of table D.2 it is found that this combination of required teacher resources clashes with every teacher-class set of the 3rd year level classes, i.e. no teacher-class set of the 3rd year level classes can be allocated to the same time-period as this

2nd year level set. This required a re-organization of this combination of teachers for the activity by the school administrators. With the use of the teacher load matrix and the clash matrix this was easily accomplished since all details of the teacher resources involved and loads of all staff members was detailed. Craigmore administrators accept these two matrices, together with the directions printed by the computer program as an important aspect of this solution method. The time taken to determine that no solution existed was approximately 1 sec. C.P. time.

Other more complicated problems were determined to have no solution but the method of detection and correction was still the same. In all cases the school administrator located the problem area quickly and made the required corrections. The maximum time taken to determine that no solution existed was less than 2 minutes C.P. time on a Craigmore problem involving 23 staff, 8 time-period day and 13 classes.

8.5 CONCLUSIONS ON THE CRAIGMORE PROBLEM

The computer program was shown to be practical for the school situation and results were produced in much less time than by manual methods. The deputy headmaster of Craigmore High School indicated that manual methods had required two weeks to produce a solution earlier in the year. The computer method needed a day for the compilation of manual data, a time of a few hours for card punching and relatively few minutes for computer processing. In all only 2 days were involved using this automated solution method.

The system solved all practical problems and indicated "no solution" situations when they arose. All results were acceptable and were easily initiated into the school organization. No solution problems were quickly corrected through the clash and resource load matrices. The system was found to be of benefit in staff utilization since several arrangements of teacher-class sets could be tried and results produced compared. This was not possible previously due to the time involved by the use of manual methods. Thus an optimal solution could be achieved for the school organization in preference to the ad hoc methods employed to produce any solution.

CHAPTER 9

DISCUSSION

9.1 DISCUSSION

Publications on the topic of school timetables are many and varied, but still there remain relatively few that have practical computer programs for the solution of real-life school problems. In many cases the models, as presented, are highly theoretical, and bear little resemblance to the practical situations. Others attempt to formulate in a computer program, heuristic techniques that have been applied to specific school problems, with no guarantee that a solution will be produced.

Reports of successful approaches to the problem, such as the Stanford School Scheduling System (S.S.S.S.), Ontario School Timetable System and the Generalized Academic Simulator Program (GASP) of M.I.T. have been noted. However, these systems are costly in terms of computer time, with no guarantee of a successful solution (assuming a solution does exist to a given problem). The cost factor is even more critical when one body, such as the Education Department of South Australia, must absorb the expense for some 147 solutions to the 147 timetables present in this state. There was also a need for investigations into the detection of infeasibilities in problems that had no solution. This was important for administrators that were not in direct contact with the computer centre processing solutions. Thus the broad aims of the research of this thesis were first, to investigate a practical solution method for the production of school timetables for South Australian secondary schools, keeping in mind the cost factor, and second, to provide effective error-detection-correction techniques for problems that had no solution so that problems could be corrected quickly.

The success of the method presented in this thesis depended firstly, upon the effectiveness of the implication algorithm that reduced infeasible possibilities from each stage of the solution procedure, and second, on the ability of the bijective mapping algorithm to generate only feasible allocations for class requirements at each assignment stage. The method of approach was to consider daily problems, and each assignment stage allocated time-periods to a set of daily class-activity requirements. The daily approach has several advantages over the theoretically optimal weekly methods. Firstly it reduces the larger problem to a more managable size. Second, it permits direct administrative control over the distribution of course and resource loads for the school week. Third, it allows for the possibility of a partial weekly solution if one or more daily problems are infeasible. Apart from any practical or theoretical advantages, the daily approach was requested by school administrators, so that there was still some control over the layout of the solution timetable, as produced by the computer.

There are a number of difficulties within secondary school timetables. First, the demand for full-time teachers exceeds the supply. Thus part-time staff are employed, but they have a limiting effect on the timetable, caused by their restricted availability, and on some occasions, fixed times for lessons. Second, students in the upper levels of the school may choose from a variety of course combinations. Ideally, all combinations of subjects should be available for selection, provided that the examination conditions are satisfied, but due to the limited teacher resources, the number of course options is, in practice, limited. Nevertheless, teacher-class sets are constructed to increase the number of possible course options at the expense of increased complexity in the time-table problem.

Lesson distributions impose a further restriction on the problem. Block-periods of two or more consecutive lessons are often required in the timetable solution, together with the desired even distribution of teacher and course loads throughout the school week. The block-periods impose further restrictions when they are confined to specific time-periods of a school day, e.g. a block-period size two can only be allocated in time-periods 1-2 or 2-3, or 4-5 or 5-6 in a 7 period day.

Lastly, the time for the production of manual timetables can take up to 3 weeks, and thus impose administrative burdens on students at the start of the school term. Similar difficulties arise during the year when new solutions are necessary because of staff changes. It is therefore desirable that a more efficient method, in terms of time, be found for the production of school timetables.

The daily, combinatorial approach, presented in this thesis considers the meetings of classes and teachers as activities. The activity oriented method is advantageous since lessons in practical situations may involve varying numbers of school resources (teachers, classes, rooms, equipment, etc.). Another important feature of practical significance is the composite availability vector, that indicated the combined availability for resources required for an activity, for each time-period of the timetable. Thus, instead of having to allocate several resources individually to a specific time period, the assignment procedure had only to determine a single allocation.

The daily class-activity assignment technique used at each assignment stage was an important practical feature. The implications associated with an allocation of several activities at the one time permit an early recognition of infeasible situations. Thus 'faulty assignments' could be recognised more quickly and alterations made. This approach also permits early recognition of infeasible problems.

The clash matrix had important applications in two areas. These were first for indicating feasible pairs of activities for assignment to common time-periods, and second, in the detection and correction of infeasibilities of problemswith no solution. This technique has also been found useful in the production of manual timetables, and is used as an aid for manual timetable construction.

Data storage and manipulation is considered to be important since the speed of solution is associated with the data of the problem. The approach used in this work involved packing data into computer words using the individual bits,and operating on the data with logical operators. This technique contributed to the speed of the solution method.

The program was tested on the Craigmore High School timetable problem and the solution was produced. The practical features were consistent with the needs of the school, and Craigmore is at present operating under a timetable produced by the computer program presented here. The method is to be progressively adopted in other departmental schools. The Education Department has accepted the program and has aims of extending its use to other aspects of resource utilization. It is foreseen that the program will be applicable to other schools beside the departmental schools.

9.2 FUTURE RESEARCH

Problems associated with the manual production of school timetables are becoming increasingly more difficult. There is a need for more research into practical problems associated with real-school situations, to overcome not only the complexities of varying course requirements, but also to make them amenable to solution by automated methods.

Further research into the size and composition of allocation groups could be undertaken. In its present form, the allocation-group consists of the daily activities of a class, that are allocated to time-periods in one assignment stage within the solution method described in this thesis. Then, through the implication algorithm, the combined effect of these activities on the unallocated requirements of the timetable problem is determined. The benefits of any change in the allocation-group size should be weighed against such suggested factors as the increased computation time necessary to study the implications of an assignment stage, the ability to quickly detect infeasibilities, and the complexity of the implication algorithm itself to cope with any variation in the group size. This work could result in the determination of an optimal allocation-group size for the timetable problem.

There is a need for research into problems that have no solution. It appears that there are at least three aspects that such a study could encompass. The first involves the determination of conditions and constraints most likely to cause infeasibilities in timetable problems. These could then be placed in some relative order such that constraints most likely to cause a problem to have no solution would be identified. Second, further investigations into error detection and correction techniques such as the clash matrix of this thesis, could be beneficial. The importance of of indications into causes for infeasibilities should not be neglected and the fact that school administrators would no longer be completely conversant with the stages of construction of the timetable solution should not be overlooked. The third, aspect is associated with the quantification of computer time, necessary to determine that problems have no solution. At present, the method will determine that no solution exists for a given problem, but no accurate estimate of computer time can be made for the computation of this result.

Another problem that calls for study is the allocation of teachers to classes, according to the subject requirements of a school. This work could be linked with a study on staffing of schools as determined by course and student needs. Such research would seek an optimal staffing strategy for schools with staffing problems.

The concept of an activity-oriented timetable within schools, as described in this work is advantageous since the number of resources involved in lessons is not necessarily constant. The use of composite availability vectors to describe the available time-periods for activities could be used with other techniques for the allocation of lesson times. e.g. PERT. The practical aspects of the school timetable problem should not be overlooked when new methods of assignments are considered.

The approach presented in this thesis has already been beneficial to the South Australian Education Department. Computer generated timetables are being produced and further extensions to the work are anticipated in the future.

Hemmerling, M. B. (1972, May). A computer timetable solution for the South Australian Secondary Schools (437-440). *Proceedings of 5th Australian Computer Conference*, Australian Computer Society, [Brisbane], Qld.

NOTE:

This publication is included in the print copy
of the thesis held in the University of Adelaide Library.

APPENDIX B

The School data for the five daily timetable problems of Craigmore High School. The first table indicates the codes assigned to the teacher. resources of the school. Tables 2 to 6 contain data for the 5 daily problems in the form of :-

Required activity : Resources of the school required for the activity : Number of time-periods involved (if a repeated activity)

: block-period size required : fixed time-period designation if required

e.g. Multiplicity = 2 implies the activity is to be allocated twice in the solution.

Block = 2 implies a double lesson in consecutive time-periods.

Tables 7 and 8 are the teacher-resource load matrix and clash sub-matrix for the Tuesday problem discussed in Chapter 8.

TABLE B.1

TEACHER RESOURCE DATA FOR CRAIGMORE HIGH SCHOOL

H.M.	-	Headmaster	T	-	Teacher
D.H.M.	-	Deputy Head Master	F	-	Full-time
S.M.	-	Senior Master/Mistress	P	-	Part-time
R.I.	-	Religious Instructor			

Code	Name	Positional Status	Full-time or Part-time	Subject Teaching
01	W.C.	H.M.	F	Sc.
02	B.C.	T	F	Sc, Ma.
03	K.C.	S.M.	F	Sc, Ma.
04	E.C.	T	F	Cr.
05	A.D.	S.M.	F	Gg, Hist, Ma.
06	B.D.	T	F	P.E., Sc.
07	S.F.	T	F	Library, Eg, Fr.
08	H.F.	T	P	Eg, Hist.
09	M.G.	T	F	Basic Elect., Ma, Sc.
10	R.G.	T	F	Hist, Eg, Ty, Library
11	C.I.	T	F	Hist, Eg, Library
12	J.L.	T	F	Civics, Eg, Hist, As.St.
13	I.M.	T	F	Library

Code	Name	Positional Status	Full-time or Part-time	Subject Teaching
14	A.M.	T	F	Ma.
15	H.N.	T	F	Art
16	R.R.	T	F	Ma, Gm.
17	P.R.	D.H.M.	F	Ma.
18	C.R.	T	F	Art
19	T.R.	T	F	Ma, Gg.
20	J.R.	S.M.	F	Eg, Ma, Cons. Ed., Gg.
21	J.S.	S.M.	F	Sc., G.g. Comp.Sc.
22	I.W.	T	F	Fr., Eg, Hist, Film.
23	L.W.	T	F	Cr.
24	*External persons		P	R.I.
25			P	R.I.

Note : Classes and other resources have not been included on this list. Class codes as discussed in chapter 3, section 3.2 will be used in the description to avoid confusion.

TABLE B.2.

THE RESOURCE REQUIREMENT LIST DATA INPUT

FOR MONDAY

ACTIVITY	CLASSES	TEACHERS	OTHERS	MULTI-PLICITY	BLOCK	FIXED
(1,2)	101	12		2	1	-
(3, 4)	101	3		2	1	-
(5,6)	101	16		1	1	-
(7,8)	101	6		1	1	-
(9,10)	101,103	22,16		1	1	-
(11,12)	101	10		1	1	-
(13,14)	102	11		1	1	-
(15,16)	102,104,105	6,7,16		2	1	-
(17,18)	102	7		2	1	-
(19,20)	102	9		2	1	-
(21,22)	102	1		1	1	-
(23,24)	103	5		2	2	-
(25,26)	103	19		1	1	-
(27,28)	103	2		2	1	-
(29,30)	103	10		1	1	-
(31,32)	103	11		1	1	-

TABLE B.2 (CONT'D)

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(33,34)	104	14		2	1	-
(35,36)	104	11		2	1	-
(37,38)	104	15,18		1	1	-
(39,40)	104	3		1	1	-
(41,42)	105	10		1	1	-
(43,44)	105	5		2	2	-
(45,46)	105	12		1	1	-
(47,48)	105	14		1	1	-
(49,50)	105	5		1	1	-
(51,52)	201,202,203,211,	7,8,20,22		1	1	-
(53,54)	201,202,203,211	1,2,9,15,18		2	2	-
(55,56)	201,202,203,211	14,17,19,20		2	1	-
(57,58)	201,202,203,211	4,5,10,12,13, 15,18,23		1	1	-
(59,60)	201,202,203,211	6,11,19,21		1	1	-
(61,62)	201,202,203,211	4,10,15,18,20, 22,23		1	1	-
(63,64)	301,302,303,311	2,3,9,21		1	1	-
(65,66)	301,302,303,311	8,12,20,22		2	1	-
(67,68)	301,302,303,311	4,9,10,15,18, 21,22,23		2	1	-
(69,70)	301,302,303,311	2,3,17,21		1	1	-
(71,72)	301,302,303,311	2,3,4,17,23		1	1	-
(73,74)	301,302,303,311	6,7,16,19		1	1	-

TABLE B.3

THE RESOURCE REQUIREMENTS LIST FOR TUESDAY

RESOURCES

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(1,2)	101	15,18		2	2	-
(3,4)	101	6		1	1	-
(5,6)	101,103	16,22		1	1	-
(7,8)	101	12		1	1	-
(9,10)	101	10		1	1	-
(11,12)	101	3		1	1	-
(13,14)	101	16		1	1	-
(15,16)	102	11		1	1	-
(17,18)	102	9		2	1	-
(19,20)	102	15,18		1	1	-
(21,22)	102	5		2	2	-
(23,24)	102	4,23		1	1	-
(25,26)	102	1		1	1	-
(27,28)	103	19		2	1	-
(29,30)	103	10		2	1	-
(31,32)	103	11		2	1	-
(33,34)	103	2		1	1	-

TABLE B.3 (Contd.)

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(35,36)	104	14		2	1	–
(37,38)	104	3		1	1	–
(39,40)	104	12		1	1	–
(41,42)	104	11		2	1	–
(43,44)	104	15,18		2	2	–
(45,46)	105	4,23		2	2	–
(47,48)	105	5		2	1	–
(49,50)	105	14		2	1	–
(51,52)	105	11		2	1	–
(53,54)	201,202,203,211	7,8,16,20		2	1	–
(55,56)	201,202,203,211	6,11,19,21		1	1	–
(57,58)	201,202,203,211	6,15,20,21		1	1	–
(59,60)	201,202,203,211	1,2,3,21		1	1	–
(61,62)	201,202,203,211	2,14,17,19		1	1	–
(63,64)	201,202,203,211	4,5,10,12,13,15, 18,23		2	1	–
(65,66)	301,302,303,311	2,3,9,21		2	1	–
(67,68)	301,302,303,311	8,12,20,22		1	1	–
(69,70)	301,302,303,311	2,3,4,17,23		1	1	3
(71,72)	301,302,303,311	2,3,4,17,23		1	1	4
(73,74)	301,302,303,311	6,7,16,19		2	1	–
(75,76)	301,302,303,311	6,12,20,21,22		1	1	–

TABLE B.4

THE RESOURCE REQUIREMENTS LIST FOR WEDNESDAY

RESOURCES

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(1,2)	101	4,23		2	2	-
(3,4)	101	12		2	1	-
(5,6)	101	16		1	1	-
(7,8)	101	3		1	1	-
(9,10)	101	5		1	1	-
(11,12)	101,103	16,22		1	1	-
(13,14)	102	9		1	1	-
(15,16)	102	11		1	1	-
(17,18)	102	1		1	1	-
(19,20)	102	7		2	1	-
(21,22)	102,104,105	6,7,16		1	1	-
(23,24)	102	15,18		2	2	-
(25,26)	103	19		2	1	-
(27,28)	103	2		1	1	-
(29,30)	103	11		2	1	-
(31,32)	103	13		1	1	-
(33,34)	103	10		1	1	-
(35,36)	104	14		1	1	-
(37,38)	104	12		2	1	-
(39,40)	104	4,23		2	2	-

TABLE B.4 (Contd.)

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(41,42)	104	11		1	1	-
(43,44)	104	3		1	1	-
(45,46)	105	11		1	1	-
(47,48)	105	15,18		1	1	-
(49,50)	105	10		2	1	-
(51,52)	105	14		2	1	-
(53,54)	105	4,23		1	1	-
(55,56)	201,202,203,211	7,8,20,24		2	1	-
(57,58)	201,202,203,211	7,8,16,20		1	1	-
(59,60)	201,202,203,211	1,2,3,9		2	1	-
(61,62)	201,202,203,211	2,14,17,19		1	1	-
(63,64)	201,202,203,211	2,4,14,19,23		1	1	-
(65,66)	201,202,203,211	6,11,19,21		1	1	-
(67,68)	301,302,303,311	2,3,17,21		2	1	-
(69,70)	301,302,303,311	3,9,15,18,21		1	1	3
(71,72)	301,302,303,311	5,10,13,19,20,22		1	1	4
(73,74)	301,302,303,311	5,10,13,19,20,22		1	1	5
(75,76)	301,302,303,311	6,12,20,21,22		2	1	-
(77,78)	301,302,303,311	4,5,9,10,17,18, 23		1	1	-

TABLE B.5

THE RESOURCE REQUIREMENT LIST DATA INPUT FOR THURSDAY

*Religious Instruction teachers (not part of teaching staff)

RESOURCES

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(1,2)	101	6		2	1	-
(3,4)	101	10		2	1	-
(5,6)	101	12		2	1	-
(7,8)	101	4,23		1	1	-
(9,10)	101,102,103,104) 105) 201,202,203,211) 301,302,303,311)	24,25*		1	1	8
(11,12)	102	11		1	1	-
(13,14)	102	4,23		2	2	-
(15,16)	102	7		1	1	-
(17,18)	102,104,105	6,7,16		1	1	-
(19,20)	102	9		1	1	-
(21,22)	102	5		1	1	-
(23,24)	103	19		1	1	-
(25,26)	103	15,18		2	2	-
(27,28)	103	10		1	1	-
(29,30)	103	11		1	1	-
(31,32)	103	4,23		2	2	-

TABLE B.5 (Contd.)

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(33,34)	104	3		1	1	-
(35,36)	104	5		2	2	-
(37,38)	104	12		1	1	-
(39,40)	104	11		1	1	-
(41,42)	104	14		1	1	-
(43,44)	105	5		1	1	-
(45,46)	105	14		1	1	-
(47,48)	105	10		1	1	-
(49,50)	105	11		1	1	-
(51,52)	105	15,18		2	2	-
(53,54)	201,202,203,211	7,8,16,20		1	1	-
(55,56)	201,202,203,211	6,11,19,21		1	1	-
(57,58)	201,202,203,211	7,8,20,22,		1	1	-
(59,60)	201,202,203,211	5,13,14,15,18, 19,22		2	2	-
(61,62)	201,202,203,211	2,14,17,19		1	1	-
(63,64)	201,202,203,211	1,2,3,9		1	1	-
(65,66)	301,302,303,311	4,9,10,15,18, 21,22,23		1	1	-
(67,68)	301,302,303,311	8,12,20,22		1	1	-
(69,70)	301,302,303,311	2,3,17,21		1	1	-
(71,72)	301,302,303,311	2,3,9,21		2	1	-
(73,74)	301,302,303,311	6,12,20,21,22		1	1	-
(75,76)	301,302,303,311	6,7,16,19		1	1	-

TABLE B.6

THE RESOURCE REQUIREMENT LIST DATA INPUT FOR FRIDAY

RESOURCES

ACTIVITY	CLASSES	TEACHERS	OTHER	MULTI-PLICITY	BLOCK	FIXED
(1, 2)	101	15,18		1	1	-
(3, 4)	101	6		1	1	-
(5, 6)	101	3		1	1	-
(7, 8)	101,103	16,22		2	1	-
(9, 10)	101	10		1	1	-
(11,12)	101	5		2	2	-
(13, 14)	102	9		2	1	-
(15, 16)	102	11		1	1	-
(17, 18)	102	1		2	1	-
(19, 20)	102	7		2	1	-
(21, 22)	102,104,105	6,7,16		1	1	-
(23, 24)	103	5		1	1	-
(25, 26)	103	19		3	1	-
(27, 28)	103	4,23		1	1	-
(29, 30)	103	2		1	1	-
(31, 32)	104	4,23		1	1	-
(33, 34)	104	3		1	1	-
(35, 36)	104	14		2	1	-
(37, 38)	104	5		1	1	-
(39, 40)	104	11		1	1	-
(41, 42)	104	12		1	1	-

TABLE B.6 (Contd.)

ACTIVITY	CLASSES	TEACHERS	OTHERS	MULTI-PLICITY	BLOCK	FIXED
(43, 44)	105	10		2	1	-
(45, 46)	105	14		2	1	-
(47, 48)	105	11		1	1	-
(49, 50)	105	5		2	1	-
(51, 52)	201,202,203,211	6,11,19,21		1	1	-
(53, 54)	201,202,203,211	7,8,20,22		1	1	-
(55, 56)	201,202,203,211	7,8,16,20		1	1	-
(57, 58)	201,202,203,211	4,19,14,20,23		2	2	-
(59, 60)	201,202,203,211	1,2,3,9		1	1	-
(61, 62)	201,202,203,211	4,10,15,18,20,22, 23		2	2	-
(63, 64)	301,302,303,311	8,12,20,22		1	1	-
(65, 66)	301,302,303,311	4,5,9,10,15,17, 18,23		2	2	-
(67,68)	301,302,303,311	3,9,15,18,21		2	2	-
(69,70)	301,302,303,311	6,12,20,21,22		1	1	-
(71, 72)	301,302,303,311	2,3,17,21		1	1	-
(73, 74)	301,302,303,311	6,7,16,19		1	1	-

TABLE B.7

THE DAILY TEACHER RESOURCE LOADS

* A part-time teacher

	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
1	3 : 8	2 : 8	3 : 8	1 : 8	3 : 8
2	7 : 8	7 : 8	7 : 8	5 : 8	3 : 8
3	6 : 8	7 : 8	6 : 8	5 : 8	6 : 8
4	5 : 8	7 : 8	7 : 8	6 : 8	8 : 8
5	6 : 8	6 : 8	2 : 8	6 : 8	8 : 8
6	5 : 8	6 : 8	4 : 8	6 : 8	5 : 8
7	6 : 8	4 : 8	6 : 8	5 : 8	6 : 8
8*	3 : 3	3 : 3	3 : 3	3 : 3	3 : 3
9	7 : 8	4 : 8	5 : 8	5 : 8	7 : 8
10	7 : 8	5 : 8	6 : 8	5 : 8	7 : 8
11	5 : 8	8 : 8	6 : 8	5 : 8	4 : 8
12	6 : 8	6 : 8	6 : 8	5 : 8	3 : 8
13	1 : 8	2 : 8	3 : 8	2 : 8	0 : 8
14	5 : 8	5 : 8	5 : 8	5 : 8	6 : 8
15	7 : 8	8 : 8	6 : 8	7 : 8	7 : 8
16	5 : 8	6 : 8	4 : 8	3 : 8	5 : 8
17	4 : 8	3 : 8	4 : 8	2 : 8	3 : 8
18	7 : 8	7 : 8	6 : 8	7 : 8	7 : 8
19	5 : 8	6 : 8	7 : 8	6 : 8	7 : 8
20	6 : 8	5 : 8	6 : 8	4 : 8	8 : 8
21	5 : 8	6 : 8	6 : 8	6 : 8	5 : 8
22	7 : 8	3 : 8	7 : 8	6 : 8	7 : 8
23	5 : 8	7 : 8	7 : 8	6 : 8	8 : 8
24*	0 : 0	0 : 0	0 : 0	1 : 1	0 : 0
25*	0 : 0	0 : 0	0 : 0	1 : 1	0 : 0

Note : The resource load matrix indicates the number of activities each resource (teacher) is required to be allocated in the timetable solution.

TABLE B.8

The clash sub-matrix for the Tuesday timetable problem. (0 = cannot allocate activities to common time-periods)

* indicates activities (67,68) and (5,6) cannot be allocated to the same time-period because of common requirements for resource 22.

activity	T1 (1,2)	T3 (5,6)	T27 (53,54)	T28 (55,56)	T29 (57,58)	T30 (59,60)	T31 (61,62)	T32 (63,64)	T33 (65,66)	T34 (67,68)	T35 (69,70)	T37 (73,74)	T38 (75,76)
(1,2)	0	1	1	1	1	1	1	0	1	1	1	1	1
(5,6)	1	0	0	1	1	1	1	1	1	1	1	0	1
(53,54)	1	0	0	1	0	1	1	1	1	0	1	0	0
(55,56)	1	1	1	0	0	0	0	1	0	1	1	0	0
(57,58)	1	1	0	0	0	0	1	0	0	0	1	0	0
(59,60)	1	1	1	0	0	0	0	1	0	1	0	0	0
(61,62)	1	0	1	0	1	0	0	1	0	1	0	1	1
(63,64)	0	1	1	1	0	1	1	0	1	0	0	0	0
(65,66)	1	1	1	0	0	0	0	1	0	1	0	0	0
(67,68)	1	0*	0	1	0	1	1	0	1	0	1	0	0
(69,70)	1	1	1	1	1	0	0	0	0	1	0	1	1
(73,74)	1	0	0	0	0	1	0	1	1	1	1	0	0

TABLE B.8 (Contd.)

activity	T1 (1,2)	T3 (5,6)	T27 (53,54)	T28 (55,56)	T29 (57,58)	T30 (59,60)	T31 (61,62)	T32 (63,64)	T33 (65,66)	T34 (67,68)	T35 (69,70)	T37 (73,74)	T38 (75,76)
(75,76)	1	0	0	0	0	0	1	0	0	0	1	0	0
Teacher Resources 1	1	1	1	1	1	0	1	1	1	1	1	1	1
2	1	1	1	1	1	0	0	1	0	1	0	1	1
3	1	1	1	1	1	0	1	1	0	1	0	1	1
4	1	1	1	1	1	1	1	0	1	1	0	1	1
5	1	1	1	1	1	1	1	0	1	1	1	1	1
6	1	1	1	0	0	1	1	1	1	1	1	0	0
7	1	1	0	1	1	1	1	1	1	1	1	0	1
8	1	1	0	1	1	1	1	1	1	0	1	1	1
9	1	1	1	1	1	1	1	1	0	1	1	1	1
10	1	1	1	1	1	1	1	0	1	1	1	1	1
11	1	1	1	0	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	0	1	0	1	1	0
13	1	1	1	1	1	1	1	0	1	1	1	1	1
14	1	1	1	1	1	1	0	1	1	1	1	1	1

TABLE B.8 (Contd.)

activity	T1 (1,2)	T3 (5,6)	T27 (53,54)	T28 (55,56)	T29 (57,58)	T30 (59,60)	T31 (61,62)	T32 (63,64)	T33 (65,66)	T34 (67,68)	T35 (69,70)	T37 (73,74)	T38 (75,76)
15	0	1	1	1	0	1	1	0	1	1	1	1	1
16	1	0	0	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	0	1	1	1	0	1	1
18	0	1	1	1	1	1	1	0	1	1	1	1	1
19	1	1	1	0	1	1	0	1	1	1	1	1	1
20	1	1	0	1	0	1	1	1	1	0	1	0	0
21	1	1	1	0	0	0	1	1	0	1	1	0	0
22	1	0	1	1	1	1	1	1	1	0	1	0	1
23	1	1	1	1	1	1	0	1	1	1	0	1	1

DAY	TEACHER CODE	PERIODS NOT AVAILABLE FOR ALLOCATION
Monday	8	4, 5, 6, 7, 8
Tuesday	8	4, 5, 6, 7, 8
Wednesday	8	4, 5, 6, 7, 8
Thursday	8	4, 5, 6, 7, 8
Thursday	24,25	1, 2, 3, 4, 5, 6, 7
Friday	8	4, 5, 6, 7, 8

TABLE B.9

Table of the weekly time-periods that are not available for allocation, for the resources indicated, within the timetable solution.

Note : The table defines the unavailable time-period for each of the part-time resources.

APPENDIX C

The five daily timetable solutions for the Craigmore High School problem have been tabulated in the following form :-

Time-periods allocated by the computer program	:	Required activity and resources as described by tables in Appendix B

Notes have been included where necessary to indicate special features included in the solutions. Attention is drawn to the fact that various output formats are possible to give solutions in the form of teacher timetables, class timetables or period timetables (as presented in this section). The other forms are mentioned in the text. By relating the codes shown in these tables to the initials of table B.1, appendix B, the teacher initials may be substituted, and subjects related to the lessons.

Allocated Time-period	Required Activity	Required Resources
1	(3, 4)	101, 3
	(15, 16)	102, 104, 105, 6, 7, 16
	(23, 24)	103, 5
	(53, 54)	201, 202, 203, 211, 1, 2, 9, 15, 18
	(65, 66)	301, 302, 303, 311, 8, 12, 20, 22
2	(11, 12)	101, 10
	(15, 16)	102, 104, 105, 6, 7, 16
	(23, 24)	103, 5
	(53, 54)	201, 202, 203, 211, 1, 2, 9, 15, 18
	(65, 66)	301, 302, 303, 311, 8, 12, 20, 22
3	(1, 2)	101, 12
	(13, 14)	102, 11
	(25, 26)	103, 19
	(33, 34)	104, 14
	(41, 42)	105, 10
	(51, 52)	201, 202, 203, 211, 7, 8, 20, 22
	(63, 64)	301, 302, 303, 311, 2, 3, 9, 21

Allocated Time-period	Required Activity	Required Resources
4	(1, 2)	101, 12
	(17, 18)	102, 7
	(27, 28)	103, 2
	(35, 36)	104, 11
	(49, 50)	105, 5
	(55, 56)	201, 202, 203, 211, 14, 17, 19, 20
	(67, 68)	301, 302, 303, 311, 4, 9, 10, 15, 18, 21, 22, 23
5	(9, 10)	101, 103, 16, 22
	(21, 22)	102, 1
	(35, 36)	104, 11
	(47, 48)	105, 14
	(57, 58)	201, 202, 203, 211, 4, 5,10, 12, 13, 15, 18, 23
	(69, 70)	301, 302, 303, 311, 2, 3, 17, 21
6	(5, 6)	101, 16
	(19, 20)	102, 9
	(29, 30)	103, 10
	(37, 38)	104, 15, 18
	(43, 44)	105, 5
	(59, 60)	201, 202, 203, 211, 6, 11, 19, 21
	(71, 72)	301, 302, 303, 311, 2, 3, 4, 17, 23

Allocated Time-period	Required Activity	Required Resources
7	(3, 4)	101, 3
	(19, 20)	102, 9
	(31, 32)	103, 11
	(33, 34)	104, 14
	(43, 44)	105, 5
	(61, 62)	201, 202, 203, 211, 4, 10, 15, 18, 20, 22, 23
	(73, 74)	301, 302, 303, 311, 6, 7, 16, 19
8	(7, 8)	101, 6
	(17, 18)	102, 7
	(27, 28)	103, 2
	(39, 40)	104, 3
	(45, 46)	105, 12
	(55, 56)	201, 202, 203, 211, 14, 17, 19, 20
	(67, 68)	301, 302, 303, 311, 4, 9, 10, 15, 18, 21, 22, 23

TABLE C.1

Solution in order of time-periods for the Monday timetable problem of Craigmore High School.

(Data presented in table B.2, Appendix B)

Note : that all classes are fully utilized for every time-period.

Allocated Time-period	Required Activity	Required Resources
1	(1, 2)**	101, 15, 18
	(23, 24)	102, 4, 23
	(27, 28)	103, 19
	(39, 40)	104, 12
	(51, 52)	105, 11
	(53, 54)	201, 202, 203, 211, 7, 8, 16, 20
	(65, 66)	301, 302, 303, 311, 2, 3, 9, 21
2	(1, 2)**	101, 15, 18
	(17, 18)	102, 9
	(29, 30)	103, 10
	(39,40)	104, 3
	(47, 48)	105, 5
	(55, 56)	201, 202, 203, 211, 6, 11, 19, 21
	(67, 68)	301, 302, 303, 311, 8, 12, 20, 22
3	(3, 4)	101, 6
	(19, 20)	102, 15, 18
	(29, 30)	103, 10
	(41, 42)	104, 11
	(49, 50)	105, 14
	(53, 54)	201, 202, 203, 211, 7, 8, 16, 20
	(69, 70)*	301, 302, 303, 311, 2, 3, 4, 17, 23

Allocated Time-period	Required Activity	Required Resources
4	(5, 6)	101, 103, 16, 22
	(15, 16)	102, 11
	(35, 36)	104, 14
	(47, 48)	105, 5
	(57, 58)	201, 202, 203, 211, 6, 15, 20, 21
	(71, 72)*	301, 302, 303, 311, 2, 3, 4, 17, 23
5	(11, 12)	101, 3
	(17, 18)	102, 9
	(33, 34)	103, 2
	(35, 36)	104, 14
	(51, 52)	105, 11
	(63, 64)	201, 202, 203, 211, 4, 5, 10, 12, 13, 15, 18, 23
	(73, 74)	301, 302, 303, 311, 6, 7, 16, 19
6	(7, 8)	101, 12
	(21, 22)**	102, 5
	(31, 32)	103, 11
	(43, 44)**	104, 15, 18
	(45, 46)**	105, 4, 23
	(59, 60)	201, 202, 203, 211, 1, 2, 3, 21
	(73, 74)	301, 302, 303, 311, 6, 7, 16, 19

Allocated Time-period	Required Activity	Required Resources
7	(9, 10)	101, 10
	(21, 22)**	102, 5
	(31, 32)	103, 11
	(43, 44)**	104, 15, 18
	(45, 46)**	105, 4, 23
	(61, 62)	201, 202, 203, 211, 2, 14, 17, 19
	(75, 76)	301, 302, 303, 311, 6, 12, 20, 21, 22
8	(13, 14)	101, 16
	(25, 26)	102, 1
	(27, 28)	103, 19
	(41, 42)	104, 11
	(49, 50)	105, 14
	(63, 64)	201, 202, 203, 211, 4, 5, 10, 12/3, 15, 18, 23
	(65, 66)	301, 302, 303, 311, 2, 3, 9, 21

TABLE C.2

Solution in order of time-periods for the Tuesday timetable problem of Craigmore High School

(Data is presented in table B.3, Appendix B)

* fixed time-period requirements for periods 3 and 4

** block-period requirements allocated to consecutive lessons.

Allocated Time-period	Required Activity	Required Resources
1	(3, 4)	101, 12,
	(23, 24)	102, 15, 18
	(25, 26)	103, 19
	(39, 40)	104, 4, 23
	(45, 46)	105, 11
	(55, 56)	201, 202, 203, 211, 7, 8, 20, 22
	(67, 68)	301, 302, 303, 311, 2, 3, 17, 21
2	(5, 6)	101, 16
	(23, 24)	102, 15, 18
	(25, 26)	103, 19
	(39, 40)	104, 4, 23
	(49, 50)	105, 10
	(55, 56)	201, 202, 203, 211, 7, 8, 20, 22
	(67, 68)	301, 302, 303, 311, 2, 3, 17, 21
3	(3, 4)	101, 12
	(15, 16)	102, 11
	(27, 28)	103, 2
	(35, 36)	104, 14
	(53, 54)	105, 4, 23
	(57, 58)	201, 202, 203, 211, 7, 8, 16, 20
	(69, 70)*	301, 302, 303, 311, 3, 9, 15, 18, 21

Allocated Time-period	Required Activity	Required Resources
4	(1, 2)	101, 4, 23
	(19, 20)	102, 7
	(29, 30)	103, 11
	(37, 38)	104, 12
	(51, 52)	105, 14
	(59, 60)	201, 202, 203, 211, 1, 2, 3, 9
	(71, 72)*	301, 302, 303, 311, 5, 10, 13, 15, 18, 19, 22
5	(1, 2)	101, 4, 23
	(21, 22)	102, 104, 105, 6, 7, 16
	(29, 30)	103, 11
	(59, 60)	201, 202, 203, 211, 1, 2, 3, 9
	(73, 74)*	301, 302, 303, 311, 5, 10, 13, 19, 20, 22
6	(7, 8)	101, 3
	(13, 14)	102, 9
	(31, 32)	103, 13
	(41, 42)	104, 11
	(49, 50)	105, 10
	(61, 62)	201, 202, 203, 211, 2, 14, 17, 19
	(75, 76)	301, 302, 303, 311, 6, 12, 20, 21, 22

Allocated Time-period	Required Activity	Required Resources
7	(9, 10)	101, 5
	(17, 18)	102, 1
	(33, 34)	103, 10
	(43, 44)	104, 3
	(47, 48)	105, 15, 18
	(63, 64)	201, 202, 203, 211, 2, 4, 14, 19, 23
	(75, 76)	301, 302, 303, 311, 6, 12, 20, 21, 22
8	(11, 12)	101, 103, 16, 22
	(19, 20)	102, 7
	(37, 38)	104, 12
	(51, 52)	105, 14
	(65, 66)	201, 202, 203, 211, 6, 11, 19, 21
	(77, 78)	301, 302, 303, 311, 4, 5, 9, 10, 17, 18, 23

TABLE C.3

Solution in order of time-periods for the Wednesday timetable problem of Craigmore High School.

(Data presented in table B.4, Appendix B)

* fixed time-period requirements to the periods 3, 4, 5.

Allocated Time-period	Required Activity	Required Resources
1	(3, 4)	101, 10
	(13, 14)	102, 4, 23
	(25, 26)	103, 15, 18
	(35, 36)	104, 5
	(45, 46)	105, 14
	(55, 56)	201, 202, 203, 211, 6, 11, 19, 21
	(67, 68)	301, 302, 303, 311, 8, 12, 20, 22
2	(1, 2)	101, 6
	(13, 14)	102, 4, 23
	(25, 26)	103, 15, 18
	(35, 36)	104, 5
	(47, 48)	105, 10
	(57, 58)	201, 202, 203, 211, 7, 8, 20, 22
	(69, 70)	301, 302, 303, 311, 2, 3, 17, 21
3	(5, 6)	101, 12
	(11, 12)	102, 11
	(23, 24)	103, 19
	(33, 34)	104, 3
	(43, 44)	105, 5
	(53, 54)	201, 202, 203, 211, 7, 8, 16, 20
	(65, 66)	301, 302, 303, 311, 4, 9, 10, 15, 18, 21, 22, 23

Allocated Time-period	Required Activity	Required Resources
4	(1, 2)	101, 6
	(15, 16)	102, 7
	(31, 32)	103, 4, 23
	(37, 38)	104, 12
	(49, 50)	105, 11
	(59, 60)	201, 202, 203, 211, 5, 13, 14, 15, 18, 19, 22
	(71, 72)	301, 302, 303, 311, 2, 3, 9, 21
5	(3, 4)	101, 10
	(17, 18)	102, 104, 105, 6, 7, 16
	(31, 32)	103, 4, 23
	(59, 60)	201, 202, 203, 211, 5, 13, 14, 15, 18, 19, 22
	(71, 72)	301, 302, 303, 311, 2, 3, 9, 21
6	(7, 8)	101, 4, 23
	(19, 20)	102, 9
	(27, 28)	103, 10
	(39, 40)	104, 11
	(51, 52)	105, 15, 18
	(61, 62)	201, 202, 203, 211, 2, 14, 17, 19
	(73, 74)	301, 302, 303, 311, 6, 12, 20, 21, 22

Allocated Time-period	Required Activity	Required Resources
7	(5, 6)	101, 12
	(21, 22)	102, 5
	(29, 30)	103, 11
	(41, 42)	104, 14
	(51, 52)	105, 15, 18
	(63, 64)	201, 202, 203, 211, 1, 2, 3, 9
	(75, 76)	301, 302, 303, 311, 6, 7, 16, 19
8	(9, 10)*	101, 102, 103, 104, 105, 201, 202, 203, 211, 301, 302, 303, 311, 24, 25

TABLE C.4

Solution in order of time-periods for the Thursday timetable problem of Craigmore High School.

(Data presented in table B.5, Appendix B)

* a fixed time-period requirement for all classes in time-period 8 for the purpose of a religious instruction lesson. Teachers 24, 25 are external to the school as indicated in table B.5, Appendix B.

APPENDIX C

Allocated Time-period	Required Activity	Required Resources
1	(3, 4)	101, 6
	(15, 16)	102, 11
	(25, 26)	103, 19
	(33, 34)	104, 3
	(45, 46)	105, 14
	(55, 56)	201, 202, 203, 211, 7, 8, 16, 20
	(65, 66)	301, 302, 303, 311, 4, 5, 9, 10, 15, 17, 18, 23
2	(5, 6)	101, 3
	(17, 18)	102, 1
	(25, 26)	103, 19
	(35, 36)	104, 14
	(47, 48)	105, 11
	(53, 54)	201, 202, 203, 211, 7, 8, 20, 22
	(65, 66)	301, 302, 303, 311, 4, 5, 9, 15, 17, 18, 23
3	(1, 2)	101, 15, 18
	(13, 14)	102, 9
	(23, 24)	103, 5
	(31, 32)	104, 4, 23
	(43, 44)	105, 10
	(51, 52)	201, 202, 203, 211, 6, 11, 19, 21
	(63, 64)	301, 302, 303, 311, 8, 12, 20, 22

Time Time-period	Required Activity	Required Resources
4	(7, 8)	101, 103, 16, 22
	(19, 20)	102, 7
	(37, 38)	104, 5
	(43, 44)	105, 10
	(57, 58)	201, 202, 203, 211, 4, 14, 19, 20, 23
	(67, 68)	301, 302, 303, 311, 3, 9, 15, 18, 21
5	(7, 8)	101, 103, 16, 22
	(17, 18)	102, 1
	(39, 40)	104, 11
	(49, 50)	105, 5
	(57, 58)	201, 202, 203, 211, 4, 14, 19, 20, 23
	(67, 68)	301, 302, 303, 311, 3, 9, 15, 18, 21
6	(11, 12)	101, 5
	(21, 22)	102, 104, 105, 6, 7, 16
	(25, 26)	103, 19
	(61, 62)	201, 202, 203, 211, 4, 10, 15, 18, 20 22, 23
	(71, 72)	301, 302, 303, 311, 13, 17, 21

Allocated Time-periods	Required Activity	Required Resources
7	(11, 12)	101, 5
	(13, 14)	102, 9
	(29, 30)	103, 2
	(41, 42)	104, 12
	(45, 46)	105, 14
	(61, 62)	201, 202, 203, 211, 4, 10, 15, 18, 20, 22, 23
	(73, 74)	301, 302, 303, 311, 6, 7, 16, 19
8	(9, 10)	101, 10
	(19, 20)	102, 7
	(27, 28)	103, 4, 23
	(35, 36)	104, 14
	(49, 50)	105, 5
	(59, 60)	201, 202, 203, 211, 1, 2, 3, 9
	(69, 70)	301, 302, 303, 311, 6, 12, 20, 21, 22

TABLE C.5

Solution in order of time-periods for the Friday timetable problem of Craigmore High School.

(Date is presented in Table B.6, Appendix B)

APPENDIX D

The following tables relate to a practical problem that had no solution. The clash sub-matrix of Table D.2 indicates the activity causing this infeasibility. Table D.1 details the activities and resource requirements of the problem whilst Table D.3 summarizes the teacher loads. Table D.4 indicates the teacher availability constraints to be considered in the solution.

TABLE D.1

THE ACTIVITY AND RESOURCE REQUIREMENTS FOR A TIMETABLE PROBLEM THAT HAS NO SOLUTION

ACTIVITY	RESOURCES: CLASSES	TEACHERS	OTHERS	MULTI-PLICITY	BLOCK	FIXED
(1,2)	101	12		2	1	-
(3,4)	101	6		2	1	-
(5,6)	101	10		2	1	-
(7,8)	101	4,23		1	1	-
(9,10)	102	11		2	1	-
(11,12)	102	15,18		2	2	-
(13,14)	102	9		1	1	-
(15,16)	102	5		1	1	-
(17.18)	102,104,105	6,7,16		1	1	-
(19,20)	103	2		1	1	-
(21,22)	103	10		1	1	-
(23,24)	103	19		1	1	-
(25,26)	103	15,18		1	1	-
(27,28)	103	4,23		2	2	-
(29,30)	103	11		1	1	-
(31,32)	104	14		2	1	-
(33,34)	104	11		2	1	-
(35,36)	104	12		1	1	-
(37,38)	104	3		1	1	-

TABLE D.1 (Contd.)

ACTIVITY	CLASSES	TEACHERS	OTHERS	MULTI-PLICITY	BLOCK	FIXED
(39,40)	105	5		2	2	-
(41,42)	105	14		1	1	-
(43,44)	105	10		1	1	-
(45,46)	105	15,18		2	2	-
(47,48)	201,202,203,211	6,11,19,21		1	1	-
(49,50)	201,202,203,211	2,14,17,19		1	1	-
(51,52)	201,202,203,211	7,8,20,22		1	1	1
(53,54)	201,202,203,211	3,5,13,14,19,22		2	2	-
(55,56)	201,202,203,211	7,8,16,20		1	1	2
(57,58)	201,202,203,211	1,2,9,22		1	1	-
(59,60)	301,302,303,311	4,10,15,18,21, 22,23		1	1	-
(61,62)	301,302,303,311	8,12,20,22		1	1	3
(63,64)	301,302,303,311	2,3,17,21		1	1	-
(65,66)	301,302,303,311	6,12,20,21,22		1	1	-
(67,68)	301,302,303,311	2,3,9,21		2	1	-
(69,70)	301,302,303,311	6,7,16,19		1	1	-

TABLE D.2

THE CLASH SUB-MATRIX FOR THE TIMETABLE PROBLEM WITH NO SOLUTION DEFINED IN TABLE D.1

	Distinct activities with more than 2 resources														
ctivity	(7,8)	(11,12)	(17,18)	(47,48)	(49,50)	(51,52)	(53,54)	(55,56)	(57,58)	(59,60)	(61,62)	(63,64)	(65,66)	(67,68)	(69,70)
7,8)	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1
11,12)	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1
17,18)	1	1	0	0	1	0	1	0	1	1	1	1	0	1	0
47,48)	1	1	0	0	0	1	0	1	1	0	1	0	0	0	0
49,50)	1	1	1	0	0	1	0	1	0	1	1	0	1	0	0
51,52)	1	1	0	1	1	0	0	0	0	0	0	1	0	1	0
53,54)	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0
55,56)	1	1	0	1	1	0	1	0	1	1	0	1	0	1	0
57,58)	1	1	1	1	0	0	0	1	0	0	0	0	0	0	1
59,60)	0	0	1	0	1	0	0	1	0	0	0	0	0	0	1
61,62)	1	1	1	1	1	0	0	0	0	0	0	1	0	1	1
63,64)	1	1	1	0	0	1	0	1	0	0	1	0	0	0	1
65,66)	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
67,68)	1	1	1	0	0	1	0	1	0	0	1	0	0	0	1
69,70)	1	1	0	0	0	0	0	0	1	1	1	1	0	1	0

ABLE D.2 (Contd.)

Distinct activities with more than 2 resources

ctivity		(7,8)	(11,12)	(17,18)	(47,48)	(49,50)	(51,52)	(53,54)	(55,56)	(57,58)	(59,60)	(61,62)	(63,64)	(65,66)	(67,68)	(69,70)
Teacher Resources	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	2	1	1	1	1	0	1	1	1	0	1	1	0	1	0	1
	3	1	1	1	1	1	1	0	1	1	1	1	0	1	0	1
	4	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1
	5	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	6	1	1	0	0	1	1	1	1	1	1	1	1	0	1	0
	7	1	1	0	1	1	0	1	0	1	1	1	1	1	1	0
	8	1	1	1	1	1	0	1	0	1	1	0	1	1	1	1
	9	1	1	1	1	1	1	1	1	0	1	1	1	1	0	1
	10	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
	11	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
	12	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1
	13	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
	14	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1
	15	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1
	16	1	1	0	1	1	1	1	0	1	1	1	1	1	1	0

ABLE D.2 (Contd.)

ctivity	(7,8)	(11,12)	(17,18)	(47,48)	(49,50)	(51,52)	(53,54)	(55,56)	(57,58)	(59,60)	(61,62)	(63,64)	(65,66)	(67,68)	(69,70)
17	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1
18	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1
19	1	1	1	0	0	1	0	1	1	1	1	1	1	1	0
20	1	1	1	1	1	0	1	0	1	1	0	1	0	1	1
21	1	1	1	0	1	1	1	1	1	0	1	0	0	0	1
22	1	1	1	1	1	0	0	1	0	0	0	1	0	1	1
23	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1
lumber of Clashes	4	4	9	13	11	13	17	10	13	18	11	12	17	12	12

Note that activity (53,54) has only 4 available activities that may be allocated to the same time-period as (53,54), i.e. there are only 4 non zero elements in the row associated with (53,54). Also note that all 3rd year level activities are from activities (59,60) to (69,70) and none are available for allocation together with (53,54), i.e.. all zero elements in row (53,54) for columns associated with (59,60) to (69,70). Hence no feasible allocation for a solution satisfying the timetable constraints can be determined and thus the problem as defined has no solution.

TABLE D.3

A TABLE OF THE REQUIRED TEACHER RESOURCE LOADS FOR THE TIMETABLE PROBLEM DEFINED IN TABLE D.1 THAT HAS NO SOLUTION.

Teacher Code	Required Load
01	1 : 7
02	6 : 7
03	6 : 7
04	4 : 7
05	5 : 7
06	6 : 7
07	4 : 7
08*	3 : 3
09*	4 : 6
10	5 : 7
11	6 : 7
12	5 : 7
13	2 : 7
14	6 : 7
15	6 : 7
16*	3 : 5
17*	2 : 6
18	6 : 7
19	6 : 7
20*	4 : 6

TABLE D.3 (Contd.)

Teacher Code	Required Load
21	6 : 7
22	7 : 7
23	4 : 7

* a part-time teachers available for only restricted time-periods as defined in Table D.4.

TABLE D.4

Resource Code	Time-periods not available
17	1
8	4,5,6,7
9	7
20	7
16	4,5

The time-periods for resources indicated that are not available for allocation in the solution, due to resource commitments outside of the timetable problem. e.g. resource 8 is a part-time teacher, only available for the first 3 time-periods.

BIBLIOGRAPHY

1. ALLEN, D. W. (1966) — "Stanfords Computer System gives Scheduling freedom to 26 districts." Nation's Schools. 77(3), May 1966, pp. 124-125.

2. APPLEBY, S. J., BLAKE, D. V., NEWMAN, E. A. (1961) — "Techniques for producing school time-tables on a computer and their application to other scheduling problems." The Computer Journal 3(5), pp. 237-245.

3. BARRACLOUGH, E. D. (1965) — "The application of a digital computer to the construction of timetables." The Computer Journal 7(11), pp. 136-146. Also Chpt. 6, M.Sc. Thesis, University of Newcastle-upon-Tyne.

4. BERGE, C. (1971) — "Principles of Combinatorics." Maths in Science and Engineering. Vol. 72. (Academic Press)

5. BERGHUIS, J., VAN DER HEIDEN, A. J., BAKKER, R. (1964) — "The preparation of school timetables by electronic computer." BIT (Nordesk Tidskrift fir Informations Behandling) Vol. 4, pp. 106-114.

6. BLACKFORD, G. A. (1964) — "Honour Classes and individual time-tables." Canadian Education and Research Digest, 4(1), pp.17-24.

7. BUSH, R. N. (1964) — "Decision for the Principal : Hand or computer scheduling." The Bulletin of the National Association of Secondary School Principles, 48(291), pp. 141-146.

8. BUSH, R. N., CAFFREY, J. G., OAKFORD, R. V., ALLEN, D. W. (1961)
"Using machines to make the High-school Schedule."
The School Review, 69(1), pp. 58-59.

9. CISMA, J. (1965)
"Investigations on a timetable problem."
Ph.D. thesis, University of Toronto, Canada.

10. CISMA, J. and GOTLIEB, C. C. (1964)
"Tests on a computer method for constructing school timetables."
Comm. A.C.M., 7(3), pp.160-163.

11. CLACHER, R. M. (1969)
"Solving the general timetable problem."
Publication of I.C.L., Brisbane, Australia.

12. CLAGUE, D. J. (1969)
"Timetables : A computer solution to a recurring problem."
Australian Journal of Education, Vol. 13, October, pp. 323-337.

13. DE WARRA, D. (1972)
"Construction of school timetables by Flow methods."
Canadian Journal of Operational Research and Information Processing, pp. 12-22.

14. DUNCAN, A. K. (1965)
"Further results on computer construction of school timetables."
Comm. A.C.M., 8(1), p. 72.

15. DUNCAN, A. K. (1966)
"Computer construction of High school timetables."
Proceedings of third Australian Computer Conference, 1966 Adelaide, South Australia, pp. 363-365.

16. FLANAGAN, S. (1961)
"Machine Programming at Huntington Beach High School."
(California) Journal of Secondary Education, Vol. 36, pp. 371-373.

17. G.A.S.P. (1964) — "Generalized Academic Simulator Program : School Scheduling by Computer."
A report from the Educational Facilities Laboratories, New York.

18. GOTLIEB, C. C. (1963) — "The construction of class-teacher timetables."
Proceedings of IFIP Congress, 1962, Munich, North Holland Publication Co., pp. 73-77.

19. HALL, M. (Jnr.) (1967) — "Combinatorial Theory."
Publisher, Blaisdell.

20. HALL, M. (Jnr.) (1956) — "An algorithm for distinct representatives."
American Mathematical Monthly, Vol. 63, December, pp. 716-717.

21. HALL, P. (1935) — "On representatives of subsets."
Journal of London, Math. Society. 10(1935), pp. 26-30.

22. HALMOS, P. R. and VAUGHAN, H. E. (1950) — "The Marriage Problem."
American Journal of Mathematics, 72(1950), pp. 214-215.

23. HARARY, F. (1969) — "Graph Theory."
Addison-Wesley, publishers.

24. HEMMERLING, M. B. (1972) — "A computer timetable solution for the South Australian Secondary Schools."
Proceedings of the 5th Australian Computer Conference, May 1972, Brisbane, pp. 437-440.

25. HONOUR, V. G. (1970) — "The Computer Timetable."
Quest 7(October), Queensland Department of Education.

26. JOHNSTON, H. C. and WOLFENDEN, K. (1968) "Computer aided construction of school timetables." Proceedings of IFIP Congress, 1968.

27. LAWRIE, N. L. (1968) "School timetabling by computer." Proceedings of Glasgow Programmed Learning Conference, April, 1968.

28. LEWIS, C. F. (1961) "The School Timetable." Cambridge University Press, Cambridge.

29. LIONS, J. (1966) "Matrix reduction using the Hungarian method for the generation of school timetables." Comm. A.C.M., 9(5), pp. 349-354.

30. LIONS, J. (1966) "A counter-example for Gotlieb's method for the construction of school timetables." Comm. A.C.M., 9(9), p. 697.

31. LIONS, J. (1967) "The construction of timetables for Ontario schools using a computer." Paper presented to the Ad Hoc Group on Efficiency in Resource Utilization in Education. O.E.C.D., Paris.

32. LIONS, J. (1967) "The Ontario School Scheduling Program." The Computer Journal, 10(1), pp. 14-21.

33. LIONS, J. (1968) "A generalisation of a method for the construction of a class-teacher timetable." Proceedings IFIP Congress, 1968 (Edinburgh).

34. LIU, C. (1968) "Introduction to Combinatorial Analysis." New York, McGraw-Hill.

35. MIHOC, G. H. and BALAS, E. (1965) "The Problem of Optimal Timetables." Revue Roumaine de Mathematiques Pures and Appliquees, 10(4), pp. 381-388.

36. MIRSKY, L. (ND) "Transversal Theory." Mathematics in Science and Engineering. Vol. 75. (Academic Press).

37. MIRSKY, L. and PERFECT, M. (1966) "Systems of Representatives." Journal of Math. Analysis and Applications, 15(1966), p. 520.

38. OAKFORD, R. V. (1961) "Machine assistance for constructing High school schedules." Journal of Secondary Education, October 1961.

39. OAKFORD, R. V., ALLEN, D. W., CHATTERTON, L. A. (1966-67) "School Scheduling - Practice and Theory." Journal of Education Data Processing, 4(1), pp. 16-50.

40. OLIVER, I. (1968) "Tree-searching School Timetables." The Australian Computer Journal, 1(3), pp. 153-157.

41. ORD-SMITH, R. J. (1970) "Generation of permutation sequences : Part 1." The Computer Journal, 13(2), pp. 152-155.

42. PORTUGAL, V. M. (1967) "Application of a combinatorial method for solving the problem of constucting a timetable." Cybernetics, 3(4), pp. 81-82.

43. RAGHAVARAO, D. (1971) "Construction of Combinatorial problems in design of experiments." Wiley.

44. RIORDAN, J. (1958)

"Introduction to Combinatorial Analysis."

Wiley.

45. ROBINSON, A. E. (1945)

"The one man timetable."

The School, Secondary Edition (Ontario College of Education), Vol. 34, pp. 304-308.

46. RYAN, D. M. (1969)

"A survey of computer-aided timetable construction."

Proceedings of the weekend seminar on the role of the computer in secondary schools, Adelaide, August 15-17, 1969, pp. 158-165.

47. RYSER, H. J. (1963)

"Combinatorial Mathematics."

The Carcus Mathematical Monographs No. 14, Mathematical Association of America, New York.

48. S.S.S.S. (1968)

"An introduction to modular scheduling with the Stanford School Scheduling System."

A report from Stanford University by D. Sharpes.

49. S.S.S.S. (1968)

"Computer Scheduling Educational Reform." using S ."

An SPL/EFL report of Stanford University by R. Bergquist.

50. S.S.S.S. (1968)

Extracts from the Final report on "Flexibility for Vocational Education through computer Scheduling."

A project of the School of Education, Stanford University.

51. SEFTON, I. M. (1969)

"Generation of school timetables by computer."

Technical report No. 57, August 1969. Basser Computing Department, University of Sydney.

52. SEFTON, I. M. (1970) "Generation of academic timetables by computer." M.Sc. Thesis, University of Sydney.

53. SHERMAN, G. R. (1963) "A combinatorial problem arising from scheduling university classes." Journal of Tennessee Academy of Science, Vol. 38, pp. 115-117.

54. STOCKMAN, J.W. (1964) "A guide to automated class scheduling." Data Processing, 6(1964), pp. 30-33.

55. WELLS, M. B. (N.D.) "Elements of Combinatorial Computing." Pergamon, publishers.

56. WELSH, V. N. (1971) "Combinatorial Mathematics and its applications." Academic Press.

57. WELTON, G. (1961) "Data Processing and the school schedule : A.Burroughs approach." Journal of Secondary Education, 36(6), pp. 382-384.

58. WULFF, B. H. (1961) "Data Processing for student scheduling : An IBM approach." Journal of Secondary Education, 36(10), pp. 380-381.

59. "Our Secondary Schools", Adelaide, (Government Printer).