An improvement algorithm for school timetabling

R. J. Aust

Department of Applied Mathematics, University of Adelaide, Adelaide, South Australia 5001, Australia

This paper describes an algorithm for improving infeasible timetables. It reduces the teaching resource, break and spread infeasibilities in three stages. The first of these involves the solution of a series of capacitated transportation problems and is used when an initial timetable is not given. Under the limitations imposed by actual timetables this stage may be simplified. The other two stages each involve solving a series of small integer programming problems which will be called interchange problems, and they determine the movement of entries within the timetable. Such an algorithm can handle fixed and block meetings, sets, allocation of special rooms and variable teacher availability while producing an acceptable spread of repeated meetings.

(Received April 1975)

1. Introduction

There have been many algorithms suggested for producing secondary school timetables using a computer. They may be divided broadly into two categories, assignment algorithms and improvement algorithms. The aim of all of these algorithms is to schedule a given set of meetings in a limited set of time slots called periods so that certain logical and other imposed constraints are satisfied.

The assignment algorithms seek to allocate meetings or groups of meetings to periods in the timetable until all meetings are accommodated without any of the constraints being violated. The distinguishing feature of these algorithms is that at no stage do they accept the violation of a constraint. Their rules for choosing the next meeting to be assigned and its assignment position are selected to avoid such situations. However, when they do occur, the algorithm may either back track on previous assignments, or modify a constraint or refuse to assign the particular meeting. With this modus operandi established, the important theoretical question is whether, given a set of meetings which have been assigned already, the remaining meetings can be assigned. For the case when a meeting is of only one period's duration and involves only one teacher and class, there is the work of Csima and Gotlieb (1964) using the Hall condition on a three-dimensional availability matrix. Application of these ideas have been given by Lions (1967) with further computational improvements (1971). The difficulty inherent in the counter-examples to the sufficiency of the conditions was identified by Dempster (1968) and resolved in his graph recolouring algorithms, (1971). Hemmerling (1972) more recently has considered the case when there are sets and block meetings. The approach is enumerative in nature. The Hall conditions and certain look ahead features are used to reduce the number of possibilities considered. As with most algorithms based on the Hall conditions, the problem must be first divided into daily problems to reduce computation time and this too provides the means for spreading repeated meetings.

On the other hand if one considers a schedule in which all the meetings have been inserted but for which constraint violation exists, then an algorithm which attempts to resolve these conflicts can be considered an improvement algorithm. For example, Lawrie's integer programming model (1969) for solving problems expressed in terms of layouts can be considered an improvement algorithm. Lawrie's model only admits meetings of one period's duration but allows setting through the use of year group layouts. There are at least three further aspects of the practical timetabling problem not included in Lawrie's model. These are that some meetings must occur at fixed times. Some meetings require a block of consecutive periods which must not span any of the natural breaks in the timetable such as lunch or between days. Finally where meetings are required to be repeated (other than in blocks) they should be spread evenly through the timetable.

This paper is concerned with producing a timetable in which these additional practical considerations are included. The procedure is based on the observation that it is easy to construct a timetable in which the correct number of lines (as defined by Lawrie) appear, but, in general, it will not be feasible. There are three sources of infeasibilities in these timetables. Firstly, the teaching resources may be exceeded in a period; secondly, a block may span a break leading to a break infeasibility; thirdly, a particular line may appear too often in a day giving a spread infeasibility.

The algorithm tackles the problem in three stages.

1. Constructs a break feasible timetable.
2. Reduces teacher infeasibilities while remaining break feasible.
3. Improves spread characteristics while remaining break feasible and not increasing the resource infeasibility.

The first stage constructs the initial timetable and it is successful whenever there is at least one break feasible timetable. The second and third stages are performed by a heuristic improvement algorithm which reduces infeasibilities by interchanging entries within the timetable. Given an infeasible part of the timetable, the improvement algorithm determines the interchange to be made by generating and then solving a small integer programme which will be called an interchange problem. The process of selecting, generating and solving an interchange problem will be called a minor iteration. Looping in the exchange of entries in the timetable is avoided by insisting that each interchange used to alter the timetable must not only be an optimal solution to an interchange problem but it must also make a positive contribution to the reduction of the infeasibilities. Those interchange problems which result in changes to the timetable are called major iterations and a count of these indicates the extent to which the initial timetable has had to be altered.

2. Model description

Let n be the number of year groups in the school, p be the number of periods in the timetable and s the number of subjects in the layouts.

The layout for year group i has a matrix structure in which the rows represent subjects and the columns represent the different subject class collections taught at the same time. The elements of the matrix are the number of teachers required by subject in each subject class collection. The columns of this matrix will be called layout vectors.

To extend Lawrie's description of layouts a set of requirements R is defined. It states how many times each layout vector should be used, any periods which require a specific layout vector, and the block structure for layout vectors. Blocks may be composed of different layout vectors thus enabling the repetition of a subject class in consecutive periods without needing to repeat the same collection of subject classes.

The layout vectors from all of the year groups are used to form a three dimensional matrix of size $n \times p \times s$ which will be called a timetable matrix. The number and manner in which the layout vectors appear in the timetable matrix, is described by R.

By considering planes parallel to the faces of the timetable matrix (**Fig. 1**) it may be viewed as:

1. s $n \times p$ matrices which will be called the subject matrices. They contain all the requirements for a specific subject.
2. p $s \times n$ matrices which will be called the period matrices because they each represent one period.
3. n $s \times p$ matrices which will be called the year group matrices.

All the layout vectors from a given year group must lie in the same year group matrix and, including repetitions defined by R, must exactly fill it. The only permissible transformation on the timetable matrix is the interchange of layout vectors within a year group matrix.

For an arbitrary period j, let $P_j = (a_{ki})$ be its period matrix. The row sum $\sum_{i=1}^{n} a_{ki}$ represents the number of reachers required for subject k in period j. If for each k this sum is less than or equal to the number of teachers available, t_k, then P_j is said to be feasible in teaching resources.

Define

$$g_k(j) = \sum_{i=1}^{n} a_{ki} - t_k$$

p_j is feasible if $g_k(j) \leqslant 0$. A period matrix feasible in teaching resources corresponds to an arrangement in Lawrie's terminology.

To introduce a break structure into the timetable, an extra component is added to each layout vector. The requirement a_{s+1i} in this component is

(*a*) 0 for a single period

(*b*) 0 for the first period of a double, 1 for the second

(*c*) 0 for the first period of a triple, 1 for the last two.

The availability in this component $u(j)$ is defined to be 0 if period j follows a break and is n otherwise.

Define

$$h(j) = \sum_{i=1}^{n} a_{s+1i} - u(j)$$

If for each j, $h(j)$ is less than or equal to zero, then the timetable is said to be break feasible.

To characterise the spread of the layout vectors, the allowable number of repetitions per day is calculated for every layout vector.

Number of repetitions allowed =
maximum of {number of largest block/number of days}
× block size and
{number of periods used/number of days}

where {} indicate the smallest integer greater than.

If for each day the actual number of repetitions of the layout vectors is less than or equal to the allowable number then the timetable is said to be spread feasible.

A timetable matrix is said to be feasible if it is feasible with respect to teaching resources, break and spread. The problem of producing Lawrie's outline timetable becomes one of finding a feasible timetable matrix given an initial timetable matrix satisfying R.

3. Break feasible timetable

A block of size l is an ordered set of l layout vectors which must be placed in l consecutive periods. Consider one year group and let $c(l)$ be the number of blocks of size l and m be the size of the largest block. Because every period has to contain a layout vector, we require

$$\sum_{l=1}^{m} lc(l) = p \ .$$

The constraint on allocating blocks to periods is that some of the blocks of size 1 are assigned already to periods from which they cannot be moved. The problem is to assign the remaining blocks to periods so that no block has to be split because of the assignments already made. The condition that blocks should not span breaks is incorporated into this model by replacing breaks by generated fixed single blocks.

In any solution the fixed blocks must be in the periods designated, therefore only the gaps formed after they have been assigned, need be considered. Let q be the number of gaps formed and $d(r)$ be the number of periods in the rth gap. It is convenient to redefine $c(1)$ to be the number of blocks of size 1 which are not fixed. Then

$$\sum_{l=1}^{m} lc(l) = \sum_{r=1}^{q} d(r) \ .$$

Define $y(l, r)$ to be the number of blocks of size l assigned to the rth gap. The problem is to find an integer solution to the following set of constraints.

$$\sum_{l=1}^{m} ly(l, r) = d(r) \quad r = 1, 2, \ldots, q$$

$$\sum_{r=1}^{q} y(l, r) = c(l \quad l = 1, 2, \ldots, m$$

$$y(l, r) \geqslant 0 \text{ and integer} \ .$$

By multiplying the last m constraints by the appropriate l the quantity $ly(l, r)$ may be replaced by $z(l, r)$. The problem so formed is the classical balanced transporation problem which has a solution in integers. A solution to the original problem requires that $z(l, r)$ be divisible by l. This may be achieved by a branch and bound style enumeration in which if a particular $z(l, r)$ is not divisible by l, then two new problems are created. In one an upper bound of the largest integer multiple of l less than $z(l, r)$ is added and in the other a lower bound of the smallest integer multiple of l greater than $z(l, r)$ is added. The lower bound may be reduced to zero by translation without affecting the process because the amount by which it is translated is divisible by l.

In general the above process would require the solution of a series of capacitated transportation problems. In practical problems the introduction of the generated break fixed periods enables one to assume that $d(r) \leqslant 5$. With this condition and by restricting the maximum block size to three one can use the following algorithm to generate break feasible timetables. Each step represents an assignment pass in which every gap is scanned.

1. Blocks of size three are inserted in gaps of size three or five. In the latter case a gap of size two is left.
2. Any remaining blocks of size three are inserted into gaps of size four. If any blocks of size three remain unplaced after this step then no break feasible timetable exists.
3. Blocks of size two are inserted so that not more than one gap of size one is created in any existing gap. If any blocks of size two remain unplaced then no break feasible timetable exists.
4. The remaining gaps are filled with the remaining blocks of size 1.

To prove that this algorithm will construct a break feasible timetable when one exists, one notes that all of the new gaps of length one introduced in steps 2 and 3 are forced. Therefore if there are not sufficient single blocks at step 4, no feasible timetable exists.

4. Interchange problems

In stages two and three of the algorithm, teaching resource and period spread infeasibilities are reduced while maintaining break feasibility. This is achieved by interchanging layout vectors within year groups. The blocks of size three are fixed after stage one and therefore only the movement of blocks of size two and one will be considered.

Consider an interchange between period matrices in periods $j(1)$ and $j(2)$. Because of the presence of blocks of size two, this could also affect periods $j(1) \pm 1$ and $j(2) \pm 1$. The allowable interchanges will be restricted to those which influence periods $j(1)$, $j(1) - 1$, $j(2)$, $j(2) - 1$. If either $j(1)$ or $j(2)$ is period one then $j(1) + 1$ or $j(2) + 1$ is used. There are five possible configurations which could be found in the year groups of each of these pairs of periods (**Fig. 2**(*a*)). This leads

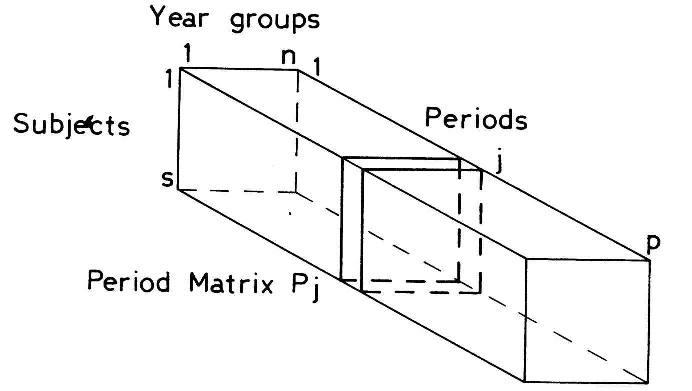

Fig. 1 Timetable matrix

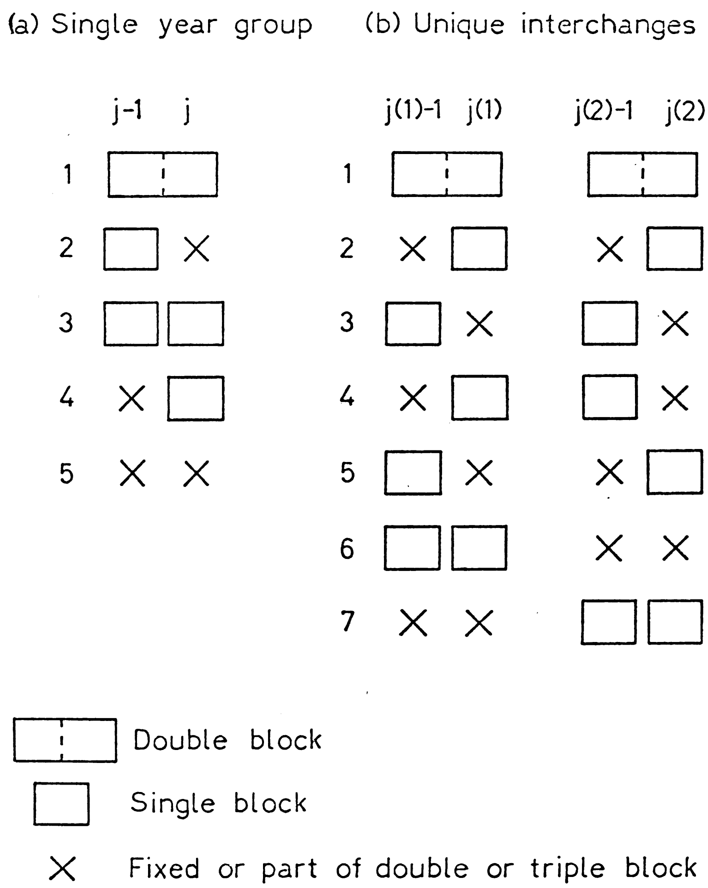

Fig. 2 Layout vector configurations

to 25 possible combinations of which seven define a unique interchange (Fig. 2(*b*)), seven enable a choice of movements and 11 do not give an allowable interchange. Where more than one movement is possible, any one may be chosen.

If all the layout vectors in period $j(1)$ are blocks of size 1 then the interchange movements allowed are restricted to be between only blocks of size 1 in $j(1)$ and $j(2)$. In this case the interchange between period matrix $P_{j(1)} = (a_{ki})$ and $P_{j(2)} = (k_{ki})$ may be defined as a transformation:

$$P_{j(1)} \rightarrow P'_{j(1)};\, P_{j(2)} \rightarrow P'_{j(2)}$$

where

$$\begin{aligned} P'_{j(1)} &= (a'_{ki}) \text{ with } a'_{ki} = a_{ki}(1 - x_i) + b_{ki}x_i \qquad k = 1, s \\ P'_{j(2)} &= (b'_{ki}) \qquad b'_{ki} = a_{ki}x_i \qquad + b_{ki}(1 - x_i) \quad k = 1, s \\ & \qquad x_i = 0 \text{ or } 1 \quad i = 1, n\ . \end{aligned}$$

If x_i equals one then the layout vectors of year group i are interchanged in period $j(1)$ and $j(2)$. Under the transformation $g_k(j(1)) \rightarrow g'_k(j(1))$ and $g_k(j(2)) \rightarrow g'_k(j(2))$. They satisfy the equation

$$g_k(j(1)) + g_k(j(2)) = g'_k(j(1)) + g'_k(j(2)) \qquad (4.1)$$

This implies $\sum_{j=1}^{p} g_k(j)$ is fixed for each k and therefore a necessary condition for feasibility is that this sum be non-positive.

In a similar way one can give expressions using one zero-one variable for each of the interchanges in Fig. 2(*b*). These can be used to derive expressions for $g_k(j)$ in each of the four periods.

The improvement in a subject made by an interchange is the amount by which the infeasibilities in that subject are reduced. A method of choosing the interchange would be to find the one which maximises the sum of the improvements over all subjects. A formulation of this problem is given in Aust (1973). Based on experimental results in the same source, an alternative objective is used which approximates this objective. It has the advantage that only as many variables as there are year groups are required in the formulation. This gives a shorter solution time per interchange and it was found that the number of interchanges required was not increased significantly.

For each k define

$$\begin{aligned} j_k &= j(1), \bar{j_k} = j(2) \text{ if } g_k(j(1)) \geqslant g_k(j(2)) \\ j_k &= j(2), \bar{j_k} = j(1) \text{ if } g_k(j(2)) > g_k(j(1)) \end{aligned}$$

From equation (4.1) it can be seen that if $g'_k(j_k)$ is to be less than $g_k(j_k)$ then $g'_k(\bar{j_k})$ will be greater than $g_k(\bar{j_k})$. To control the reduction in $g'_k(j_k)$ the following constraint is imposed.

$$\begin{aligned} g'_k(\bar{j_k}) &\leqslant 0 \text{ if } g_k(\bar{j_k}) \leqslant 0 \qquad (4.2) \\ g'_k(j_k) &\geqslant g'_k(\bar{j_k}) \text{ otherwise}\ . \end{aligned}$$

Because there is a trade off between subjects a greater overall improvement may be made if the $g'_k(j_k)$ in certain subjects are allowed to increase. To achieve this without loosing feasibility in an already feasible period matrix the constraint

$$g'_k(j_k) \leqslant 0 \text{ if } g_k(j_k) \leqslant 0 \qquad (4.3)$$

is introduced.

Subject to constraints (4.2) and (4.3) the interchange selected is that which maximises $\sum_{k=1}^{s} (g_k(j_k) - g'_k(j_k))d_k$ where $d_k = 0$ if $g_k(j_k) \leqslant 0$ and is 1 otherwise. The quantity $g_k(j_k) - g'_k(j_k)$ measures the amount by which the larger $g_k(j)$ before the interchange is reduced. This does not always give the improvement in the subject but it does have a positive value whenever there is a positive improvement. The coefficient d_k is used to remove the effect of $g_k(j)$ movements in feasible subjects of the period matrices and constraints (4.2) and (4.3) ensure they remain feasible. Where there are four periods involved one defines similar constraints for the pair $j(1) - 1$ and $j(2) - 1$.

Constraints (4.2) and (4.3) relate to teaching resources. One can define analogues of these in terms of $h(j)$ to maintain break feasibility.

The remaining constraints on interchanges relate to period spread and are introduced in stage 3 of the algorithm. The year groups are independent with respect to this factor and may be considered separately. Interchanges involve only two layout vectors except for those of type 1 which may involve four. Constraints are constructed analogously to those for teaching resources, except now they relate to changes in the number of layout vectors of a particular type in a day. Quantities similar to the $g_k(j)$ may be defined for each day as the actual number of layout vectors minus the allowable number.

In two cases of type 1 the immediate analogues of the pair wise generation of constraints will give incorrect coefficients. This is because the movements between periods $j(1)$, $j(2)$ and $j(1) - 1$, $j(2) - 1$ are not independent. These situations are readily identified and the coefficients changed accordingly.

5. Interchange problem solution

The interchange problems are solved by a modified complete enumeration procedure. Because of the large number of problems to be solved consideration must be given both to the method of generation and the method of solution.

The period spread constraints are generated first. Those year groups which do not satisfy their respective constraints may be eliminated. Also if the layout vectors to be interchanged are identical then the year group may be eliminated.

Initially all possible solutions are potentially feasible and a list of the numbers 1 to $2^n - 1$ is made where n is number of year groups not eliminated. The binary representation of these numbers will give the appropriate x values. The zero or 'do nothing' interchange is always feasible and therefore is not considered. The teaching resource constraints are generated subject by subject and those solutions which are no longer feasible are removed from the list. A similar procedure is used for the break constraints. If at any point the list is exhausted then this implies that the only feasible solution is the zero interchange and another problem is tried.

The process of generating constraints also generates the coefficients for the objective function which is their weighted sum. The final step is to find which of the remaining solutions has the highest value. If this is positive then that interchange is made.

The value of this procedure is that it is possible to show that there is only one feasible solution without generating the complete problem or even the objective function. This is important since at least 90 per cent of the interchange problems are of this type. It is also practicable because the value of n is small ($\leqslant 7$).

6. Period matrix selection

Two different strategies are used to pick the period matrices for stages two and three. In stage two, two lists of the period matrices are formed, one in which they are put in descending order according to the sum of their positive $g_k(j)$ and in the other in descending order according to the sum of their negative $g_k(j)$.

One period matrix is selected from the top of the first list and the other from the bottom of the second. If they do not produce a positive valued interchange the next to bottom period on the second list is chosen. This process continues until either a positive valued interchange is produced or all of the period matrices in the second list have been tried with the first on the other list. In the former case, the interchange is made and the lists updated; in the latter, the second period matrix of the first list is chosen and the first period matrix is said to require modified selection.

If when a positive valued interchange occurs, there is at least one period matrix which requires modified selection then the next interchanges tried are those between the modified selection period matrix and the two/four new period matrices. This eliminates the repetition of a large number of zero interchanges that would otherwise be tried. If further positive valued interchanges are made during modified selection then steps are taken to keep track of any period matrices which still require modified selection and those which are newly created.

The process may terminate in one of two ways. Firstly if the sum of the positive $g_k(j)$ of the period matrix at the top of the first list is zero then the timetable matrix is feasible with respect to teaching resources. Secondly if the sum of the positive $g_k(j)$ of the period matrix currently being used from the first list is non-positive, then no further positive valued interchanges can be made. This because all further interchange problems, which would be tried, would have objective functions identically equal to zero from the effect of the flagging coefficients, d_k. Though the modified selection period matrices also appear on the second list they are never selected from that list as all possible interchange problems involving them have been tried. If the process terminates in the first way, stage two is said to have been successful; if it terminates in the second way, stage two is said to have been completed.

In stage three only one list in which the days are ordered according to the sum of their infeasibilities in period spread is used. The day first on the list is chosen and its period matrices are examined in turn to find one with a layout vector contributing to the infeasibility. This period matrix is tried with every other period matrix in the remaining days until either a positive valued interchange is found or all have been tried. In the former case the interchange is made and the necessary book-keeping performed and in the latter the next suitable period matrix found within that day. This continues until either the first day on the list is feasible or the next day selected is feasible. In the former case, stage three is said to have been successful and in the latter it is said to have been completed.

During stage three it is possible to make interchanges between period matrices which are feasible with respect to teaching resources. Therefore if stage two was not successful it may be possible to further reduce the remaining teaching resource infeasibilities by returning to stage two. This may be done with or without using period spread constraints.

7. Results

The algorithm has been programmed in FORTRAN and has principally been run on a CDC 6400 computer. The program has been dimensioned to take problems with up to seven year groups, 50 periods and 20 subjects. It takes approximately 28 CP seconds to compile, and the compiled code requires 65000 (octal) words of core to load. The program will either generate an initial matrix as described in Section 3 or will accept a given initial matrix. The latter facility allows the reprocessing of manual alterations to a previously constructed timetable.

The initial testing of the algorithm's performance has been done on the same problems as used by Lawrie (1969). He considered four schools two of which had feasible solutions and two of which did not. These problems only contain blocks of size one. The results are given in **Table 1.** The infeasibilities are stated in two parts—the first indicating the number and the second the sum.

The current implementation repeats stages 2 and 3. If the initial stage 2 was not successful then the interchanges made in stage 3 may lead to further improvements in teaching resource infeasibilities when reconsidered by a second stage 2.

For both feasible problems the initial stage 2 was successful.

Table 1

School	*B*	*D*	*G*	*W*
Initial teaching infeasibilities	55/144	93/320	46/129	43/113
End 1st stage 2/3				
Teaching infeasibilities	0/0	16/19	0/0	10/10
Spread infeasibilities	6/6	15/17	0/0	1/1
Major iterations	58	92	44	82
Minor iterations	922	4031	243	3000
End 2nd Stage 2/3				
Teaching infeasibilities	0/0	15/18	0/0	9/9
Spread infeasibilities	6/6	9/10	0/0	0/0
Major iterations	0	8	0	4
Minor iterations	384	2697	0	521
Total time (CP seconds)	12·81	49·97	4·44	21·84

Table 2

School	*B with 19 doubles*		*G with 16 doubles*	
	Run 1	Run 2	Run 1	Run 2
Initial teaching infeasibilities	69/276	1/2	43/113	1/1
End 1st Stage 2/3				
Teaching infeasibilities	3/4	0/0	1/4	1/1
Spread infeasibilities	3/3	1/1	2/2	2/2
Major iterations	81	3	37	0
Minor iterations	2827	187	1055	161
End 2nd Stage 2/3				
Teaching infeasibilities	3/4	0/0	1/4	1/1
Spread infeasibilities	3/3	1/1	2/2	2/2
Major iterations	0	0	0	0
Minor iterations	362	96	161	161
Total time (CP seconds)	81·01	7·31	13·89	5·49

For the two infeasible problems, the program identifies which subjects should be considered for the modification of their layout vectors or requirements. For school *W* two subjects contained all the infeasibilities. Knowing which subjects to examine one can show that a lower bound on the number and sum of the infeasibilities is 8/8. For school *D* a similar analysis on three subjects yields a lower bound of 7/7. These analyses assume that the other subjects will still remain feasible when the lower bound arrangement is used. This may not be possible and so part of the gap between the lower bound and the actual value achieved may be attributed to this further interaction and part to the heuristic nature of the algorithm.

With regard to the spread feasibility, stage 3 was successful for schools *G* and *W* but not *B* and *D*. However, if one compares the distributions which give rise to these infeasibilities with Boyes' table of acceptable distributions reproduced in Clementson (1968), then one finds that in every case the number of periods placed unsatisfactorily is zero. The commonest distribution giving rise to infeasibilities is five periods spread 21110 instead of 11111.

To test the ability of the algorithm to handle problems with block requirements, 16 blocks of size two were defined for school *G* and 19 for school *B*. In both cases at the end of the first run some teaching resource infeasibilities remained. For school *G* this was then reduced by a simple manual alteration but could not be improved by further use of the algorithm. For school *B* again manual alteration gave an improvement but now a feasible solution was produced when the algorithm was reused. The results are given in **Table 2.**

School *G* with doubles has been shown to be feasible by an alternative method akin to Lawrie's original suggestion. The problem is run with all blocks initially specified as single periods. The resulting feasible period matrices are then patched together to form the required blocks. This timetable matrix is then reprocessed to improve the spread.

For this problem, the first run took 4·44 CP seconds, the patching process took about an hour by hand, the spread improvement 3·63 CP seconds. The final timetable was feasible with respect to teaching resources but had one spread infeasibility.

The disadvantage with this approach is that it may not be possible to patch the given feasible period matrices without introducing break infeasibilities. Although it was not required here, the program has the facility to reduce break infeasibilities in the same way as it reduces teaching resource infeasibilities.

8. Extensions

In the model described in Section 2, it was assumed that for every subject, the teacher availability in each period is constant. This restriction is easily overcome by introducing teacher availability as a year group with negative entries in the layout vectors. This is equivalent to rearranging the teaching resource inequality for a period

$$\text{number of teachers required} \leqslant \text{number available}$$

into

$$\text{number of teachers required} - \text{number available} \leqslant 0 \ .$$

This restores a constant (zero) right hand side. In most cases all the layout vectors of this additional year group would be fixed.

The timetable produced by the algorithm ensures that for each period there are sufficient teachers by subject to cover the requirements. This is a necessary but not sufficient condition on the additional requirement that the same teacher be assigned to each meeting of a subject class.

To assist the production of the final timetable a simple program is used to summarise the resource clashes and availabilities which result from a given set of teacher and room assignments to the subject classes. Teachers or other resources which are overutilised in a period are reallocated. In cases of difficulty particular teachers may be specified as additional components of the layout vectors.

As an extreme case one can imagine all the resources specified as additional components of the layout vectors. In this light one sees the layout vectors used by the current model as a simplification of these further components. The same interchange techniques could be applied to problems involving these larger layout vectors though it would be necessary to exploit the 0,1 nature of their elements both for storage and computation.

References

Aust, R. J. (1973). The school timetable problem, Ph.D. Thesis, University of Sussex.

Csima, J., and Gotlieb, G. C. (1964). Tests on a computer method for construction of school timetables, *CACM*, Vol. 7, No. 3, pp. 160-163.

Clementson, A. (1968). Computer Timetabling Data Manual. Published by the Local Government Operational Research Unit.

Dempster, M. A. H. (1968). On the Gotlieb-Csima timetabling algorithm, *Canad. J. Math.*, Vol. 20, pp. 103-119.

Dempster, M. A. H. (1971). Two algorithms for the timetable problem, *Combinatorial Mathematics and its applications*, Ed. D. J. A. Welsh, Academic Press.

Hemmerling, M. B. (1972). A Computer timetable solution for the South Australian secondary schools, *Proc. 5th Australian Computer Conference*, Brisbane, pp. 437-440.

Lawrie, N. L. (1969). An integer linear programming model of a school timetabling problem, *The Computer Journal*, Vol. 12, pp. 307-316.

Lions, J. (1967). The Ontario school scheduling program, *The Computer Journal*, Vol. 10, pp. 14-21.

Lions, J. (1971). Some results concerning the reduction of binary matrices, *JACM*, Vol. 18, pp. 424-430.