converted to unindexed variables which in turn become the indexes of the IN(J)'s used in the loop nest.

This routine, together with the nest of loops and other routines, is set in a do loop with index K. K is transformed from decimal to a binary number, KBIN and this binary number is the input to the routine using the methods described by Weldon and Baker [4]. It represents (in reverse order) the source for which IP(J)'s are being computed. A numeral 1 in the binary indicates the corresponding letter is in the source, and a 0 indicates that the corresponding letter is absent from the source.

In the routine (Fig. 2), N is increased whenever a digit is tested, but J is increased only when the routine crosses the nexus of the figure 8. As long as a series of letters continues to be present or continues to be absent in the source letters, the routine loops on one side only and no IP(J) is produced. Thus the routine of Figure 2 executes the five rules set forth above, and produces the control constants for the nest of do loops for each source as called for by K.

The routine is written in FORTRAN with AUTOCODER inserts for convenience and compactness. It could be written in straight AUTOCODER, or it could be written in straight FORTRAN by using one word for each digit in the binary number.

Using this routine and other routines reported by Weldon and Baker [4], we have developed a program that will handle factorial designs from 1 to 10 factors. For a five-way analysis of variance, computing time on the IBM 7072 is between 2 and 3 minutes; for a ten-way analysis the computing time is approximately a half hour. We stop at 10 factors because it is the number of digits in a computer word and also a practical upper bound for current needs. There is no limit to the number of factors except computer space and time, and the needs of users.

RECEIVED NOVEMBER, 1963

REFERENCES

1. HARTLEY, H. O. Analysis of variance. In *Mathematical Methods for Digital Computers*, Ralston, A. and Wilf, H. S. (Eds.), Ch. 10, Wiley, New York, 1960.
2. GARBER, M. J. Addressing an array Yi in K-dimensions by FORTRAN for analysis of variance. *Comm. ACM 6* (Mar. 1963), 100.
3. MCCRACKEN, D. D. *A Guide to FORTRAN Programming.* Wiley, New York, 1961.
4. WELDON, R. J., AND BAKER, R. L. *Applications of Binary Number in Computer Routine.* (In press)

●

[*Editor's Note*: A referee has suggested the following comments:

"A program based on the method presented requires multiple passes through the data, once for each variance source. More efficient programs may be constructed, using index identification, where all totals are computed on a single pass through the data. This requires a multiple-modular arithmetic in order to pass from index numbers to factor level identification and vice versa. Further savings are made by building nested strings of totals whenever possible."

—D.T.]

Final Examination Scheduling

SOL BRODER
State University of New York, Stony Brook, New York

A method for scheduling final examinations to yield a minimal number of student conflicts is described. The "minimization" is achieved by repetitively evaluating a nonlinear set of equations. Imbedded in the process is a random or Monte Carlo selection of assignments. As in such heuristic techniques, the solution may not be optimum and many solutions may be found which yield locally minimal results. Computer programs are described and empirical results given.

I. Background Information

Scheduling of final examinations at large universities or similar institutions is a tedious and frustrating task. The Registrar must assign courses to blocks of exam periods which yield a minimum number of conflicts. Under normal circumstances, lacking a more formal procedure, the Registrar uses experience combined with intuition to choose an exam schedule. He then proceeds to check the schedule for conflicts in each student's program. If he has the resources he may repeat the procedure, varying the schedule in an attempt to reduce the number of conflicts. In this paper a method is discussed that formalizes the entire process, thereby eliminating the hit or miss aspect of exam scheduling.

II. Mathematical Development

1. *Definitions and Terminology*

N is the total number of courses for which exams are to be scheduled.

K is the total number of exam periods or blocks available.

$\hat{C}$ is an $(N \times N)$-matrix (c_{ij}), where c_{ij} is the total number of students who are taking both courses (and exams) i and j. $\hat{C}$ is called the matrix of potential conflicts; for this case it is evident that c_{ii} must be set to zero and that $c_{ij} = c_{ji}$.

X is the assignment matrix $(x_{ij})_{N\times K}$ in which x_{ij} is 1 if course i has its exam scheduled in period j, and is 0 if course i does not have its exam scheduled in period j. Since every course listed has one exam period, there will be one and only one nonzero entry in each row.

$\hat{Y}$ is an $(N \times N)$-matrix $(\hat{y}_{ij})_{N\times N}$ (defined in Section II.2) so that the diagonal elements $\hat{y}_{ii}$ or more exactly that the trace will measure the "goodness" of a particular assignment matrix X by showing the total number of conflicts.

σ_x is the total number of conflicts arising from a given assignment matrix X.

2. *Algebraic Form.* An algebraic expression for

"success" as a function of "potential conflict" and the "assignment schedule" is developed in this section.

The matrix XX^T is a symmetric $(N \times N)$-matrix whose ij-element is 1, if and only if, the exams for courses i and j are scheduled for the same period and is 0 otherwise. Therefore the diagonal element $\hat{y}_{ii}$ of

$$\hat{Y} = \hat{C}XX^T \tag{1}$$

gives the total number of students scheduled for the exam in course i and for at least one other exam in the same period. Since each individual conflict is counted twice (once for each course in the pair involved):

$$\sigma_x = \tfrac{1}{2} \text{ Trace } \hat{Y} = \tfrac{1}{2}\sum^{N} \hat{y}_{ii} . \tag{2}$$

From (1), the diagonal element $\hat{y}_{ii}$ of Y may be expressed as follows:

$$\hat{y}_{ii} = \sum_{j=1}^{N}\sum_{k=1}^{K} x_{ik}x_{jk}c_{ij} \tag{3}$$

which states that in determining the total number of conflicts in which course i is involved the number, c_{ij}, of potential conflicts in the ij course pair should be counted if and only if courses i and j are scheduled for the same hour.

For purposes of computation in the method utilized below it is convenient to define C as the $\hat{C}$-matrix with elements above the diagonal set equal to zero. Accordingly, eq. (1) becomes

$$Y = CXX^T \tag{1a}$$

with corresponding eqs. (2) and (3) rewritten as:

$$\sigma_x = \text{Trace } Y \tag{2a}$$

where each conflict is counted only once, and

$$y_{ii} = \sum_{j=1}^{i-1}\sum_{k=1}^{K} x_{ik}x_{jk}c_{ij} , \quad (i > 1). \tag{3a}$$

Basically, the reason for this alternate set of equations is due to the computation of y_{ii} as a function of the assignment of the previous $i-1$ courses.

The problem now is to find a matrix X that will result in the smallest total number of conflicts, σ_x. There appears to be no direct method of finding the X which results in a minimal σ using eq. (1). There are a total of K^N possibilities since each of the N rows has K different choices. Computation of σ_x for each possible X is not practical; the method is described in Section III.

III. Assignment and "Minimization" Technique

The assignment process is described in detail in this section.

1. Course 1 is assigned to block q_1, i.e., taking $x_{1q_1} = 1$. The choice of q_1 is arbitrary.

2. Courses 2 to N are assigned in order using eq. (3a), varying i accordingly. For example, course 2 is assigned considering the assignment of course 1 to block q_1. Eq. (3a) for $i = 2$ in expanded form is:

$$(c_{21}x_{11})x_{21} + (c_{21}x_{12})x_{22} + (c_{21}x_{13})x_{23} + \cdots + (c_{21}x_{1q_1})x_{2q_1} + \cdots + (c_{21}x_{1k})x_{2k} = y_{22} ,$$

or

$$(c_{21}\cdot 0)x_{21} + (c_{21}\cdot 0)x_{22} + (c_{21}\cdot 0)x_{23} + \cdots + (c_{21}\cdot 1)x_{2q_1} + \cdots + (c_{21}\cdot 0)x_{2k} = y_{22} .$$

If $c_{21} = 0$, clearly all terms within the brackets are zero. Should $c_{21} \neq 0$, then the bracketed q_1-term is not zero. Thus depending on c_{21}, a random selection is made from K or $K-1$ terms to choose one from these zero (minimal) terms. Then $x_{2q_2} = 1$ and $x_{2j} = 0$ for all $j \neq q_2$ and $y_{22} = 0$.

Course 3 is assigned considering the assignment of courses 1 and 2; again

$$(c_{31}x_{11} + c_{32}x_{21})x_{31} + (c_{31}x_{12} + c_{32}x_{22})x_{32} + (c_{31}x_{13} + c_{32}x_{23})x_{33} + \cdots + (c_{31}x_{1q_1} + c_{32}x_{2q_1})x_{3q_1} + \cdots + (c_{31}x_{1q_2} + c_{32}x_{2q_2})x_{3q_2} + \cdots + (c_{31}x_{1k} + c_{31}x_{2k})x_{3k} = y_{33} ,$$

or

$$(c_{31}\cdot 0 + c_{32}\cdot 0)x_{31} + (c_{31}\cdot 0 + c_{32}\cdot 0)x_{32} + (c_{31}\cdot 0 + c_{32}\cdot 0)x_{33} + (c_{31}\cdot 1 + c_{32}\cdot 0)x_{3q_1} + \cdots + (c_{31}\cdot 0 + c_{32}\cdot 1)x_{3q_2} + \cdots + (c_{31}\cdot 0 + c_{32}\cdot 0)x_{3k} = y_{33}$$

Each term within the bracket is again reviewed and a random selection is made to choose one from the zero (minimal) terms and thus $x_{3q_3} = 1$ and $x_{3j} = 0$ for all $j \neq q_3$.

In general the assignment of course i is made after the assignment of course 1 to course $i-1$ has been made; i.e., after the first $i-1$ rows of X have been determined.

Eq. (3a) can also be rewritten as,

$$\sum_{j=1}^{K} r_j x_{ij} = r_1x_{i1} + r_2x_{i2} + \cdots + r_kx_{iK} = y_{ii} \tag{4}$$

where r_j is the known portion of (3) and (4); i.e., $r_k = \sum_{p=1}^{i=1} c_{ip}x_{pk}$ and as noted above all the x's have been determined for the previous $i-1$ rows of X. Based on the smallest r_{j_0}, $x_{ij_0} = 1$ and $x_{ij} = 0$ for $j \neq j_0$ and the smallest possible y_{ii} results from this choice. As before, in those cases where there exists more than one r in the minimal set a selection is made on a random basis.

This method will provide a schedule from a set of "minimal" types of schedules. The repetition of this procedure with different random numbers will provide a distribution of different "minimal" schedules, from which a "best" schedule can be chosen. This procedure of selecting from nonlinear systems, assuming a statistical distribution, is a standard operations research technique.

The procedure does not guarantee the "best" solution but usually a "good" solution is found.[1]

Tabulated results from computer runs indicate that solutions could be found more quickly by rearranging the conflict matrix so that courses that are potentially in conflict with many other courses are scheduled first.

[1] See SISSON, R. L. *Progress in Operations Research, Vol. 1.* Ch. 7, Sec. 4.3, John Wiley, New York, 1961.

TABLE 1. SAMPLE STUDENT CLASS SCHEDULE

	Student									
Course	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
*k*1		x		x			x	x		
*k*2					x		x	x	x	
*k*3	x	x		x	x		x		x	x
*k*4	x			x	x		x	x	x	
*k*5	x	x		x	x			x		
*k*6		x				x				
*k*7	x		x			x				
*k*8			x							x
*k*9			x			x				
*k*10			x							

The method described in this paper has also been applied to improving the final exam schedule. The objective of the assignment procedure can also be extended to include not only minimizing the number of conflicts but also to minimizing the number of times that a student has (see Appendix): (1) two finals scheduled on the same day, (2) three finals scheduled in three successive blocks, (3) three finals scheduled in two days (four blocks), and (4) four finals scheduled in three days.

IV. Computer Program and Sample Calculation

The programs to implement this procedure have been written as a research project on the IBM 1620 and as such consist of parts with nonmatching interfaces. Preparation of these programs is presently in process for release to 1620 users. Also, conversion of them for use on the 7040, taking advantage of the larger memory and tape, is planned.

In these programs final exam scheduling is accomplished in four steps: (1) development of the conflict matrix, (2) rearrangement of the conflict matrix, (3) assignment of courses to blocks minimizing conflicts, and (4) schedule checking.

1. *Development of the Conflict Matrix.* This program takes as input the list of course numbers followed by the student class cards. Potential conflicts are tallied in matrix form; this matrix, of course, is symmetric. How-

TABLE 2. CONFLICT MATRIX (FULL MATRIX)

Course	*k*1	*k*2	*k*3	*k*4	*k*5	*k*6	*k*7	*k*8	*k*9	*k*10	*Number of entries for each row or column*
*k*1	0	2	3	3	3	1	0	0	0	0	5
*k*2	*GH** 2	0	3	4	2	0	0	0	0	0	4
*k*3	*BDG* 3	*EGI* 3	0	5	4	1	1	1	0	0	7
*k*4	*DGH* 3	*EGHI* 4	*ADEGI* 5	0	4	0	1	0	0	0	5
*k*5	*BDH* 3	*EH* 2	*ABDE* 4	*ADEH* 4	0	1	1	0	0	0	6
*k*6	*B* 1	0	*B* 1	0	*B* 1	0	1	0	1	0	5
*k*7	0	0	*A* 1	*A* 1	*A* 1	*F* 1	0	2	2	1	7
*k*8	0	0	*J* 1	0	0	0	*C* 1	0	2	1	4
*k*9	0	0	0	0	0	*F* 1	*CF* 2	*CF* 2	0	1	4
*k*10	0	0	0	0	0	0	*C* 1	*C* 1	*C* 1	0	3

* Courses (*k*2, *k*1) are taken by students *G*, *H* for an entry of 2 in this matrix.

ever in this development only the lower diagonal portion is used. An example of a given student schedule is shown in Table 1 and the corresponding conflict matrix in Table 2.

2. *Rearrangement of the Conflict Matrix.* Each row of the conflict matrix, developed and punched out in step 1, is tallied for frequency of conflict yielding a list of row (course) versus frequency of conflicts. With this new list rearranged in descending frequency order, step 1 is performed again. The net result is a rearranged matrix which provides a statistically favorable environment for the assignment process (Table 3).

3. *Assignment of Courses to Blocks Minimizing Conflicts.* The input to this program includes the number of courses, the number of blocks, and the conflict matrix read one row at a time. As each row is read a good assignment is made. The number of conflicts, if any, is also recorded. Currently, a procedure is being developed which will minimize other undesirable aspects of a given schedule, i.e., three exams in three successive blocks, three exams in four successive blocks, etc. The assignment program may be repeated until the best schedule is developed. To illustrate the procedure, using Table 3, assign these ten courses to six examination blocks. The examination schedule assignment process follows.

Step 1. Assign first course-$k7$ to block 3; thus $x_{13} = 1$; $x_{ij} = 0$ for $j \neq 3$.

Step 2. Using eq. (3a) and the results of step 1, and where $r_j = c_{21}x_{1j}$ or,

$$r_1x_{21} + r_2x_{22} + r_3x_{23} + r_4x_{24} + r_5x_{25} + r_6x_{26} = y_{22}$$
$$= (1\cdot0)x_{21} + (1\cdot0)x_{22} + (1\cdot1)x_{23} + (1\cdot0)x_{24}$$
$$+ (1\cdot0)x_{25} + (1\cdot0)x_{26} = y_{22} = \text{Min.}$$

TABLE 3. REARRANGED C-MATRIX

Course	*k*7	*k*3	*k*5	*k*6	*k*4	*k*1	*k*8	*k*2	*k*9	*k*10
*k*7	0									
*k*3	1	0								
*k*5	1	4	0							
*k*6	1	1	1	0						
*k*4	1	5	4	0	0					
*k*1	0	3	3	1	3	0				
*k*8	1	1	0	0	0	0	0			
*k*2	0	3	2	0	4	2	0	0		
*k*9	2	0	0	1	0	0	2	0	0	
*k*10	1	0	0	0	0	0	1	0	1	0
Number of entries for row i, column i	7	7	6	5	5	5	4	4	4	3

select x_{2j} such that y_{22} will be the smallest result. Any of the five assignments $x_{2j} = 1$ with $j \neq 3$ will yield $y_{22} = 0$, and for x_{23}, $y_{22} = 1$. Next tossing a five-sided die numbered 1–5 will help in making the choice. Suppose this number was 5 or the fifth from the equally available x_{2j}, specifically $x_{26} = 1$ and $x_{2j} = 0$ for $j \neq 6$; course-$k3$ is assigned to block 6.

Step 3. Again expanding eq. (3a) to assign the third course-$k5$:

$$\begin{aligned}
&((c_{31}x_{11} + c_{32}x_{21})x_{31} = (1\cdot0 + 4\cdot0)x_{31})\\
&+ ((c_{31}x_{12} + c_{32}x_{22})x_{32} = (1\cdot0 + 4\cdot0)x_{32})\\
&+ ((c_{31}x_{13} + c_{32}x_{23})x_{33} = (1\cdot1 + 4\cdot0)x_{33})\\
&+ ((c_{31}x_{14} + c_{32}x_{24})x_{34} = (1\cdot0 + 4\cdot0)x_{34})\\
&+ ((c_{31}x_{15} + c_{32}x_{25})x_{35} = (1\cdot0 + 4\cdot0)x_{35})\\
&+ ((c_{31}x_{16} + c_{32}x_{26})x_{36} = (1\cdot0 + 4\cdot1)x_{36}) = y_{33},\\
&0\cdot x_{31} + 0\cdot x_{32} + 1\cdot x_{33} + 0\cdot x_{34} + 0\cdot x_{35} + 4\cdot x_{36} = y_{33}.
\end{aligned}$$

Setting $x_{33} = 1$ or $x_{36} = 1$ gives $y_{33} = 1$ or $y_{33} = 4$, respectively. However, any choice of x_{31} or x_{32} or x_{34} or $x_{35} = 1$ yields $y_{33} = 0$. A random selection, one of four from the latter group, is made; $x_{31} = 1$ or course-$k5$ is assigned to block 1.

Step 4. Similarly the fourth course, $k6$, is assigned by using eq. 3,

$$\begin{aligned}
&(1\cdot0 + 1\cdot0 + 1\cdot1)x_{41} + (1\cdot0 + 1\cdot0 + 1\cdot0)x_{42}\\
&+ (1\cdot1 + 1\cdot0 + 1\cdot0)x_{43} + (1\cdot0 + 1\cdot0 + 1\cdot0)x_{44}\\
&+ (1\cdot0 + 1\cdot0 + 1\cdot0)x_{45} + (1\cdot0 + 1\cdot1 + 1\cdot0)x_{46} = y_{44},\\
&1\cdot x_{41} + 0\cdot x_{42} + 1\cdot x_{43} + 0\cdot x_{44} + 0\cdot x_{45} + 1\cdot x_{46} = y_{44};
\end{aligned}$$

and choosing from x_{42}, x_{44} or x_{45}, block 2, 4 or 5 yields $y_{44} = 0$.

Steps 5–10. These steps assign corresponding courses. Computations are similarly performed but become too unwieldy for this demonstration.

Step 11. If the schedule is satisfactory, an acceptable number of total conflicts is produced, no further computation is required.

However, if the results are unsatisfactory, the continuous repetition of steps 1–10 will produce additional schedules with other σ_x, total number of conflicts. This difference is due to the random selections made in this procedure. Note that the repetition starting with step 1 can begin as soon as the accumulated number of conflicts for a single pass exceeds some preceding σ_x. In addition, in this case a six-block schedule was attempted, however, larger or smaller block numbers could as well be tested and then all the schedules can be evaluated.

4. *Schedule Checking.* The input to this program is the schedule developed in step 3 followed by the student class cards. As each student's cards are read the program checks and lists the conflicts and the other undesirable types, if any.

V. Summary of Data

1. The first problem attempted, using this procedure, was the scheduling of final exams for approximately 700 students taking 99 different courses into 14, 16, 20 blocks. The following observations were made.

a. From data created by the above parameters, conflict-

free schedules were easily obtained using 14, 16, 20 blocks in no more than three attempts.

b. The number of conflicts were reduced by 50 percent after rearranging the matrix (Sec. IV).

c. The results obtained by testing for two finals on one day (Sec. III.2.1) were similar to those obtained using the combined tests mentioned in Sec. III.2.2–4.

d. Input data cards were compressed by making all numbers greater than 99 equal to 99. The net results were not affected. In most cases a single digit, 0–9, in the conflict matrix may be sufficient.

2. The statistics on the second use of the procedure involved in scheduling final exams for approximately 1100 students taking 150 different courses were compiled and are presented in Tables 4 and 5. All the observations made above except (a) apply. Tables 4 & 5 indicate the frequency distribution for the number of conflicts with 14 and 16 block schedules. These statistics were used in the

TABLE 4. FIFTY FINAL SCHEDULES CONTAINING 14 BLOCKS

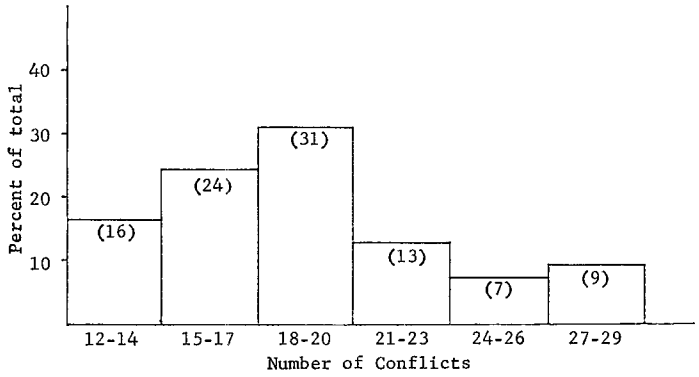
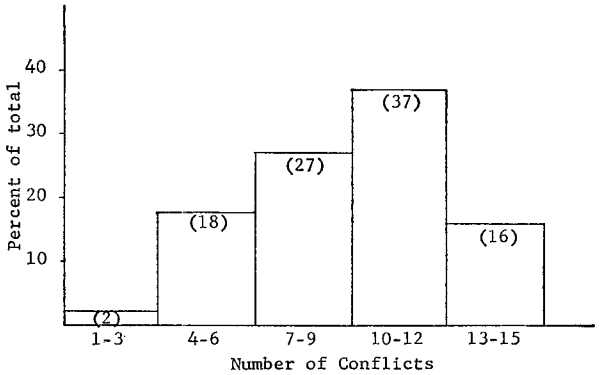


TABLE 5. FIFTY FINAL SCHEDULES CONTAINING 16 BLOCKS



selection of a 16-block schedule rather than a 14-block schedule. The undesirable effects (Sec. III.2.2–4) were reduced 50 percent by optimizing on these effects. The number of students affected was approximately 1 percent conflicts, 10 percent poor (see Sec. III) student schedules for the 16-block schedule. While the statistics were insufficient to draw a definite conclusion, as one may expect intuitively, there appeared to be a direct relationship between the number of conflicts and the number of poor-student schedules. This relationship appeared to be independent of block sizes 14-16-18 used.

Future runs are planned on a larger, faster computer for fuller statistical investigations.

Acknowledgments. Thanks are due to Dr. A. Finerman for his inspiration and cooperation, to Prof. P. Dollard for his notational contributions, to Mr. R. Birnbaum, Director of Institutional Records and Research, for the impetus and added insight, to Mr. J. Barr for his programming assistance. All played an important part in developing this procedure.

APPENDIX

In Sec. III references are made to improvements in schedules. In addition, other items to be optimized, such as room assignments, may be considered. The method of solution for this group of extensions of the basic conflict elimination procedure varies depending upon the item to be optimized and therefore is treated separately. One possible item to be considered is the number of occurrences of two exams per day. The day is defined as a pair of blocks $j, j+1$, where j is an odd number. In the assignment process for course i, consider those days from which the equal nonconflict blocks are to be selected. Next define each pair of blocks as z_{is} the union of x_{ij} and x_{ij+1}. Thus $(r_j + r_{j+1})z_{is}$, for $z_{is} = 1$ is the number of conflicts plus the number of occurrences of students taking two exams on one day. The selection of a $z_{is} = 1$ such that its corresponding $(r_j + r_{j+1})$ is the smallest of the set considered, yields a selection first from the minimal conflict group and then from the minimal two-exams-per-day set. Of course, choices from equal sets are made using a random selection.

In the sample problem the use of this method could be included. For example, in step 3, assignment of course 3, noting that: $r_j x_{3j} = y_{33}$ or $0 \cdot x_{31} + 0 \cdot x_{32} + 1 \cdot x_{33} + 0 \cdot x_{34} + 0 \cdot x_{35} + 4 \cdot x_{36} = y_{33}$, with x_{31}, x_{32} reference blocks in day 1 denoted as z_1; x_{33}, x_{34} reference blocks in day 2 denoted as z_2; x_{35}, x_{36} reference blocks in day 3 denoted as z_3. Then $\sum_{s=1}^{3} (r_j + r_{j+1})z_{3s} = y_{33}$ and $j = 2s-1$ or $0 \cdot z_{31} + 1 \cdot z_{32} + 4z_{33} = y_{33}$.

A conflict-free choice can be made from blocks 1, 2, 4 or 5, which occur on days 1, 2 or 3. Then, a choice from available blocks 4 or 5 and corresponding days 2 or 3 ($z_{32} = 1$ or $z_{33} = 1$) will yield zero conflicts and 1 or 4, respectively, occurrences of students who have two exams on one day. However, a selection from available blocks 1 or 2 in day 1 yields no occurrences of students who have two exams on one day. Since block 1 or 2 have the same contribution to the total result a random selection can be made to choose from block 1 or block 2 in day 1. Introducing this method into the conflict procedure, all courses can be assigned. By similar treatment but more involved logical definitions, Sec. III.2.2–4 can be mechanized. Each item can then be assigned a weight and treated in the totality. The procedural discussion of items 2–4 is omitted since it involves extensive logical detail.

RECEIVED MARCH, 1964; REVISED MAY, 1964.