Mathematical Scheduling

ROBERT GIBBONS AND DEAN JOHNSON

Communicated by Dr. Shafii-Mousavi

ABSTRACT: Often companies or organizations hold large conferences, with the goal of dispensing vast amounts of information to a diverse set of people. They also may wish to maintain a high level of interaction within small groups of participants. A model to design the groups of such a seminar should be built in such a way as to allow different classes of participation. We use basic ideas from modern combinatorics to develop methods for placing attendees into blocks. We then employ a greedy algorithm that chooses column shifts to be performed on these blocks. The combination of these two methods improves upon the results typically obtained from the combinatorial methods. Our algorithm is much faster than current greedy techniques.

In a day when information and the exchange of ideas is essential to our society, it is unfortunate that there are few optimization methods available to arrange conferences. Using a simple method of filling and shifting tables, we can greatly improve the productivity of any conference. As an example, we will use a variation of a problem on the 1997 COMAP Mathematical Contest in Modeling, and then generalize our method.

EXAMPLE PROBLEM

A company wishes to schedule a two day meeting to be attended by 29 board members. The first day will consist of three sessions of six discussion groups each with each discussion group being led by a senior officer, who is not a board member. The second day will have four sessions consisting of four discussion groups each. The senior officers will not participate in the second days sessions. Additionally, of the 29 board members, nine are in-house members. Our goal is to attempt to allow each board member to meet with each other board member the same number of times while minimizing common memberships of the groups for the different sessions. In addition we wish to satisfy the following criteria:

1. For the first day's sessions, no board member should be in the same senior officer's discussion group twice.
2. No discussion group should contain a disproportionate number of in-house members.

DEAN JOHNSON will graduate in May 1999 with a B.S. in Mathematics. After graduatation, he plans to enter a graduate program and become a professor of mathematics. Dean is currently the president of the IUSB chapter of the Mathematics Association of America.

ROBERT GIBBONS is a junior and plans to graduate with a B.S. in Mathematics. Robert is currently the treasurer of the IUSB chapter of the Mathematics Association of America, and is vice president of the IUSB Student Activities Board.

This paper is a continuation of their solution to the 1997 COMAP Mathematical Contest in Modeling.

KEY POINTS AND ASSUMPTIONS

1. The constraint that no board member should be in the same senior officer's discussion group twice must be absolutely adhered to.
2. The size of each group, and the number of in-house members in each group must be within one member.
3. Due to the complex nature of the given situation, in some circumstances a given pair of board members will be unable to meet at all, while other pairs will be required to meet more than once. By minimizing common occurrences we mean that we wish to minimize this situation as much as the algorithm allows.

APPROACH

There are two different approaches to this type of problem. One method uses a greedy algorithm that examines the placement of each member individually. The other uses combinatorial methods to fill the groups. Both of these methods have their strengths and weaknesses. A greedy algorithm allows for complete control over which group the members are placed in; however, it involves large amounts of computer calculation and memory. Current combinatorial methods are much faster, but involve constraints that most real situations do not follow.

Our approach to this problem is a combination of these two methods. We first applied a modified method from block design and then used a greedy algorithm to obtain the shifting pattern. This combination creates a method that works quickly, and still allows the required amount of control.

PROCEDURE

Day 1

We begin this problem by arranging the 29 board members vertically in a table as shown below.

Day 1, Session 1

Group 1	A	1	7	13	19	25
Group 2	B	2	8	14	20	26
Group 3	C	3	9	15	21	27
Group 4	D	4	10	16	22	28
Group 5	E	5	11	17	23	29
Group 6	F	6	12	18	24	

The first column represents the six senior officers, the unshaded parts of columns 3 and 4 represent the nine in-house board members, and the shaded parts of the four rightmost columns represent the non-in-house board members. From this diagram, the groups of the first session can be formed from each of the different rows of the table. It is important to note that the placement of members in the initial table is arbitrary, as long as the different types are kept together.

Now that we have our six discussion groups for the first session, the second and third sessions can be considered combinations of the first. By shifting the elements of each column, by a different amount, we can obtain these combinations. In this paper, columns will always be shifted up. To minimize repetition and avoid a board member meeting with the same senior officer twice, we must shift our columns according to the following rules:

1. Every column must shift up from one to five spaces per session and the cumulative position of the elements must be a value from one to five, to avoid meeting one of the six senior officers twice. The cumulative position of a column is an indicator of the total number of cells that a column has been shifted relative to its initial position.
2. In order to minimize the common memberships of successive groups, we need to minimize the number of times that any two columns are shifted up the same number of spaces during any given shift. This must also be true of our cumulative shifts.

If these rules are unable to be met exactly, then our goal is to minimize the number of times a rule is broken.

For the first day, the shifts and cumulative positions we find for each column are shown here.

Shifting Scheme for Day 1

Column #	**2**	**3**	**4**	**5**	**6**
First Shift	**1**	**2**	**3**	**4**	**5**
Cumulative	1	2	3	4	5
Second Shift	**1**	**3**	**4**	**5**	**2**
Cumulative	2	5	1	3	1

Applying the First Shift to the initial table gives us Session 2.

Day 1, Session 2

Group 1	A	2	9	16	23	
Group 2	B	3	10	17	24	25
Group 3	C	4	11	18	19	26
Group 4	D	5	12	13	20	27
Group 5	E	6	7	14	21	28
Group 6	F	1	8	15	22	29

To get Session 3 we can either apply the Second Shift to Session 2, or apply the second Cumulative Shift to the initial table.

Day 1, Session 3

Group 1	A	3	12	14	22	26
Group 2	B	4	7	15	23	27
Group 3	C	5	8	16	24	28
Group 4	D	6	9	17	19	29
Group 5	E	1	10	18	20	
Group 6	F	2	11	13	21	25

Day 2

Some of the board members failed to meet even once during the first day due to the fact that they were in the same column. To remedy this, we use the columns from the initial table in the first day as the rows in the initial table of the second day. This allows those members to meet on the second day. We apply this type of rotation to each of the different types of members attending the conference independently.

Day 2, Session 1

Group 1	1	2	3	10	11	12	13	14
Group 2	4	5	15	16	17	18	19	
Group 3	6	7	20	21	22	23	24	
Group 4	8	9	25	26	27	28	29	

We now apply a new set of shifts to the above table. Recall that the six senior officers do not participate in the second day. This allows us to shift columns by 0 spaces (stay where it is). This allows us unique shifts of 0 to 3. Following the same shifting scheme as the first day, we establish the following shifts.

Shifting Scheme for Day 2

Column #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
First Shift	**0**	**1**	**2**	**3**	**0**	**1**	**2**	**3**
Cumulative	0	1	2	3	0	1	2	3
Second Shift	**0**	**2**	**3**	**1**	**2**	**0**	**1**	**3**
Cumulative	0	3	1	0	2	1	3	2
Third Shift	**0**	**3**	**2**	**1**	**1**	**3**	**0**	**2**
Cumulative	0	2	3	1	3	0	3	0

On the second day, the number of groups is less than the size of each group. This forces two rows to move the same number of spaces for each shift. In this situation, we must minimize the number of times a pair of columns have shifted together. Performing the shifts for the second day, we obtain the following Sessions.

Day 2, Session 2

Group 1	1	5	20	26	11	18	24	
Group 2	4	7	25	10	17	23	29	14
Group 3	6	9	3	16	22	28	13	
Group 4	8	2	15	21	27	12	19	

Day 2, Session 3

Group 1	1	9	15	10	22	18	29	
Group 2	4	2	20	16	27	23	13	
Group 3	6	5	25	21	11	28	19	14
Group 4	8	7	3	26	17	12	24	

Day 2, Session 4

Group 1	1	7	25	16	27	12	29	14
Group 2	4	9	3	21	11	18	13	
Group 3	6	2	15	26	17	23	19	
Group 4	8	5	20	10	22	28	24	

GENERALIZED ALGORITHM FOR THE SCHEDULING PROBLEM

Our general algorithm works exactly the same way as our example did.

Initialization - Fill a table vertically with members to attend the conference, keeping different types of members together in as few columns as possible. We can ignore unused spaces, and fill them in if more people attend the conference.

Shifting - Fill a shifting table keeping the shifts and cumulative shifts as diverse as possible. If two columns are forced to shift together, check to see that they have not shifted together in the past. If members are allowed to be in the same group more than once, we will allow shifts and cumulative positions of zero. The rows in the initial table and after each shift are the group assignments.

MULTIPLE BLOCKS OF TIME

If the conference a runs for more than one day (as our conference did), use this method to transpose the tables between days.

Transposition - Fill a new table horizontally, such that there is no more than one empty seat in any group after each class of member is added to the table.

Shifting - Create a shifting table with the same rules as before, keeping in mind whether shifts of zero are allowed or not.

STRENGTHS AND WEAKNESSES

To consider the strengths and weaknesses of our method, we must compare it to the current greedy methods. Shifting entire columns of elements at a time reduces the number of variable that we must consider. This creates an algorithm that is easier to visualize, easier for individuals with little or no mathematical background to understand, and much faster for a computer to process. Table A shows the processing times that result from applying the two different types of algorithms on conferences of different sizes.

The differences in calculation time become even more noticeable in conferences that involve significantly more people. Not surprisingly, this increase in speed results in a small loss in optimality. To measure the effectiveness of the mixing algorithms, we can report an averaged count of how many new members each attendee meets. Out of a seminar of 29 people it is possible to meet a maximum of 28 people (one cannot meet oneself). Table B shows the results of our method compared to the results of four different greedy algorithms.

Our method scores slightly lower in mixing of members, however it only differs by 2.28 members in this example. In larger seminars the cost of calculation time is not justified by the slight increase in optimality.

ADAPTABILITY

Our use of combinatorial methods becomes apparent when we try to adapt to other models. This method allows us to schedule any event that involves several activities occurring simultaneously. However, we also have an added constraint that there must be more than two people attending each group. For example, our model would not be useful in the scheduling of a chess tournament, since only two players can participate in each game.

We can, however, allow the number of groups to vary with each session to make this into a good model for scheduling classes for a high school, where there are a many students that have the same set of required classes.

REFERENCES:

1) Blinn, K., Boyer, E., Eversole, S., and Gibbons, R., "Scheduling a Dinner Party", Indiana University South Bend (unpublished), 1996.
2) Cao, H., Shi, Z., Yang, H., "An Assignment Model for Fruitful Discussions", *The UMAP Journal* 18 (3) 297-306, COMAP Inc., 1997.
3) Castro, D., Renze, J., Weininger, N., "Using Simulated Annealing to Solve the Discussion Groups Problem", *The UMAP Journal* 18 (3) 307-320, COMAP Inc., 1997.
4) Corduneanu, A., Hsai, C., O'Donnell, R., "A Greedy Algorithm for Solving Meeting Mixing Problems", *The UMAP Journal* 18 (3) 331-342, COMAP Inc., 1997.
5) Horstman, J., Kawabata, J., Moore, J., "Meetings, Bloody Meetings!", *The UMAP Journal* 18 (3) 321-329, COMAP Inc., 1997.
6) Norvig, P., Russell, S., *Artificial Intelligence: A Modern Approach*, New Jersey: Prentice-Hall, 1995.

Table A: Processing Times

Machine Type:	Pentium 66	HP 712 / 60	Pentium 100	Pentium 90
Number of Members Attending	Our Method	Greedy 1[4]	Greedy 2[3]	Greedy 3[6]
29	7 sec	6 min 38 sec	5 min	7 min
48	10 sec	9 min 53 sec	N. A.	N. A.
59	14 sec	22 min 6 sec	N. A.	N. A.
100	39 sec	58 min 1 sec	N. A.	N. A.
200	3 min 54 sec	N. A.	N. A.	N. A.

Table B: Mixing Effectiveness

	Our Method	Greedy 1	Greedy 2	Greedy 3	Greedy 4[5]
New Aquaintences	24.79	25.79	27.07	26.64	26.71