EMPLOYEE SCHEDULING METHODS USING A CALCULATOR

Judith Aronow
5590 Frost
Beaumont, TX 77706
aronowju@hal.lamar.edu
409/892404383

EMPLOYEE SCHEDULING METHODS USING A CALCULATOR

Scheduling employee times-on and times-off

Objective and constraints:

Minimize the difference (slack) between supply and demand; i.e. between the number of employee-time units scheduled and the number of employee time units required. We want to be able to perform this feat subject to two primary constraints:

1. **the time off-time on planning horizon (how long a period in time units the schedule covers)**
2. **the rules for time-off-time-on, such as five days on, two days off**

Attributes of the problem:

- **skill level of the worker**
- **length of tenure of the worker (privilege status of the worker)**
- **time requirements in person-time units, such as worker-hours, days, weeks (demand for services)**
- **rules for number of time units off or on**
- **consecutivity of time units-off or time-units on**

A measure of success of the schedule is its efficiency, E:

$$E = \frac{\text{Total employee time units required}}{\text{Total employee time units scheduled}}$$

The closer E is to 1, the more efficient the schedule is.

In this presentation, we will make some simplifying assumptions:

- **All workers have the same skill level.**
- **The privilege status of each of the workers is the same.**
- **Time units on and time units off will be uniform for all workers.**
- **Time units off will be consecutive.**

Let the planning horizon be one week long. Let the days on-days off be five days on and two consecutive days off. Then the possibilities for the two days off are

1. **Monday and Tuesday (MT)**
2. **Tuesday and Wednesday (TW)**

3. Wednesday and Thursday (WT)
4. Thursday and Friday (TF)
5. Friday and Saturday (FS)
6. Saturday and Sunday (SS)
7. Sunday and Monday (SM)

Let the daily hourly worker requirements be:

Monday	**9 hrs.**
Tuesday	**8 hrs.**
Wednesday	**8 hrs.**
Thursday	**10 hrs.**
Friday	**8 hrs.**
Saturday	**9 hrs.**
Sunday	**8 hrs.**
Total	**60 hrs.**

We wish to create a schedule which minimizes slack and in which each of 12 workers has a pair of consecutive days off.

The algorithm we will pursue has three steps:

1. **Choose two consecutive days each of which have the lowest requirements .**
2. **If there is a tie, choose the pair with the lowest sum of requirements**
3. **If there is still a tie, choose the first of the pairs.**

Solution and Schedule

1. We create a vector, A1 whose elements are the 7 daily requirements:

A1 = [9 8 8 10 8 9 8].

2. We create and store the 7 vectors MT, TW, WT, TF, FS, SS, SM. Each of the vectors will consist of 2 zero's for the two days off, and the remaining five will be -1's. For example,

MT = [0 0 -1 -1 -1 -1 -1] SM = [0 -1 -1 -1 -1 -1 0],
and so forth.

3. Following the steps of the algorithm, add A1 and the chosen days-off vector. The 2 days will be TW. We store the result as A2. Then

A2← A1 + TW

This sum reduces the work requirement by 1 hour for each worker on each of h/h 5 days on, and by zero on the 2 days off, Tuesday and Wednesday.

4. We create a 7 X 7 <u>schedule</u> matrix, W,[1] whose columns are the days of the week, M, T, W, T, F, S, SU and whose rows are the days-off possibilities, MT, TW, WT, TF, FS, SS, SM. Then, for example, the element (TF, W) = (4, 3), and is the number of workers working on Wednesday having the combination Thursday-Friday as days off.

Select the row corresponding to the days-off combination and substitute a new row consisting of the entries in the old row plus the negative of the chosen days-off vector. For example

$$W(6) - SS \rightarrow W(6)$$

replaces the 6th row of W, the Saturday-Sunday row, with that row diminished by the vector SS. In effect it adds 1 (hour) to every element in the row except the 2 days off, because SS = [-1,-1,-1,-1,-1,0,0] = -[1,1,1,1,1,0,0].

5. We repeat the procedure for A2, and produce an A3 which is stored.

6. Repeat beginning at step 1.

7. Stop when all requirements are filled, or equivalently when An+1 consists of zeroes.

Note, that if there are n employees, there will be n iterations and n+1 A-vectors. It is not essential to store the A-vectors. However, mistakes are easier to trace when the intermediate vectors are available. The entry in each cell of the matrix tells the number of workers for that day who have a certain 2-day pair off. The sum of each of the columns is the hourly requirement for that day.

	Mon	Tues	Wed	Thus	Fri	Sat	Sun
MT	0	0	0	0	0	0	0
TW	4	0	0	4	4	4	4
WT	0	0	0	0	0	0	0
TF	2	2	2	0	0	2	2

[1]**Be sure to press [MATRX][EDIT] to define the matrix W. It will automatically be filled with zeroes.**

FS	**2**	**2**	**2**	**2**	**0**	**0**	**2**
SS	**1**	**1**	**1**	**1**	**1**	**0**	**0**
SM	**0**	**3**	**3**	**3**	**3**	**3**	**0**
	Σ = 9	**Σ = 8**	**Σ = 8**	**Σ = 9**	**Σ = 8**	**Σ = 9**	**Σ = 8**

The matrix W, will, of course, have no headings. It will be necessary then, to keep in mind the correspondence between numbers of the rows and columns and their meanings.

```
[[0 0 0 0 0 0 0]
 [4 0 0 4 4 4 4]
 [0 0 0 0 0 0 0]
 [2 2 2 0 0 2 2]
 [2 2 2 2 0 0 2]
 [1 1 1 1 1 0 0]
 [0 3 3 3 3 3 0]]
```

Schedule Matrix W

Notice that the efficiency is

$$E = \frac{\text{Total number hours required}}{\text{Total number hours scheduled}} = 1$$

and there is no slack.

In this example

R1. $\frac{\text{total number of hours required for the week}}{\text{number of days-on for any employee}} = \frac{60}{5} = 12$**, an integer**

The integer quotient is a requirement for a no-slack answer.

Also notice that the

R2. **number of employees is at least as large as the maximum hour requirement for any one day.**

To vary the problem, try violating R1 or R2. The efficiency will be less than 1 because there would then be slack.

In this problem, the days-off vectors for each A are:

A1---TW	**A8-----TW**
A2---SS	**A9-----FS**
A3---SM	**A10----SM**
A4---TW	**A11----TW**
A5---FS	**A12----TF**
A7---SM	

This exercise in scheduling is given interactively to keep the problem in mind as a sequence of decisions and consequences and also to increase efficiency in use of the calculator. However, the algorithm is programmable with interactive decisions to be made, programmed in. Creating a program could be a good exercise in itself.

It should be mentioned to the student, that employee scheduling problems can be extremely varied in their requirements. Days-off are not always sequential, and are sometimes rotated. Work rules and requirements can further complicate the problem. When there are very large numbers of employees and/or the set of constraints is also large, integer progamming is used and these problems solved using Integer Programming software for computers.

References

Nanda, Ravinder and Jim Browne, <u>Introduction to Employee Scheduling</u>. Van Nostrand Reinhold, 1992 (New York).

Wagner, Harvey M., <u>Principles of Operations Research</u>. Prentice Hall, Inc., 1975 (Englewood Cliffs, N.J.)

Judith Aronow
5590 Frost
Beaumont, TX 77706
email: aronowju@hal.lamar.edu