SHIFT SCHEDULING WITH
OPTIMIZED SERVICE LEVELS
AND EMPLOYEE SATISFACTION

TANJA VAN HECKE,* University College Ghent

Abstract

This paper discusses the integer programming approach for dealing with resource allocation problems and employee satisfaction during shift scheduling. The ability to take into account the personal preferences of shift workers has an important impact on employee satisfaction. This paper describes an enhanced approach to this scheduling problem based on operations research.

Keywords: Scheduling; call center; integer programming

2010 Mathematics Subject Classification: Primary 90C10
Secondary 90B35; 68M20

1. Introduction

Companies have as their common objectives the delivery of high quality products or services and the reduction of labour costs. One way of keeping employees motivated is to grant their wishes concerning shift preferences. A good working schedule attempts to reconcile two requirements: fulfilling as closely as possible the wishes of the employees, and guaranteeing minimum employee deployment. In the case of call centers, the number of employees required to be present per unit of time (e.g. an hour) is derived from the predicted workload by means of the Erlang C formula [1]. A distinction has to be made between different shifts, e.g. part-time shifts, full-time shifts, morning shifts, evening shifts, shifts that ensure services at noon, and so on. Schrijver [3] and Wolsey [4] formulated the allocation problem of shifts and employees as an integer programming problem, where an optimal solution of a minimization problem is required, given certain restrictions. This paper describes how this allocation problem of employees and shifts, based on a given workload, can be enlarged to a system where the personal employee shift preferences are incorporated. Moreover it is shown that the goal of greater job satisfaction may be associated with further cost reduction.

2. Call centers

In the case of call centers, the required staff may be estimated by means of the Erlang C or Erlang delay formula, named after the Danish mathematician A. K. Erlang (1878–1929) who derived the formula [1]. It states that the chance that any client of a call center has to wait less than \( t \) seconds is

\[
C(s, \lambda \beta) e^{-(s/\beta - \lambda)t},
\]

Received 28 September 2010; revision received 20 October 2010.

* Postal address: Faculty of Applied Engineering Sciences, University College Ghent, Schoonmeersstraat 52, 9000 Ghent, Belgium. Email address: tanja.vanhecke@hogent.be

89
where
\[
C(s, a) = \frac{a^s}{(s-1)! (s-a)} \left( \sum_{j=0}^{s-1} \frac{a^j}{j!} + \frac{a^s}{(s-1)! (s-a)} \right)^{-1}
\]
is the probability of delay, i.e. the probability that an arbitrary caller finds all agents (employees) occupied. Here \( s \) is the number of agents, \( \lambda \) is the average number of arrivals per time unit, and \( \beta \) is the average service time of calls or average holding time. Then \( a = \lambda \beta \) is the load of the call center, a unit called the Erlang. The formula is applicable under the assumption that there are no disconnections: thus every caller waits until he or she reaches an agent, otherwise the Erlang A formula should be used. Moreover we assume that the number of agents is higher than the load, so that \( s > a \), and variations in the offered load can be absorbed due to the overcapacity \((s-a)\) of the system.

By means of this Erlang C formula it is possible to determine the required staff, given the average number of incoming calls per time unit, the average service time and the service level offered, by means of the requisite waiting times. The Erlang C calculator [6] can be helpful.

3. Integer programming problem

The allocation problem of employees and shifts was formalized in [2] as the integer programming problem outlined in (1), below (see [3] and [4]), using the arrays \( A = (a_{ij})_{n \times m} \), \( C = (c_j)_{m \times 1} \), \( B = (b_i)_{n \times 1} \), and \( X = (x_j)_{m \times 1} \), where \( m \) is the number of shifts and a day is divided into \( n \) time periods. The problem is to find

\[
\min C^\top \cdot X,
\]
under the condition \( A \cdot X \geq B \), where \( x_j \geq 0 \), \( x_j \) an integer \((j = 1, \ldots, m)\).

The variable \( x_j \) represents the number of employees working during shift \( j \), \( c_j \) denotes the cost of one employee working during shift \( j \), and \( b_i \) denotes the minimal staffing during period \( i \). The array \( A \) links the time periods with the different shifts as follows:

\[
a_{ij} = \begin{cases} 
1 & \text{if period } i \text{ is a working period for shift } j, \\
0 & \text{if period } i \text{ is not a working period for shift } j.
\end{cases}
\]

A measure for comparing solutions of an allocation problem is the productivity of a working schedule. We define the productivity \( P \) in time as the ratio of the sum of the working hours of the working agents and the sum of the working hours of all required agents to cover the minimal staffing, or

\[
P = \frac{\sum_{j=1}^{m} \sum_{i=1}^{n} a_{ij} x_j}{\sum_{i=1}^{n} b_i}.
\]

Solving problem (1) creates a working schedule, but does not cope with employee preferences. Therefore we need a more general formulation of the problem, where a \((k \times m)\)-array \( X \) of unknown \( x_{ij} \) values is used, with \( k \) the total number of employees. Here,

\[
x_{ij} = \begin{cases} 
1 & \text{if agent } i \text{ is working during shift } j, \\
0 & \text{if agent } i \text{ is not working during shift } j.
\end{cases}
\]
To express the preferences, we use the \((k \times m)\) cost array \(CP = (c_{pij})_{k \times m}\), where \(c_{pij}\) expresses the cost of allowing agent \(i\) to work in shift \(j\). If the agent dislikes the shift, the cost to allow him to work in this shift will be higher. Defining the vector \(I = (1 \ldots 1)^T\) of dimension \(k\), the problem can be formulated as finding

\[
\min \sum_{i=1}^{k} \sum_{j=1}^{m} c_{pij}x_{ij},
\]

under the condition \(A \cdot X^T \cdot I \geq B\),

where \(x_{ij} \in \{0, 1\}\),

\[
\sum_{j=1}^{m} x_{ij} \leq 1 (i = 1, \ldots, m).
\]

The last set of inequalities expresses that an employee can be allowed to work for one shift at most.

4. Example

To illustrate the method described, we use the following simple example. We consider a company with ten employees where service is offered on Monday from 8 am to 8 pm. The day is divided into 12 periods of one hour, so we know the required staffing number, based on measures of incoming calls during previous Mondays. The required service on a Monday is modeled by the vector

\[
B = (1 \ 2 \ 1 \ 2 \ 3 \ 2 \ 2 \ 4 \ 2 \ 2 \ 1 \ 1),
\]

based on the Erlang C formula with an input of \(\lambda\) as in Table 1, with \(\beta = 10\) minutes and with a service level which ensures that 70% of the clients wait less than 1 minute.

<table>
<thead>
<tr>
<th>Period</th>
<th>(\lambda)</th>
<th>(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
The array
\[
A = \begin{pmatrix}
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}^T
\] (4)
expresses the spread of the first three shifts on a Monday, illustrated in Figure 1.

If the cost for an agent working during shifts 1, 2, or 3 is 37, 36, or 34 euros respectively, the function

\[37x_1 + 36x_2 + 34x_3\]

is to be minimized.

Using the linear programming tool LPSolve v5.5.0.15 [5] to perform the calculations of the integer programming problem (1), we find a minimal cost of 250 euros on a Monday which can be reached when two agents work during the first shift, three agents in the second shift, and two agents in the third shift. With this setting, the productivity in time (2) is only 44.68% (see Figure 2). If an additional fourth shift (as in Figure 1) is incorporated to cover the hours
at noon (with a cost of 40 euros), the function

\[ 37x_1 + 36x_2 + 34x_3 + 40x_4 \]

reaches the smaller minimal value of 217 when one agent works during the first shift, two agents during the second and third shifts, and one agent during the fourth shift. This creates less overcapacity as the productivity rises to 55.26% (see Figure 2).

We take into account the personal preferences of the ten agents, by using the array

\[
CP = \begin{pmatrix}
30 & 50 & 60 \\
30 & 30 & 30 \\
40 & 40 & 40 \\
20 & 20 & 30 \\
40 & 40 & 20 \\
30 & 30 & 30 \\
50 & 40 & 30 \\
50 & 50 & 50 \\
60 & 30 & 30 \\
20 & 30 & 20
\end{pmatrix},
\]

which expresses that agent 2 shows indifference to the work periods, while agent 5 prefers to work in the evening. This array is chosen so that the average values per column are 37, 36, or 34 respectively (the corresponding values of the basic example). The cost is determined partly by the experience and seniority of the agent and partly by remuneration for willingness to work during that shift.

The optimal solution of problem (3) is reached when only the variables \(x_{22}, x_{41}, x_{53}, x_{62}, x_{73}, x_{92},\) and \(x_{101}\) are 1, with all other unknowns having the value 0. This means that agents 4 and 10 are at work during the first shift, agents 2, 6, and 9 during the second shift, and agents 5 and 7 during the third shift. Then the total cost reaches its minimal value of 180 euros while offering good quality services. Even here there is an overcapacity during several time periods, e.g. five agents are available from 10 am to 11 am, where only one is needed. With these settings, no better solution can be found due to the bottleneck over lunch time for this example. More optimal results can be reached by distinguishing more types of shifts, e.g. the extra fourth shift as in Figure 1. Then a more fitting solution of problem (3), namely agent 4 working during the first shift, agents 2 and 6 during the second shift, agents 5 and 10 during the third shift, and agent 7 during the fourth shift, is offered together with a further reduction of the cost to 160 euros. This solution was obtained when the array \(CP\) (5) was supplemented with a fourth column where each element takes the average value 40. The array \(A\) from (4) was adjusted as well.

5. Conclusions

Applying the technique of integer programming makes it possible to construct adjusted shift schedules which take into account working time regulations as well as employee preferences. An example shows that our approach to the allocation problem of employees and shifts, taking employee preferences into account, should not lead to a cost increase. On the contrary, an extra cost reduction makes this approach even more optimal.
References