Job Rotation Using the Multi-Period Assignment Problem

By: J. Bhadury and Z. Radovilsky

Bhadury, J. and Z. Radovilsky. "Job Rotation Using the Multi-Period Assignment Problem" with Z. Radovilsky. Published in *International Journal of Production Research*, Vol. 44, No. 20, 4431-4444 (Oct 2006).

Abstract:
This paper addresses the current gap in the literature on quantitative tools that enable a manager to build schedules which incorporate job rotation by employees. This is done within the framework of the well-known Assignment Model recast in a multi-period setting. In addition to the usual objective of minimizing the total cost of assignment, we also consider a requirement to minimize the boredom felt by employees due to continued repetition of the same task over consecutive periods. Depending on alternative definitions of 'boredom', different bi-objective optimization models are formulated and solved using polynomial time algorithms or simple heuristics. In the case of heuristics, their implementation is discussed and computational experience is also reported.
Keywords: Assignment model; Repetitive tasks; Job rotation; Cyclic scheduling

Article:

1. Introduction

The most well known quantitative model for assigning workers to jobs is the *Assignment Model*, which has been utilized for a long time in numerous scheduling applications (Pinedo 1995). The traditional assignment problem seeks to assign a given number of workers to a given set of tasks while attempting to optimize some criteria such as the total cost or profit generated by the assignment. However, this approach has one major drawback: when the same problem is repeatedly solved, the worker-task assignments may turn out to be the same, i.e. the same jobs will need to be repeatedly performed by the same workers without any rotation. This may lead to boredom on the job due to repetition, thus underscoring the need for job rotation. However, this problem has not received attention in the traditional literature on scheduling of repetitive tasks (Pinedo 1995, Shtub *et al*. 1996, Al-Subhi *et al*. 1997).

It is interesting to note that independent of the multi-period assignment models, job rotation is a commonly observed practice in today's industry and an active area of research (Osterman 1994, 2000, Gittleman *et al*. 1998, Krajewski and Ritzman 1999, Ortega 2001). The purported benefits of job rotation in the literature are: fostering employee learning, reducing employee boredom, leading to increased motivation and increased human capital accumulation. In addition, Ortega (2001) argues that job rotation also results in '*firm learning*', i.e. employers learning more about its employees. However, empirical evidence of the success of job rotation is mixed. Based on

implementations at companies such as American Cyanamid, Bethlehem Steel, Ford and TRW, Griffin (2002) postulated that job rotation has not been very successful in either increasing employee motivation or satisfaction. On the other hand, Cunningham and Eberle (1990) and Davis and Taylor (1989) have reported that job rotation improves employee satisfaction, although it may not improve employee performance.

Our research shows that, in almost all the studies cited above, job rotation is done by mangers on a simplistic rule, for example a different job every day or on an *ad hoc* basis. This leads us to conjecture that the effectiveness of job rotation can be significantly increased if proper quantitative models were available to help managers decide good rotation plans. Our conjecture is supported by research done by Moorehead and Griffin (2001), Konz (1983) and Niebel (1993), who have concluded that the ineffectiveness of the job rotation approach is highly correlated with the way the jobs are being scheduled. The need for an effective job rotation model is further necessitated by the fact that contemporary management practice requires, in many cases, having multi-skilled employees that will be cross-trained in different jobs available by rotating these employees in a predetermined fashion (Dulin 1998, Thompson 1999, Anselmi and Sundarrajan 2000).

All this supports the rationale and motivation for our paper, which is to develop a bi-objective, multi-period assignment model that seeks to assign a given set of workers to a given set of tasks over a number of periods (henceforth, assumed to be days, without loss of generality) with the aim of rotation. The two objectives that we consider in the paper are: (1) minimize the total cost of assignment over all the days and (2) minimize the 'boredom' that the workers feel with their task assignments over all the days. In the next section, we discuss the preliminaries and formally state our multi-period assignment model. Thereafter, the next three sections consider three different versions of this generic model. The last section concludes the paper by summarizing our findings and proposing avenues for future research.

2. Preliminaries

In this section we will discuss the mathematical notation used in the paper and then, on this basis, present a formulation of our multi-period assignment model with rotation. We assume that the number of workers is given by $\{1, 2, \ldots, i, I\}$ and that each of them is to be assigned to a single task from a set given by $\{1, 2, \ldots, j, \ldots, J\}$. As described in the literature on Assignment Models, we will assume without loss of generality that $I = J$, since, if that is not the case, we can easily satisfy this assumption by creating dummy jobs or workers with zero assignment costs. The assignment problem is multi-period, in the sense that it has to be repeated for each of K periods denoted by $\{1, 2, 3, \ldots, k, K\}$. In this paper, without loss of generality, we will assume each period to be a day, but they could as easily be hours or weeks. For example, when three workers need to be assigned to three different tasks over 7 days, we will have $I = J = 3$ and $K = 7$. We will also assume that $C = \{c_{ij}\}$ represents the cost matrix of assignment; in other words, if worker i is asked to perform task j on any given day, the cost incurred by the organization for that day due to this assignment is c_{ij}.

Given this, the main goal of the assignment model is to identify an assignment of these I workers to these J jobs in each of the K days, while simultaneously attempting to optimize the following two, possibly conflicting, objectives.

- Objective 1. Minimize the total cost of assignment over the entire *K* days.
- Objective 2. Minimize the 'boredom' that the workers feel with their individual assignments over the entire *K* days. This is assumed to be achieved by attempting to avoid repetition of the same task by a worker over these *K* days. Thus, the second objective can also be interpreted as requiring job rotation in the assignment.

While Objective 1 is standard in the existing literature on assignment models and is therefore easily formulated, the same cannot be said about the second objective. This necessitates the formalization of that objective in terms of the input parameters of the model—and this is exactly where our paper separates into the three different models presented in sections 3, 4, and 5. Each one of these models measures boredom in a different way, thus resulting in three different formulations of Objective 2 in these sections. Nonetheless, all these formulations use the following Boolean assignment variable $x_{ij}^{(k)}$:

$$x_{ij}^{(k)} = \begin{cases} 1, & \text{if worker } i \text{ is assigned to task } j \text{ in period } k \text{ in any given assignment,} \\ 0, & \text{else.} \end{cases} \quad (1)$$

Given the definition of $x_{ij}^{(k)}$ in (1), the general multi-period assignment model that we address in this paper can be formulated as the following integer programming problem:

$$\begin{aligned} &\text{Min } \{\text{Objective 1; Objective 2}\}, \\ &\text{s.t.} \\ &\sum_{i=1}^{I} x_{ij}^{(k)} = 1 \quad \forall k,\ j, \\ &\sum_{j=1}^{J} x_{ij}^{(k)} = 1 \quad \forall i,\ k. \end{aligned} \quad (2)$$

The constraints in equations (2) are standard in all assignment models—they ensure that, in any given assignment, each worker is assigned to exactly one job and each job to exactly one worker in each of the *K* days. As mentioned above, Objective 1, which measures the total cost associated with any given assignment, is described as $\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ij}x_{ij}^{(k)}$.

3. Model 1. Minimize maximum number of repetitions

In this section, we present the first model that is based on the first formulation of Objective 2. Herein, we assume that, given any assignment, the boredom of any worker with a given task over the entire *K* days is presented by the total number of times that s/he will be assigned to this task over these periods. In other words,

$$B(i,\ j) = \text{Boredom of worker } i \text{ with task } j \text{ for a given assignment} = \sum_{k=1}^{K} x_{ij}^{(k)}.$$

Given this, we measure the boredom of this worker *i* with this entire given assignment as

$$B(i) = \text{Max}_{j=1,2,\ldots,J}\{B(i,\ j)\}. \quad (3)$$

Formula (3) measures the boredom of worker *i* with this given assignment as the maximum number of times that s/he has had to repeat any task in this assignment. With this formulation of

Objective 2, the first model that we present for solving the multi-period assignment problem in equations (2) can be represented as the following bi-objective integer programming problem:

$$\text{Min} \left\{ \text{Max}_{i=1,2,\ldots,I} \{B(i)\} ; \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} c_{ij} x_{ij}^{(k)} \right\}, \text{ s.t.}$$

$$\sum_{i=1}^{I} x_{ij}^{(k)} = 1 \quad \forall k, j, \tag{4}$$

$$\sum_{j=1}^{J} x_{ij}^{(k)} = 1 \quad \forall i, k.$$

Since problem (4) is a bi-objective optimization problem, solving it entails the finding of the efficient frontier, which is also called the set of Pareto Optimal solutions (Taha 1997). In order to do so, note that $B(i)$ is guaranteed to be an integer in the interval $[1, K]$. Thus, problem (4) can be solved by parametrizing on $B(i)$, which is defined in formula (3). In other words, the optimal solution to (4) can be found by solving the following optimization problem (5) repeatedly, once for each value of $q = 1, 2, \ldots, K$:

$$\text{Min} \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} c_{ij} x_{ij}^{(k)},$$

s.t.

$$\sum_{i=1}^{I} x_{ij}^{(k)} = 1 \quad \forall k, j, \tag{5}$$

$$\sum_{j=1}^{J} x_{ij}^{(k)} = 1 \quad \forall i, k,$$

$$B(i) \le q \quad \forall i.$$

To solve optimization problem (5), note that, for a given integral value of $q \in [1, K]$, this problem can be formulated as the following Min-Cost Flow Problem, which is known to be solvable in polynomial time (Ahuja *et al*. 1993, Taha 1997). In fact, the linear programming formulation given below can also be used to solve this model and is guaranteed to give integer solutions. This Min-Cost Flow formulation is defined on a graph $G = (V, A)$, whose vertices and arcs are as follows (see figure 1):

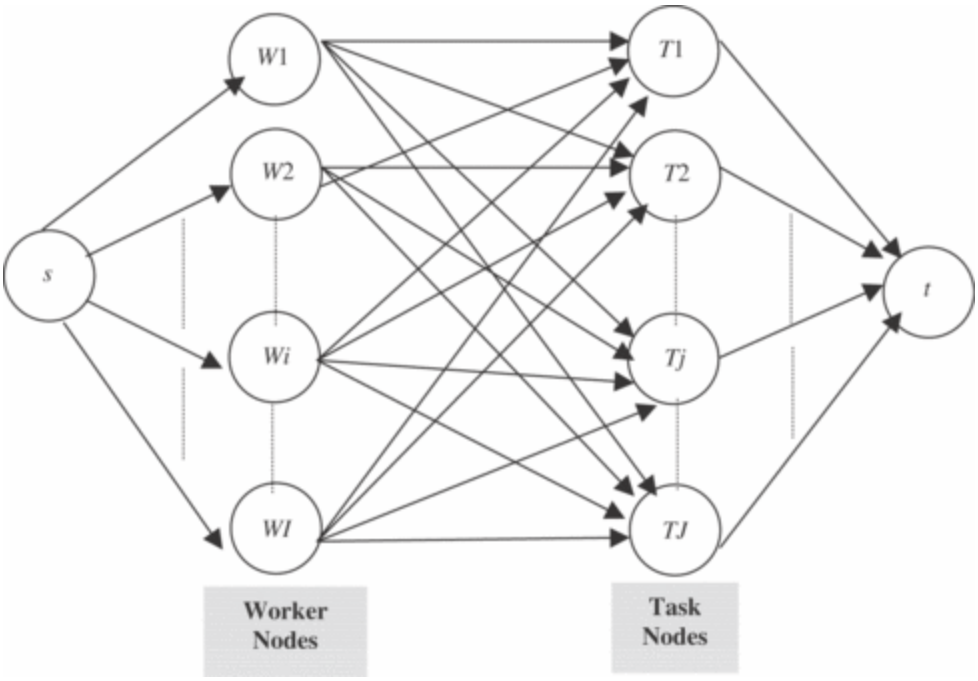

Figure 1. Graph $G(V, A)$.

V, the set of vertices, is given by

- s and t that denote source and sink nodes, respectively,
- I 'Worker Nodes', given by W1, W2, … , Wi, … , WI,
- J 'Task Nodes' given by T1, T2, … , Tj, … , TJ.

A, the set of arcs, is given by

- (s-Worker) Arcs. There are I such arcs, each of which goes from the source node s to each of the worker nodes. The lower bound, upper bound and cost of flow on each of these arcs is set to K, K and 0, respectively, to ensure that each worker has an assignment for each of the K days.
- (Tasks-t) Arcs. There are J such arcs, each of which goes from each Task Node to the sink node t. The lower bound, upper bound and cost of flow on each of these arcs is set to K, K and 0, respectively, to ensure that each job has a worker assigned to it on each of the K days.
- (Worker-Task) Arcs: there are ($I \times J$) such arcs. Each of these arcs goes from every worker node Wi to every task node Tj. On each of these arcs, the lower bound and upper bound are fixed at 0 and q, respectively. The cost of flow on an arc going from worker node Wi to task node Tj is fixed at c_{ij}. This guarantees that no worker will be assigned to any job more than q times subject to all this and other constraints imposed by the capacities of (s-Worker) and (Tasks-t) arcs; the total cost of the assignment will be minimized.

For illustrative purposes, we present in figure 2 the specific instance of the above graph $G(V, A)$ for the case where $I = J = 2$ (i.e. there are two workers to be assigned to two jobs) and the costs are as follows: $c_{11} = 10$, $c_{12} = 20$, $c_{21} = 30$ and $c_{22} = 40$. The total number of periods, $K = 7$ (i.e. one week), and we assume that we are solving (5) for $q = 4$. In that case, solving (5) will entail

finding the Min-Cost flow on the instance of *G* shown in figure 2. Note that the triple [… , … , …] on every arc in that figure represents the lower bound, upper bound and cost per unit flow, respectively, for that arc. For example, the triple [7,7,0] for the arc (*s*,*W*1) indicates that the lower bound and upper bound of flow on that arc are 7 units each and the cost per unit flow is 0. By solving the Min-Cost Flow problem (any commercially available linear programming software will suffice), one can verify that an optimal solution is as follows: Flow on (*W*1, *T*1) = Flow on (*W*2, *T*2) = 4 and Flow on (*W*1, *T*2) = Flow on (*W*2, *T*1) = 7. This translates to a policy where Worker 1 is assigned to Task 1 for the first four days of the week and to Task 2 for the last three. Worker 2 is assigned to Task 2 for the first four days and to Task 1 for the last three. The total scheduling cost is 350.

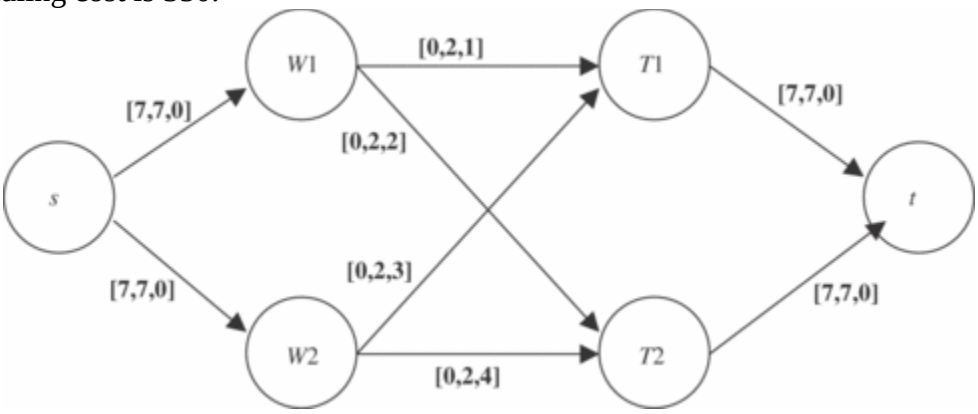

Figure 2. Specific instance of graph *G*(*V*, *A*).

4. Model 2. Minimize consecutive repetitions

In this section, we will use a different definition of 'boredom', which will lead to an alternative formulation of our model and a different solution methodology. The managerial premise of this alternative definition of boredom is that, in a good assignment, no worker should have to repeat the same job twice in any two (or preferably more) consecutive days. To formulate this mathematically, we define *DAYS-OF-ROTN* as a parameter that represents the minimum number of consecutive days over which it is desired that no worker should have to repeat any job. Hence, if *DAYS-OF-ROTN* = 2 for a given assignment, then it implies that there is a desire that no worker should repeat the same job in any two consecutive days in this assignment. Given an assignment, we define *B*(*i*, *j*) as the boredom felt by worker *i* from task *j* in this assignment over the entire *K* days as follows:

$$B(i,j) = \text{Max}_{k=1,2,\ldots,(K-DAYS\text{-}OF\text{-}ROTN)}\left\{ \sum_{q=k}^{q=k+DAYS\text{-}OF\text{-}ROTN} x_{ij}^{(q)} \right\}. \qquad (6)$$

The boredom felt by this worker *i* from the entire assignment is defined as *B*(*i*), which is given by

$$B(i) = \text{Max}_{j=1,2,\ldots,J}\{B(i,j)\}. \qquad (7)$$

Given these definitions in (6) and (7), the problem of finding the best assignment can be formulated as the following bi-objective integer programming problem:

$$\text{Min}\left\{\text{Max}_{i=1,2,\ldots,I}\{B(i)\};\ \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ij}x_{ij}^{(k)}\right\},$$

s.t.

$$\sum_{i=1}^{I} x_{ij}^{(k)} = 1 \quad \forall k, j, \tag{8}$$

$$\sum_{j=1}^{J} x_{ij}^{(k)} = 1 \quad \forall i, k,$$

$$x_{ij}^{(k)} : 0/1.$$

As is evident, optimization problem (8) is a nonlinear integer programming problem, which is generally NP Hard. Hence, we propose the following iterative heuristic for it, which is based on the idea presented below.

Assume that we have solved the assignment problem for day 1. Then, if worker *i* is assigned to task *j* on day 1, we can discourage his/her assignment to the same job on the next days by increasing the cost c_{ij} to an appropriately large amount for the next day's assignment problem. The increase must be large enough to capture the desire of the decision maker to avoid consecutive assignments over two successive days; for example, if this assignment is prohibited, then the cost should be set to infinity. If this assignment is not preferred for the next *DAYS-OF-ROTN* days, then this increase has to be done for as many days and in a manner that reflects the decision maker's inherent desire to avoid repetitions within any consecutive *DAYS-OF-ROTN* days. This is the central idea behind the heuristic. Basically, the heuristic will solve *K* different assignment problems successively (one for each day), with *K* different cost matrices. In order to present this heuristic in an algorithmic format, assume that we define *K* cost assignment matrices named $C^{(1)}, \ldots, C^{(k)}, \ldots, C^{(K)}$, where $C^{(k)}$ is the cost assignment matrix that will be used to solve the assignment model for the *k*th day and will be obtained on the basis of assignments that are made on the (*k*-1)th day. The algorithm for the heuristic is presented below.

4.1 Avoid-Consecutive-Repetitions algorithm

Input: Cost of Assignment Matrix $C = \{c_{ij}\}$, Workers: $\{i = 1, 2, \ldots, I\}$ and Tasks: $\{j = 1, 2, \ldots, J\}$

Output: $x_{ij}^{(k)} \in \{0, 1\}$ for $i = 1, 2, \ldots, I$, $j = 1, 2, \ldots, J$ and $k = 1, 2, \ldots, K$.

Initialize all cost matrices as follows: $C^{(1)} = C^{(2)} = \ldots C^{(k)} = \ldots C^{(K)} = C$

for ($k = 1, 2, 3, \ldots, K$) **do**

{

Solve the assignment model for these *I* workers to these *J* tasks for the *k*th days, using cost matrix $C^{(k)}$. Examine the assignment obtained.

for ($i = 1, 2, 3, \ldots, I$) **do**

{

for ($j = 1, 2, 3, \ldots, J$) **do**

{

if ($x_{ij}^{(k)} = 1$) **then**

{

Modify $c_{ij}^{(k+1)}, \ldots, c_{ij}^{(k+DAYS-OF-ROTN)}$ in the cost matrices $C^{(k+1)}, \ldots, C^{(k+DAYS\text{-}OF\text{-}ROTN)}$ as per a given rule that captures the desire of the decision maker to avoid consecutive assignments.

}
}
}
}

4.2 Implementation of Avoid-Consecutive-Repetitions algorithm

The *Avoid-Consecutive-Repetitions* algorithm was implemented as a user-friendly program in Microsoft Excel. Macros using customized Visual Basic were utilized to automate execution of the Solver Add-in for each consecutive day. 'If' statements were used to create an algorithm to adjust the actual cost of the worker-job assignment based on whether the worker had been assigned to that job in previous days. The number of workers and/or jobs was allowed to range from two to 12. The schedule could be customized to complete a schedule from one to ten days. Finally, the application could be adjusted to change the *DAYS-OF-ROTN* parameter from 1 to 5 days with different rules for changing c_{ij} to avoid consecutive assignments. The final algorithm was then implemented via a separate macro written in Visual Basic. This MS-Excel implementation was empirically tested on a complete grid of costs for 12 workers and 12 jobs. Four sets of random costs were generated, each in the interval (1, 10). The average cost ranged from 5.2 to 5.8 and the standard deviation ranged from 2.7 to 2.9.

4.3 Impact of changing the DAYS-OF-ROTN value

Our first set of empirical tests was to investigate the effect of imposing a greater amount of rotation (by increasing the value of *DAYS-OF-ROTN*) on total scheduling costs; intuitively, we would expect the cost of the assignment to increase. We confirmed this in our observations. Figures 3 and 4 present the average increase in costs of all randomly generated data sets in cases where *DAYS-OF-ROTN* = 1 and 3, respectively. Simulation averages with the random data sets that assumed *DAYS-OF-ROTN* = 1, i.e. rotating the assignments every other day, are summarized in figure 3, and as can be seen therein, the average costs in our simulation ranged from 111 to 128% of the optimum costs (no rotation) after just 10 days of schedule. By contrast, the simulation averages shown in figure 4 indicate a larger increase in scheduling costs when we increased the amount of rotation by making *DAYS-OF-ROTN* = 3.

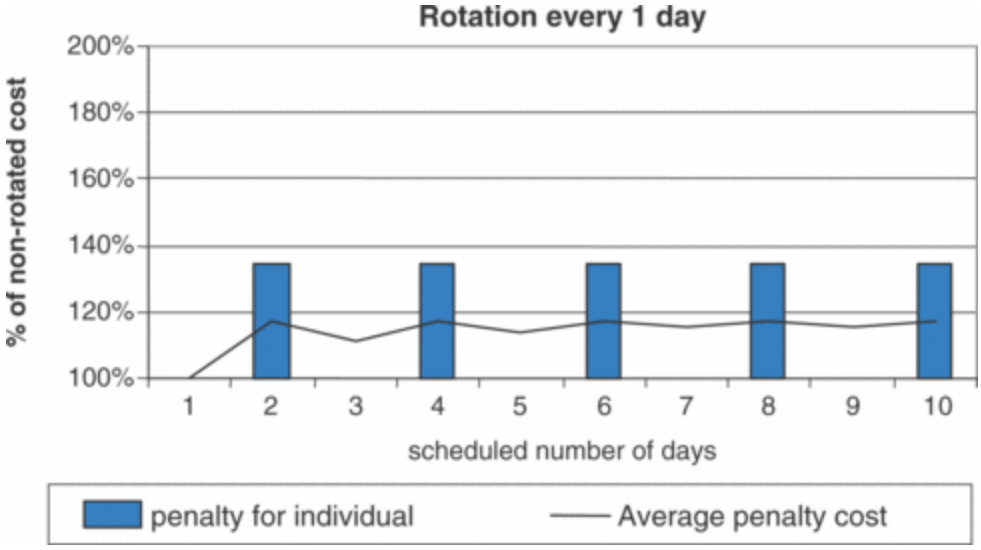

Figure 3. Cost penalty for *DAYS-OF-ROTN* = 1.

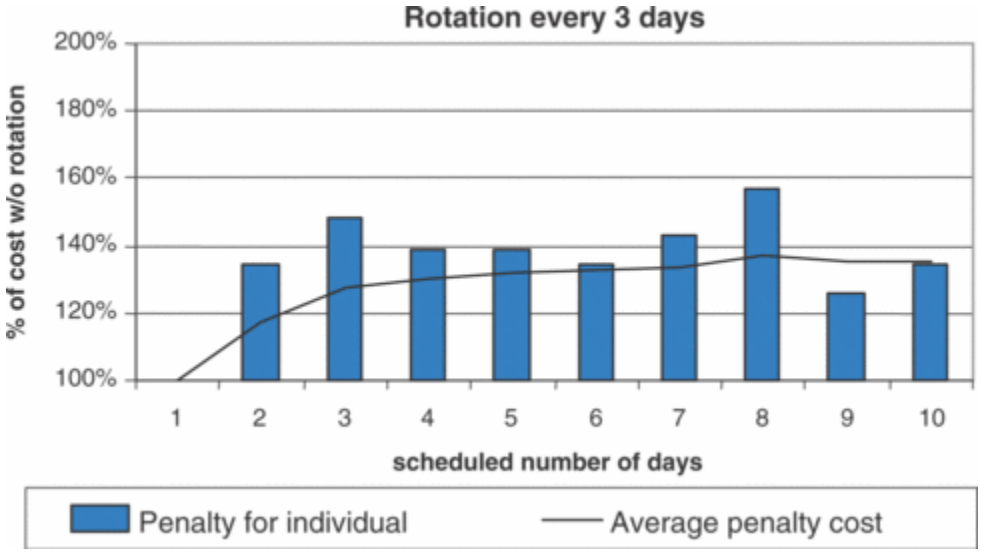

Figure 4. Cost penalty for *DAYS-OF-ROTN* = 3.

4.4 Impact of changing workers to tasks ratio

Next, we investigated the impact of changing the ratio of workers to jobs/tasks. Our empirical results indicated that having more workers than jobs leads to a higher rise in costs (relative to the cost where rotation is not required) than when the opposite is the case. Consider a typical example that we show in figure 5, which is for a specific instance of the problem where we took a (12 worker ×12 job) problem and tested with two fewer jobs than workers and then with two fewer workers than jobs.

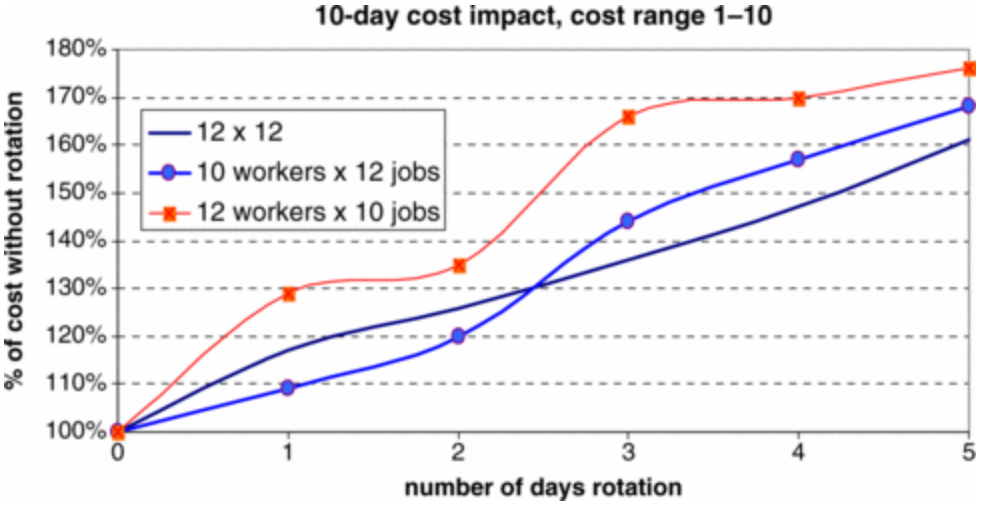

Figure 5. Impact of changing workers to job ratio.

5. Model 3. A hybrid of Models 1 and 2

In this section, we will discuss our third and final variant of the generic model given in the second section. This version is a hybrid of the models in the previous two sections. It attempts to simultaneously restrict the maximum number of assignments of a worker to any job and also the maximum number of consecutive assignments of a worker to the same job. To formalize the development of this model, we define $T(i, j)$ of worker i on job j as the total number of assignments of worker i to job j:

$$T(i, j) = \sum_k X_{ij}^k.$$

Then, one of the boredom measures of worker i will be the maximum number of assignments to a specific job j in k periods, defined as $T(i) = \max\{T(i, j)\}$. We now impose the condition that the maximum number of assignments should not exceed a predetermined limit of T_m assignments for any job in a set. Although we assume for the sake of simplicity that the same limit will be used for all scheduled workers, the model and the solution proposed can be used when this assumption is not true. Thus by our assumption, no worker should be assigned to any job more than T_m times in a k-period schedule. Along the same lines, we also impose the additional condition that the maximum number of consecutive assignments of worker i to job j, which is another measure of boredom, should not exceed a predetermined limit of B_m. Since this hybrid model combines the essential ideas of Models 1 and 2, we would expect that its results, while different in general from either of the Models 1 and 2, will be of more use to practitioners. Nonetheless, the mathematical formulation of the hybrid model is as follows:

$$\begin{aligned}
&\min \sum_i \sum_j \sum_k C_{ij} X_{ik}^k, \\
&\text{subject to} \\
&\sum_i X_{ij}^k = 1 \quad \forall j, k, \\
&\sum_j X_{ij}^k = 1 \quad \forall i, k, \\
&B(i) \le B_m \quad \forall i, \\
&T(i) \le T_m \quad \forall i.
\end{aligned} \tag{9}$$

To solve the model described in problem formulation (9), we propose the following heuristic algorithm. This algorithm is based on employing the assignment model in each period and using a heuristic approach to assess the boredom level of a worker in that period. If at any scheduled period a worker reaches the limit of consecutive assignments for a specific job, then s/he is prohibited from the assignment to this job in the next period (the cost of such an assignment is also substantially increased). Also, if at any point of time in the schedule, a worker reaches the maximum limit of assignments to a specific job, then s/he is prohibited from doing this job for the rest of the scheduled periods. This algorithm is described below.

5.1 Hybrid-Model algorithm

Input: Cost of Assignment Matrix $C = \{c_{ij}\}$, Workers: $\{i = 1, 2, \dots, I\}$ and Tasks: $\{j = 1, 2, \dots, J\}$

Output: $x_{ij}^{(k)} \in \{0, 1\}$ for $i = 1, 2, \dots, I$, $j = 1, 2, \dots, J$ and $k = 1, 2, \dots, K$.

begin

{

Step 1. In the first period ($k = 1$), all workers are numbers from 1 through I, and all jobs are numbered from 1 through J. We assume that $I = J$, otherwise a dummy number of workers (I - J) or jobs (J - I) may be added. Assign the workers to jobs by employing the assignment model.

while ($k \leq K$) **do**

{

Step 2. At any period k in the schedule, use the assignment model to identify the minimum cost assignment schedule.

Step 3. Revise the cost of assignments for the next period (k + 1). To do that, for each worker i identify the two parameters of boredom: $B(i)$, the maximum number of consecutive assignments, and $T(i)$, the maximum number of total assignments for a job. If both parameters are less than the limits, B_m and T_m, respectively, then the cost of assignments of worker i to job j for the next period (k + 1) will be the same, C_{ij}. If either of these parameters reached the limit, then the cost of assignments will be M, or a huge positive number substantially greater that the actual costs:

$$C_{ij}^{k+1} = \begin{cases} C_{ij}, & \text{if } B(i) \quad lt;\ B_m \text{ and } T(i) \quad lt;\ T_m, \\ M, & \text{if } B(i) = B_m \text{ or } T(i) = T_m. \end{cases}$$

}

}

End

5.2 Implementation of the Hybrid-Model algorithm

To identify the performance of the *Hybrid-Model* algorithm and prove its effectiveness, this algorithm was implemented in MS Excel. We developed a macro written in Visual Basic, which applied the Excel's Solver add-in to identify optimal assignments in each period. As in the case of the previously discussed algorithm Avoid-Consecutive-Repetitions, the Hybrid-Model algorithm was also empirically tested on a grid of costs for 12 workers and 12 jobs for a 10-day period. Four sets of random costs were generated with the average cost ranging from 5.5 to 6.2 and the standard deviation varying from 3.4 to 4.1.

The algorithm was tested first with only one boredom constraint, which was the maximum number of consecutive assignments, B_m. In this case, we considered B_m equal to 2. The rotation

results in terms of the penalty cost increase are shown in figure 6. This penalty cost (see also section 4.2 of this paper) is caused by an increase in the total cost of assignments over the minimum cost in the original solution where no rotation is required. As can be seen from figure 6, there is no penalty cost for the first two periods, because the boredom constraint, B_m, is equal to 2 and, thus, allows no rotation (resulting in no penalty cost) in the first two periods. The penalty cost appears in period 3, when the boredom constraint (in this case, maximum number of consecutive assignments) will require rotation, which, in turn, results in a penalty cost. The bigger the penalty cost increase, the more rotation is taking place, and the less the boredom of employees. As can be seen from figure 6, the cost increase on any particular day varied from 0 to slightly above 10% (11.4% on average for all simulation runs) with the average increase between 0 and 7.6% in a 10-day period.

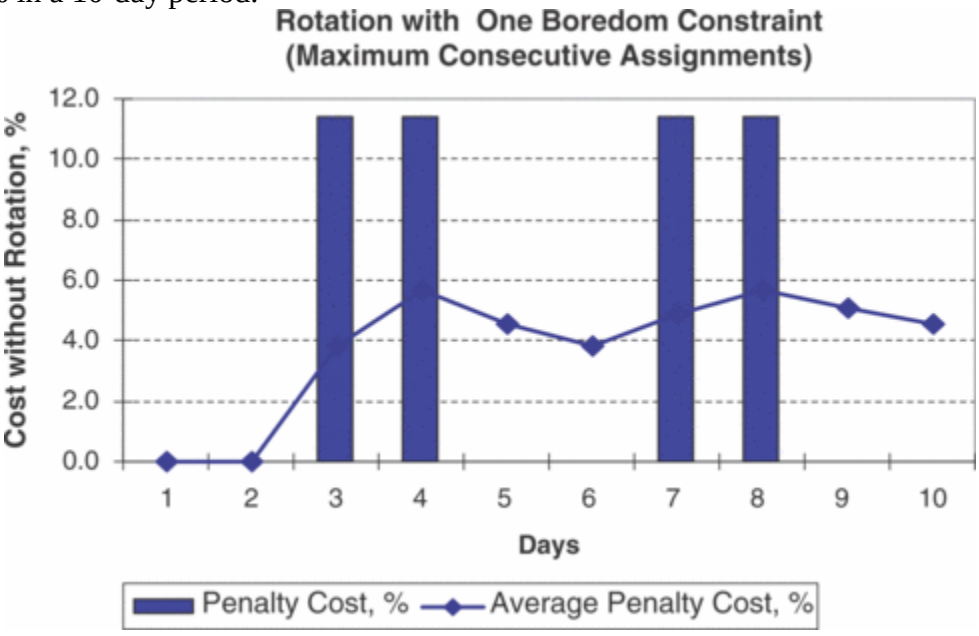

Figure 6. Penalty cost in rotation with maximum consecutive assignments' constraint.

The Hybrid-Model algorithm was also tested with both boredom constraints, i.e. the maximum number of consecutive assignments, B_m, and maximum number of assignments of an employee per job, T_m. In this case, we considered $B_m = 2$ and $T_m = 3$. The results of this test are shown in figure 7. As described before, there is no penalty cost for the first two periods (see figure 7), because rotation is not required by either of the constraints in the first two periods of this test. However, the penalty cost will appear in period 3, when the first constraint B_m necessitates rotation resulting in the incurrence of a penalty cost.

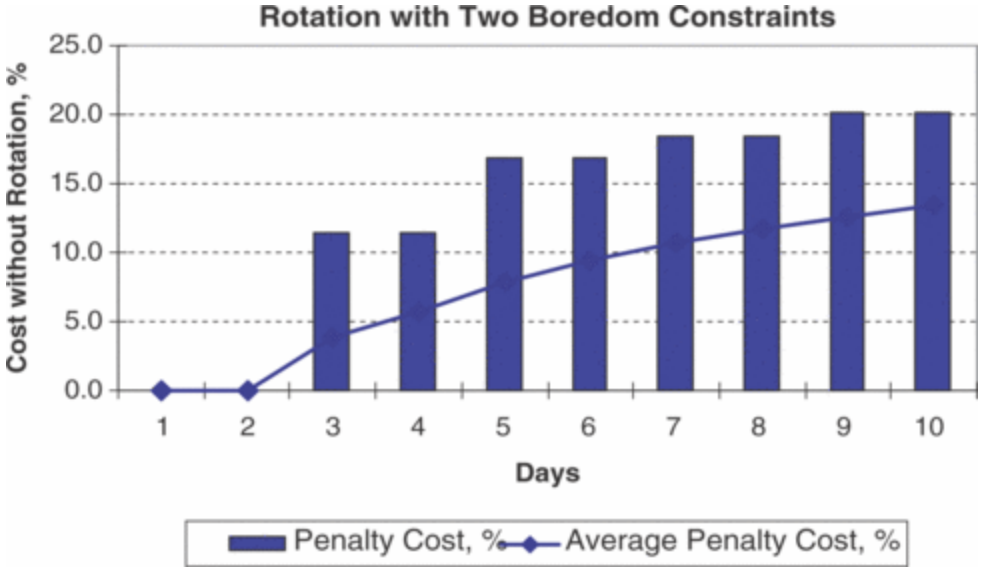

Figure 7. Penalty cost in rotation with two boredom constraints.

As can be seen from figure 7, the daily cost increase varies from 0 to slightly above 20% (20.2%) with the average cost increase from 0 to 13.4%. This average cost increase is about 80% higher than that in the first test with only one boredom constraint. Applying two constraints definitely leads to more rotations and less overall boredom of employees.

The hybrid model presented in formulation (9) can be easily adapted to accommodate a wide range of additional constraints that arise from practice. For example, a worker may not be assigned to a specific job, because of a possible lack of skills or education. In this case, introducing a substantially higher cost for these assignments will resolve the problem. Sometimes, because of specific technological and psychological conditions, a worker cannot be rotated from one job to another. Hence, not all jobs can be alternated in a schedule. A special matrix of permitted rotations may be developed to catch and resolve such problems. Usually, a worker requires some time to adjust to the working conditions associated with each job. The duration of such time is a constraint in the job rotation assignments, because a worker should do a job at least some minimal time. Besides, the number of rotations per shift may not be greater than a specified limit. All such constraints can easily be incorporated into the model.

6. Conclusion

In this paper, we have attempted to address the gap in the published literature on quantitative tools to enable a manager to rotate workers through jobs in a meaningful manner that balances cost of assignment with the boredom felt by the workers. We began by proposing a generic version of this problem as a multi-period assignment model that examines the problem of assigning workers to jobs over a fixed number of periods where it is required to (i) minimize the cost of the assignment and (ii) rotate workers through jobs such that their boredom from repetition is minimized. This was then formulated as a bi-objective optimization problem. Having done that, we presented three different versions of our generic model, each of which measured boredom in a different way. For each such version, we either developed a polynomial time algorithm or proposed a heuristic. In the case of heuristics, we also presented their implementation in MS-Excel and discussed the results of empirical testing.

Future research in this area should examine more refined models that take into account other realistic constraints that managers face in today's organizations, such as budgetary constraints, dynamically changing jobs and workforce skills, etc. Also of interest would be to conduct studies where we study the impact of our proposed models on assignments in a real organization to see if the usage of quantitative tools such as ours has a significant impact on outcomes such as employee satisfaction, motivation and performance. Such a study could also examine the employee characteristics that are the most predictive of scheduling cost increases or decreases when job rotation is required.

References

- 1. Ahuja, RK, Magnanti, TL and Orlin, JB (1993) *Network Flows: Theory, Algorithms and Applications* Prentice Hall , Englewood Cliffs, NJ
- 2. Anselmi, F. and Sundarrajan, S. (2000) Aligning workers and workloads. *IIE Sol.* pp. 28-32.
- 3. Al-Subhi, AK, Al-Sinan, M. and Shokri, ZS (1997) A multi-objective linear program for scheduling repetitive projects. *Ing. Econ.* **60/61** , pp. 17-26.
- 4. Cunningham, BJ and Eberle, T. (1990) A Guide to job enrichment and redesign. *Personnel* pp. 56-61.
- 5. Davis, LE and Taylor, JC (eds) (1989) *Design of Jobs* Goodyear Publishing , Santa Monica, CA
- 6. Dulin, E. (1998) Enterprise and plant-centric scheduling. *IIE Sol.*
- 7. Gittleman, M., Horrigan, M. and Joyce, M. (1998) Flexible workplace practices: evidence from a nationally representative survey. *Ind. Labor Rel. Rev.* **52** , pp. 99-115.
- 8. Griffin, RW (2002) *Management* Houghton Mifflin , Boston
- 9. Konz, SA (1983) *Work Design: Industrial Ergonomics* Grid , Columbus, OH
- 10. Krajewski, LJ and Ritzman, LP (1999) *Operations Management: Strategy and Analysis* Addison-Wesley , New York
- 11. Moorehead, G. and Griffin, RW (2001) *Organizational Behavior: Managing People and Organizations* Houghton Mifflin , Boston
- 12. Niebel, BW (2003) *Motion and Time Study* Erwin , Burr Ridge, IL
- 13. Ortega, J. (2001) Job rotation as a learning mechanism. *Management Science* **47** , pp. 1361-1370. [crossref]
- 14. Osterman, P. (1994) How common is workplace transformation and who adopts it?. *Ind. Labor Rel. Rev.* **47** , pp. 173-188. [crossref]
- 15. Osterman, P. (2000) Work reorganization in an era of restructuring: trends in diffusion and effects on employee welfare. *Ind. Labor Rel. Rev.* **53** , pp. 179-196. [crossref]
- 16. Pinedo, M. (1995) *Scheduling: Theory, Algorithms and Systems* Prentice Hall , Englewood Cliffs, NJ
- 17. Shtub, A., Leblanc, LJ and Cai, Z. (1996) Scheduling problems with repetitive projects: a comparison of a simulated annealing, a genetic and pair-wise swap algorithm. *Eur. J. Oper. Res.* **88** , pp. 124-138. [crossref]
- 18. Taha, HA (1997) *Operations Research* Prentice Hall , Englewood Cliffs, NJ

- 19. Thompson, GM (1999) Labor scheduling, Part 4. *Controlling Workforce Schedules in Real-time Cornell Hotel and Restaurant Administration Quarterly* **40** , pp. 85-96. [crossref]