COMPRESSED WORKWEEK SCHEDULING WITH CONSECUTIVITY, FREQUENCY AND STRETCH CONSTRAINTS

Hesham K. Alfares

King Fahd University of Petroleum & Minerals

PO Box 5067, Dhahran 31261, Saudi Arabia

hesham@ccse.kfupm.edu.sa

ABSTRACT

In this paper we consider a three-day workweek scheduling problem with three realistic days-off scheduling constraints: (1) at least two off days per week must be consecutive, (2), employees must get a given proportion of weekends off, and (3) the number of consecutive workdays in any work stretch cannot exceed four. An integer programming model is formulated and efficiently solved by an algorithm that involves three stages: (1) determining the minimum workforce size by primal-dual relations, (2) adding a workforce-size constraint to the integer programming model to expedite its solution, and (3) constructing fair and feasible multiple-week rotation schedules.

Key Words: Scheduling, Mathematical programming, Optimisation

1. INTRODUCTION

Workforce scheduling is especially significant for facilities that operate continuously, such as hospitals, restaurants, and train stations. If the organization operates seven days a week, then different employees must be assigned different days-off, some of which do not necessarily correspond to the weekend. The problem in this case is called the days-off scheduling, and its objective is to find the minimum size or cost of the workforce to satisfy the daily labour demand for every day of the week.

Most of the earlier days-off scheduling literature was directed towards the (5,7) problem, or the traditional 5 workdays per week. Recently, there has been much emphasis on compressed and flexible work schedules. McCampbell (1996) describes the phenomenal growth of alternative work schedules in the U.S. and lists alternative work schedules suggested by the U.S. Office of Personnel Management. This list includes three compressed-schedule modules: the 3-day workweek, the 4-day workweek, and the 5-4/9 plan.

The days-off scheduling problem considered in this paper involves a three-day workweek. Three types of days-off constraints are imposed: (1) the maximum length of any work stretch cannot exceed four days, (2) at least two of the four off days per week must be consecutive, and (3) each employee must get a minimum proportion of full weekends off. An integer programming (IP) model will be formulated to represent the three types of constraints, and a three-stage algorithm will be developed to minimize the number or cost of the workforce.

In the first stage of the algorithm, the primal-dual relationships are used to determine the minimum workforce size. In the second stage, a workforce-size constraint is appended to the IP model to efficiently obtain optimum integer solutions. Finally, multiple-week rotation schedules will be constructed to guarantee that work patterns assigned to each employee in

successive weeks satisfy all the constraints. The computational efficiency of the new method will be compared to existing integer programming models.

The rest of this paper is organized as follows. First, a review of relevant literature is given. Then, the integer programming model of the problem is presented. Next, the solution algorithm is described. Finally, computational comparisons are presented and conclusions are given.

2. LITERATURE SURVEY

This literature survey is concerned with recent compressed workweek scheduling, especially the three-day workweek problem. Ernst et al. (2004) provide the most comprehensive and recent survey of personnel scheduling algorithms. Hung (1991) analyses two compressed workweek scheduling models assuming that each employee must receive at least *A* out of *B* weekends off. Hung (1993) develops similar multiple-shift models for 3-day workweeks, whose objective is to minimize the workforce size. Under the same assumptions, Narasimhan (1997) considers days-off scheduling for a hierarchical workforce, where each employee cannot be assigned more than 5 consecutive working days.

Hung (1994) develops an algorithm for hierarchical workforce scheduling under variable labour demand and employee substitution, assuming the number of workdays per week can be 3, 4, or 5 days. Billionnet (1999) uses integer programming to formulate and efficiently solve the same problem. Burns and Narasimhan (1999) present a multiple-shift algorithm for 3-day and 4-day workweek scheduling of a homogeneous workforce. Constraints are imposed on weekend work frequency, work stretch length, and transition time when changing shifts. Narasimhan (2000) considers the same problem but for a hierarchical workforce consisting of several employee categories.

Burns et al. (1998) present an algorithm for 3-day and 4-day workweeks, with variable daily demand and limits on work stretch lengths. Lankford (1998) describes an actual pilot implementation of 4-day workweek schedule at the he Analytical Central Call Management (CCM) group at Hewlett Packard. Bard et al. (2003) present a model for employee tour scheduling at the US Postal Service, which can handle different days off policies, variable start times, and the use of part-time flexible workers. Alfares (2003a) presents an optimal algorithm for four-day workweek scheduling with weekend work frequency constraints. Alfares (2003b) provides manual optimal solutions for the 3-day and the 4-day workweek scheduling problems, assuming two levels of labour demands and consecutive workdays.

The algorithm presented in this paper produces an optimal solution to single-shift three-day workweek scheduling problem under the following assumptions:

(i) the demand for employees may vary from day to day for the given week,
(ii) each employee is assigned 3 workdays and 4 off-days per week,
(iii) at least 2 of the 4 weekly off-days must be consecutive,
(iv) the maximum work stretch is 4 consecutive workdays, and
(v) each employee takes on average a proportion *P* of *full weekends* off.

3. INTEGER PROGRAMMING FORMULATION

The three-day workweek scheduling problem described above can be represented by the following integer linear programming model:

$$\text{Minimize } W = \sum_{j=1}^{35} x_j \qquad (1)$$

subject to

$$\sum_{j=1}^{35} a_{ij} x_j \geq r_i, \qquad i = 1, 2, ..., 7 \tag{2}$$

$$\frac{\sum_{j\in J_2} x_j}{\sum_{j=1}^{35} x_j} \geq P, \quad \text{or} \quad -P\sum_{j\in J_0} x_j - P\sum_{j\in J_1} x_j + (1-P)\sum_{j\in J_2} x_j \geq 0 \tag{3}$$

$$x_1 \leq MQ_1 \tag{4}$$

$$\sum_{j\in E} x_j \leq MQ_2 \tag{5}$$

$$Q_3 \leq \frac{1}{2}(Q_1 + Q_2) \tag{6}$$

$$Q_3 \geq Q_1 + Q_2 - 1 \tag{7}$$

$$\sum_{j=1}^{35} x_j - \sum_{j\in E} x_j - x_1 \geq Q_3 \tag{8}$$

$$x_4 \leq MQ_4 \tag{9}$$

$$\sum_{j\in J_0} x_j \leq MQ_5 \tag{10}$$

$$Q_6 \leq \frac{1}{2}(Q_4 + Q_5) \tag{11}$$

$$Q_6 \geq Q_4 + Q_5 - 1 \tag{12}$$

$$\sum_{j=1}^{35} x_j - \sum_{j\in J_0} x_j - x_4 \geq Q_6 \tag{13}$$

$$x_j \geq 0 \text{ and integer}, \qquad j = 1, 2, ..., 35 \tag{14}$$

$$Q_j = 0 \text{ or } 1, \qquad j = 1, 2, ..., 6 \tag{15}$$

where

W = workforce size, i.e., total number of employees assigned to all patterns
x_j = number of employees assigned to weekly days-off work pattern j
$a_{ij} = 1$ if day i is a work day for pattern j,
$a_{ij} = 0$ otherwise ($i = 1, 2, ..., 7$)
$a_{8j} = -P$ if $j \in J_0$ or $j \in J_1$,
$a_{8j} = 1 - P$ if $j \in J_2$,

Table 1 shows matrix $A = \{ a_{ij}, i = 1, ..., 8, j = 1, ..., 35\}$
r_i = minimum number of employees required on day i, $i = 1, 2, ..., 7$, $r_8 = 0$
P = average proportion of full weekends off, $0 \leq P \leq 1$
M = any large number, $M \geq W$
E = set of days-off patterns in which both days 1 and 2 are workdays,
$E = \{3, 4, 9, 18, 23, 33\}$
J_k = set of days-off patterns with k weekend days off per week, $k = 0, 1, 2$
$J_0 = \{1, 2, 14, 16, 28, 31\}$
$J_1 = \{3, 7, 8, 9, 10, 13, 15, 17, 20, 21, 22, 24, 25, 27, 30, 32, 34, 35\}$
$J_2 = \{4, 5, 6, 11, 12, 18, 19, 23, 26, 29, 33\}$

The objective (1) is to minimize the workforce size. Constraints (2) ensure that the number of employees assigned for the given day i are at least equal to the total number of employees required for that day. The denominator on the left-hand side of (3) represents the total number of employees assigned for the given weekly schedule, while the numerator represents the number of employees with full weekends off.

Constraint sets (4)-(8) and (9)-(13) ensure that no more than 4 successive workdays are assigned as work patterns are linked from one week to the next during the rotation cycle. Constraints (4)-(8) ensure that work pattern 1, characterized by 3 consecutive workdays at the end of a given week, is not immediately followed by any work pattern of the set E, in which

both days 1 and 2 are workdays. Similarly, constraints (9)-(13) ensure that any work pattern of the set J_0, in which both days 6 and 7 are workdays, is not immediately followed by work pattern 4, which has 3 consecutive workdays at the start of a given week.

Table 1. Days-off matrix $A = \{a_{ij}\}$ and cost vector $C = \{c_j\}$ for the 35 days-off work patterns

	j 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
i																		
1	0	**1**	**1**	**1**	0	0	0	**1**	**1**	0	**1**	0	0	0	**1**	0	**1**	**1**
2	0	0	**1**	**1**	**1**	0	0	0	**1**	**1**	0	**1**	0	0	0	**1**	0	**1**
3	0	0	0	**1**	**1**	**1**	0	0	0	**1**	**1**	0	**1**	0	0	0	**1**	0
4	0	0	0	0	**1**	**1**	**1**	0	0	0	**1**	**1**	0	**1**	0	0	0	**1**
5	**1**	0	0	0	0	**1**	**1**	**1**	0	0	0	**1**	**1**	0	**1**	0	0	0
6	**1**	**1**	0	0	0	0	**1**	0	**1**	0	0	0	**1**	**1**	**1**	**1**	0	0
7	**1**	**1**	**1**	0	0	0	0	**1**	0	**1**	0	0	0	**1**	0	**1**	**1**	0
8	$-P$	$-P$	$-P$	$1-P$	$1-P$	$1-P$	$-P$	$-P$	$-P$	$-P$	$1-P$	$1-P$	$-P$	$-P$	$-P$	$-P$	$-P$	$1-P$
c_j	3 + 2β	3 + 2β	3 + β	3	3	3	3 + β	3 + β	3 + β	3 + β	3	3	3 + β	3 + 2β	3 + β	3 + 2β	3 + β	3

	j 19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
i																	
1	0	0	0	**1**	**1**	0	0	**1**	0	0	**1**	0	0	**1**	**1**	0	0
2	**1**	0	0	0	**1**	**1**	0	0	**1**	0	0	**1**	0	0	**1**	**1**	0
3	**1**	**1**	0	0	0	**1**	**1**	0	0	**1**	0	0	**1**	0	0	**1**	**1**
4	0	**1**	**1**	**1**	0	0	**1**	**1**	0	0	**1**	0	0	**1**	0	0	**1**
5	**1**	0	**1**	0	**1**	0	0	**1**	**1**	0	**1**	**1**	0	0	**1**	0	0
6	0	**1**	0	0	0	**1**	0	0	**1**	**1**	0	**1**	**1**	0	0	**1**	0
7	0	0	**1**	**1**	0	0	**1**	0	0	**1**	0	0	**1**	**1**	0	0	**1**
8	$1-P$	$-P$	$-P$	$-P$	$1-P$	$-P$	$-P$	$1-P$	$-P$	$-P$	$1-P$	$-P$	$-P$	$-P$	$1-P$	$-P$	$-P$
c_j	3	3 + β	3 + β	3 + β	3	3 + β	3 + β	3	3 + β	3 + 2β	3	3 + β	3 + 2β	3 + β	3	3 + β	3 + β

4. SOLUTION ALGORITHM

An efficient algorithm is developed to solve the problem optimally in three stages.

4.1. Determining the Minimum Workforce Size

Given a week's daily labour demands, the minimum workforce size W can be determined from cyclical enumeration of all dual solutions and using primal-dual relations, without integer programming. Depending on the demands r_1, ..., r_7, there are four dominant dual solutions which produce the following lower bounds on the workforce size:

$$W \geq r_{\max}$$
$$W \geq (\textstyle\sum_{i=1,\ldots,7} r_i)/3$$
$$W \geq (r_i + r_{i+2} + r_{i+4})/2, \quad i = 1, 2, \ldots, 7$$
$$W \geq \max(r_6, r_7)/(1 - P)$$

To determine W, we choose the maximum value obtained from the four above bounds, and round it up to the nearest integer. Therefore, we obtain the following expression for the minimum workforce size W:

$$W = \max \{ r_{\max}, \lceil \frac{1}{3} \sum_{i=1}^{7} r_i \rceil, \lceil \frac{R_{max}}{2} \rceil, \lceil \frac{\max(r_6, r_7)}{1-P} \rceil \} \quad (16)$$

where

$r_{\max}$ = max { r_1, r_2, ..., r_7 }
$R_{\max}$ = max { R_1, R_2, ..., R_7 }
$\lceil a \rceil$ = smallest integer $\geq a$,
$R_i = \sum_{j \in S_i} r_i$, i = 1, 2, ..., 7
s_i = { i, i + 2, i + 4 }, i = 1, 2, ..., 7

where s_i is circular set with a cycle = 7 (see Table 2)

Table 2. Sets of subscripts s_i

i	s_i
1	1, 3, 5
2	2, 4, 6
3	3, 5, 7
4	1, 4, 6
5	2, 5, 7
6	1, 3, 6
7	2, 4, 7

4.2. Days-Off Assignments

After determining the workforce size W by (16), the objective at this stage is to assign the W employees to different days-off patterns in order to minimize total cost. Assuming each employee is paid 1 unit per regular workday and 1+β units ($\beta \geq 0$) per weekend workday, the weekly costs of the 35 days-off patterns {c_1, c_2,..., c_{35}} are shown in Table 1. Changing the objective to that of minimizing total cost, (1) is replaced by:

$$\text{Minimize } Z = \sum_{j=1}^{35} c_j x_j \quad (17)$$

where
c_j = weekly cost of days-off pattern j per employee, shown in Table 1.

Introducing these costs in the IP model, the workforce size W does not change. This means that for the cost structure defined by {c_1, c_2, ..., c_{35}}, for all $\beta \geq 0$, the minimum *cost* is always obtained with the minimum *number* of employees. Given r_1, ..., r_7 , β, and P, the value of the minimum workforce size W is first calculated by (16), and then the following constraint is added to the primal integer programming model defined by (2)-(15) and (17).

$$\sum_{j=1}^{35} x_j = W \quad (18)$$

4.3. Rotation Schedules

A rotation scheme is now introduced to ensure that all constraints are satisfied as employees switch from one pattern to another over a multiple-week rotation period. The solution of the integer programming model specifies the number of employees assigned to each days-off pattern (x_1, ..., x_{35}) and the total workforce W (x_1 + ... + x_{35}) for a single week.

First, we define a W-week rotation cycle, during which each employee is assigned to each pattern j for a period of x_j weeks (j = 1, ..., 35). All W employees must go through the same sequence of assignments to patterns over this W-week rotation cycle, but employee i starts the sequence at the ith week of the cycle. For feasibility, assignments to pattern 1 cannot be

followed by patterns belonging to set E, and assignments to J_0 patterns cannot be followed by pattern 4.

5. COMPUTATIONAL RESULTS

In order to evaluate the effect of adding constraint (18), computational experiments were carried out using 252 test problems. In all these problems, the value of β was set at 0.5 to indicate 50% higher pay for weekend work. Moreover, the value of P was set at 0.5 to indicate a requirement of every other weekend off.

The 252 test problems, partially described by Alfares (1998), are divided into 12 sets with different demand types, but all have an average demand of 50 employees per day. The first six sets involve 17 problems each, while the last six sets involve 25 problems each. Sets 1-6 have a demand range of 34 to 64, and different specific labour demand patterns: level, trend, concave, convex, unimodal, and sinusoidal. Sets 7-10 have randomly distributed labour demands over the intervals: [34, 66], [0, 100], [20, 80], and [45, 55], respectively. Sets 11-12 involve two constant levels of labour demand, workdays demand ($r_1 = r_2 \ldots = r_5 = D$) which is fixed at 50, and weekends ($r_6 = r_7 = E$). Set 11 has E = 25, …, 49, or $E < D$, while set 12 has E = 51, …, 75 or $E > D$.

Microsoft Excel integer programming Solver® with default options setting, was used on a 450-MHz Pentium III PC to solve the 252 test problems. The results of computational experiments are summarized in Table 3. Without constraint (18), 135 problems could not even be solved within the 100 second time limit. For these problems, the solution time was simply assumed to be 100 seconds.

Overall, average solution time dropped by 120 times, from 55.02 seconds to 0.46 seconds. Maximum solution time fell by 60 times, from 100 seconds to only 1.65 seconds. Similarly, the standard deviation decreased from 49.43 seconds to only 0.22 seconds. Clearly, adding constraint (18) leads to a remarkable reduction in both the mean and variation of solution times.

Table 3. Solution times in seconds with and without constraint (18)

Problem set	Number 0f problems	Without constraint (18)*					With constraint (18)			
		Min	Ave.*	Max*	Std dev*	> 100	Min	Ave.	Max	Std dev
1	17	0.25	64.41	100	48.54	10	0.26	0.46	1.16	0.25
2	17	0.28	68.82	100	46.12	11	0.28	0.55	1.43	0.34
3	17	0.25	63.31	100	48.21	10	0.28	0.45	0.85	0.20
4	17	0.26	64.90	100	48.99	11	0.27	0.42	0.65	0.13
5	17	0.26	70.74	100	46.73	12	0.28	0.42	0.90	0.20
6	17	0.25	64.98	100	48.87	11	0.26	0.50	1.09	0.30
7	25	0.25	56.13	100	50.52	14	0.26	0.41	0.98	0.23
8	25	0.25	20.24	100	40.70	5	0.23	0.31	0.47	0.06
9	25	0.25	32.19	100	47.47	8	0.25	0.32	0.57	0.08
10	25	0.27	76.12	100	43.37	19	0.26	0.43	0.90	0.19
11	25	0.27	68.10	100	47.47	17	0.28	0.41	0.94	0.18
12	25	0.27	31.80	100	46.93	7	0.28	0.45	1.65	0.29
Overall	252	**0.25**	**55.02**	**100**	**49.43**	**135**	**0.23**	**0.46**	**1.65**	**0.22**

* Some solution could not be obtained in 100 seconds

6. CONCLUSIONS

An optimization algorithm for a compressed workweek days-off scheduling problem with weekend work frequency constraints has been developed. In this problem, employees are given four off days per week, out of which at least two days must be consecutive. Moreover,

employees must be given a certain proportion of weekends off, and cannot be assigned to a work stretch of more than four consecutive workdays. An integer programming model has been formulated, and an efficient solution algorithm has been developed. The three-stage algorithm involves determining the minimum workforce size, assigning employees to days-off, and developing cyclic multiple-week rotation schedules.

Based on computational experiments with 252 test problems, the addition of a workforce-size constraint to the IP model has been found essential for efficient solution. Without this constraint, more than 54% of the problems could not be to be solved. The addition of this constraint has remarkably reduced solution times and has made it possible to solve all test problems. On average, the appended model has been found to be at least 150 times faster than the traditional IP model.

ACKNOWLEDGEMENTS

The author is indebted to King Fahd University of Petroleum & Minerals for supporting this research effort.

REFERENCES

Alfares, H.K. (1998), An efficient two-phase algorithm for cyclic days-off scheduling, *Computers & Operations Research* **25**, 913-923.

Alfares, H.K. (2003a), Flexible four-day workweek scheduling with weekend work frequency constraints, *Computers & Industrial Engineering* **44**, 325-338.

Alfares, H.K. (2003b), Compressed workweek scheduling with differing weekdays and weekends labour demands, *Asia-Pacific Journal of Operational Re*search **20**, 1-20.

Bard, J.F., Binici, C., deSilva, A.H. (2003), Staff scheduling at the United States Postal Service, *Computers and Operations Research* **30**, 745-771.

Billionnet, A. (1999), Integer programming to schedule a hierarchical workforce with variable demands, *European Journal of Operational Research* **114**, 105-114.

Burns, R.N. and Narasimhan, R. (1999), Multiple shift scheduling of workforce on four-day workweeks, *Journal of the Operational Research Society* **50**, 979-981.

Burns, R.N., Narasimhan, R. and Smith, L.D. (1998), A set processing algorithm for scheduling staff on 4-day or 3-day work weeks, *Naval Research Logistics* **45**, 839-853.

Cezik, T., Gunluk, O. and Luss, H. (2001), An integer programming model for the weekly tour scheduling problem, *Naval Research Logistics* **48**, 607-624.

Ernst, A.T., Jiang, H., Krishnamoorthy, M., Owens, B and Sier, D. (2004). An annotated bibliography of personnel scheduling and rostering, *Annals of Operations Research* **127**, 3-27.

Hung, R. (1991), Single-shift scheduling under a compressed workweek, *Omega* **19**, 494-497.

Hung, R. (1993), A three-day workweek multiple-shift scheduling model, *Journal of the Operational Research Society* **44**, 141-146.

Hung, R. (1994), Single-shift off-day scheduling of a hierarchical workforce with variable demands, *European Journal of Operational Research*, **78**, 49-57.

Lankford, W.M. (1998), Changing schedules: a case for alternative work schedules, *Career Development International* **3**, 161-163.

Lin, C.K.Y., Lai, K.F. and Hung, S.L. (2000), Development of a workforce management system for a customer hotline service, *Computers & Operations Research* **27**, 987-1004.

Narasimhan, R. (1997), Algorithm for a single shift scheduling of hierarchical workforce, *European Journal of Operational Research* **96**, 113-121.

Narasimhan R. (2000), An algorithm for multiple shift scheduling of hierarchical workforce on four-day or three-day workweeks, *INFOR* **38**, 14-32.

McCampbell, S.A. (1996), Benefits achieved through alternative work schedules, *Human Resources Planning* **19**, 30-37.