Journal of Mathematical Modelling and Algorithms **2:** 67–80, 2003.
© 2003 *Kluwer Academic Publishers. Printed in the Netherlands.*

Four-Day Workweek Scheduling with Two or Three Consecutive Days Off

HESHAM K. ALFARES
System Engineering Department, King Fahd University of Petroleum & Minerals, PO Box 5067, Dhahran 31261, Saudi Arabia. e-mail: hesham@ccse.kfupm.edu.sa

(Received: 5 August 2002; accepted in final form: 26 February 2003)

Abstract. A four-day workweek days-off scheduling problem is considered. Out of the three days off per week for each employee, either two or three days must be consecutive. An optimization algorithm is presented which starts by utilizing the problem's special structure to determine the minimum workforce size. Subsequently, workers are assigned to different days-off work patterns in order to minimize either the total number or the total cost of the workforce. Different procedures must be followed in assigning days-off patterns, depending on the characteristics of labor demands. In some cases, optimum days-off assignments are determined without requiring mathematical programming. In other cases, a workforce size constraint is added to the integer programming model, greatly improving computational performance.

Mathematics Subject Classifications (2000): 90B35, 90C10.

Key words: workforce scheduling, optimization, integer programming, staffing.

1. Introduction

Workforce scheduling is important to both employees and their organizations. Work schedules directly affect the employees' pay, quality of life, and structure of work, family, and leisure activities. On the other hand, employee work schedules affect the labor cost of organizations and their ability to meet time varying demands for goods and services. Labor cost constitutes a significant portion of the total expenditures in most organizations. Effective scheduling of employees can reduce both the size and the cost of the workforce. Moreover, employee scheduling is a difficult problem to solve due to several factors that include: (1) humans' need for rest, i.e., daily and weekly breaks, (2) impossibility of storing human service, (3) strict labor laws and union agreements, and (4) seniority and preferences.

Days-off scheduling is a significant problem for organizations that operate on a seven-days-per-week basis, such as police stations, restaurants, and airlines. Usually, labor demands are assumed to vary from day to day during the week, but the weekly demand pattern is assumed to be constant from week to week. The solution of the days-off scheduling problem specifies the number of employees and the work and off days of each employee. The objective is to satisfy the daily labor demands with the minimum size or minimum cost of the workforce.

The traditional Monday-to-Friday workweek is not applicable to organizations that operate seven days a week. In order to meet weekend labor demand, some employees must be assigned to work on weekends and hence must be given non-weekend off days. Usually, the $(5, 7)$ schedule is used, in which employees are given two consecutive days off per week. According to McCampbell [10], however, alternative work schedules (AWS) have grown steadily in the U.S. since 1972. The U.S. Office of Personnel Management suggests seven types of flexible schedule models and 3 types of compressed-schedule models. AWS, which include compressed week schedules, provide higher management flexibility and greater employee satisfaction.

The 4-day compressed workweek is considered in this paper, but additional flexibility is assumed with respect to the three days off per week. Out of the three days off, either two or three off days must be consecutive. An optimum algorithm that utilizes LP primal-dual relations is developed for minimizing either the total number or the total cost of employees. In some cases, a workforce size constraint is added to integer programming (IP) model of the problem to efficiently obtain optimum solutions. In other cases, the optimum integer solution is obtained without mathematical programming at all. The proposed algorithm provides computational performance that is significantly superior to the traditional integer programming solution.

This paper is organized as follows. First, relevant literature is surveyed and discussed. Subsequently, integer programming models are presented. Next, a description is given of the procedure for determining the minimum number of employees, followed by a development of methods for assigning employees to days-off patterns. Afterwards, the new algorithm is computationally compared to conventional integer programming. Finally, a numerical example is solved, and then conclusions are given.

2. Literature Survey

Nanda and Browne [11] classify employee scheduling into 10 categories that address the scheduling of: (1) days on and days off, (2) shifts and work rosters, (3) work tours, (4) integrated workweeks, (5) meal breaks and rest periods, (6) part-time employees, (7) vacations and training assignments, (8) overtime, (9) compensation, i.e. premiums/wages, and (10) alternate work patterns. Nanda and Browne [11] present a comprehensive survey of literature on all these types of employee scheduling problems. In this paper, we are primarily concerned with days-off scheduling and alternate work patterns, especially shorter workweeks.

Several algorithms have been developed for the days-off scheduling problem, assuming different combinations of the following characteristics: number of employee categories, pattern of labor demands, limit on work-stretch length, constraints on weekend work frequency, number of work days per week, and whether off days are consecutive. Narasimhan [13] compares several days-off scheduling approaches. Although nonconsecutive days off provide greater scheduling flexibility and efficiency, Nanda and Browne [11] note that they are less desirable. However, nonconsecutive days off can be made more acceptable if combined with some desirable features such as compressed workweeks.

Nanda and Browne [11] list several types of alternative work schedules, including flexitime and compressed work weeks. According to McCampbell [10], the U.S. Office of Personnel Management suggests three types of compressed work schedules: the 3-day workweek, the 4-day workweek, and the 5-4/9 plan. In the 4-day compressed workweek, each employee works 4 days and takes 3 days off per week. McCampbell [10] cites several advantages of alternative work schedules, and in particular compressed workweeks, to both employees and organizations. Employees gain greater freedom, more time for personal and family matters, and improved morale. On the other hand, employers enhance productivity, facilitate recruitment, and reduce absenteeism, overtime, and turnover.

The 4-day workweek is a popular real-life compressed workweek schedule. Under this schedule, also known as the (4, 7) or the 4×10 schedule, each employee works 4 days per week, 10 hours per workday. Lankford [9] describes a pilot application of a 4-day workweek schedule at the Analytical Central Call Management at Hewlett Packard. Gould [6] describes another 4-day schedule (the rolling four) in which two groups of employees alternate 4 consecutive workdays and 4 consecutive off days over an 8-day cycle.

Hung [7] presents two models for scheduling a homogeneous workforce under two assumptions: (i) D workers are required on weekdays, and E workers on weekends, and (ii) each worker must receive at least A out of B weekends off. Model 1 assumes each employee works 4 days and rests 3 days each week, while Model 2 assumes each employee works 4 days and rests 3 days in one week, and works 3 days and rests 4 days in the other week. Hung [8] develops a heuristic procedure for scheduling a hierarchical workforce, where the workweek length can vary between 3, 4, or 5 days. Billionnet [3] models and solves the same problem by integer programming.

Burns and Narasimhan [4] extend the work of Hung [7] by restricting the length of maximum work stretches and the transition time (number of days off) required when changing shifts. Narasimhan [12] considers a similar days-off scheduling problem for a hierarchical workforce, in which each employee cannot be assigned more than 5 consecutive workdays. Narasimhan [13] also presents another solution technique for multiple-shift 3-day or 4-day workweek scheduling of a hierarchical workforce. Burns *et al.* [5] develop an algorithm for 3-day and 4-day workweeks, assuming variable labor demand and limits on work stretch lengths. Alfares [2] presents an algorithm for 4-day workweek scheduling assuming variable labor demands and requiring the 3 off days per week to be consecutive.

This paper presents an algorithm for single-shift 4-day workweek scheduling of a homogeneous workforce under the following assumptions: (i) the demand for employees may vary from day to day for the given week, and (ii) at least 2 of the 3 weekly off-days are consecutive. The algorithm can be used to obtain an optimal solution for the problem, in some cases with enhanced integer programming and in others without mathematical programming at all.

3. Integer Programming Model

The 4-day workweek scheduling problem with 2 or 3 consecutive days off can be formulated as an integer linear programming (ILP) model, as follows:

$$\text{Minimize } W = \sum_{j=1}^{28} x_j \tag{1}$$

subject to

$$\sum_{j=1}^{28} a_{ij} x_j \geqslant r_j, \quad i = 1, 2, \ldots, 7, \tag{2}$$

$$x_j \geqslant 0 \text{ and integer}, \quad j = 1, 2, \ldots, 28, \tag{3}$$

where $W =$ total number of workers assigned to days-off patterns,
$x_j =$ number of workers assigned to weekly days-off work pattern j,
$a_{ij} = 1$ if day i is a work day for pattern j, 0 otherwise (see Table I),
$r_i =$ minimum number of workers required on day i.

The objective (1) is to minimize the workforce size, i.e., the total number of workers. Constraints (2) ensure that the number of workers assigned is at least equal to the number required for each day of the week. Since $\sum_{j=1}^{28} x_j$ is equal to W, (2) can be expressed in the following sparser matrix representation, which is easier to deal with in making days-off assignments:

$$\sum_{j=1}^{28} a_{ij}^{\mathrm{C}} x_j \leqslant b_i, \quad i = 1, 2, \ldots, 7, \tag{4}$$

where

$$a_{ij}^{\mathrm{C}} = 1 - a_{ij}, \tag{5}$$

$$b_i = W - r_i, \tag{6}$$

$$= \text{maximum number of workers off on day } i.$$

In matrix notation, constraint system (2) can be written as

$$AX \geqslant R. \tag{7}$$

4. Minimum Workforce Size

Given seven daily labor demands, $r_1, r_2, \ldots, r_7$, the minimum workforce size, W, can be determined from the solution of above model without integer programming, simply by utilizing the problem structure. Depending on the given labor demands, there are three dominant solutions.

(1) The workforce size must be greater than the labor demand for any day, thus

$$W \geqslant r_{\max} = \max\{r_1, r_2, \ldots, r_7\}. \tag{8}$$

(2) Since each employee is assigned 4 workdays per week, the total man-days assigned is $4\sum_{j=1}^{28} x_j = 4W$, which has to be greater than the total man-days required, which is $\sum_{i=1}^{7} r_i$. Therefore, $4W \geqslant \sum_{i=1}^{7} r_i$, or

$$W \geqslant \sum_{i=1}^{7} r_i/4. \tag{9}$$

(3) Let us consider the seven sets of 3 nonconsecutive days off denoted by $t_1, \ldots, t_7$ and corresponding sets of 4 workdays denoted by $s_1, \ldots, s_7$, shown in Table II. For each set of 4 workdays, s_i, the sum of labor demands for days s_i is denoted by S_i. Taking $S_1 = (r_1 + r_2 + r_4 + r_6)$ as an example and summing rows 1, 2, 4, and 6 in constraints (2) referring to matrix A in Table I, we obtain: $3\sum_{j=1}^{28} x_j \geqslant r_1 + r_2 + r_4 + r_6$, or $3W \geqslant S_1$. Thus, in order to satisfy labor demands on days 1, 2, 4, and 6, $W \geqslant S_1/3$. A similar result will be obtained for sets $S_2, \ldots, S_7$, thus leading to the following bound:

$$W \geqslant S_{\max}/3. \tag{10}$$

Table I. Days-off matrix $A = \{a_{ij}\}$ and cost vector $C = \{c_j\}$ for the 28 days-off work patterns

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
i																												
1	0	1	1	1	1	0	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	1	1	1	0	1	0	0
2	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	1	1	1	0	1	0
3	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	1	1	1	0	1
4	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	1	1	1	0
5	1	1	0	0	0	1	1	0	1	0	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	1	1	1
6	1	1	1	0	0	0	1	1	1	1	0	0	1	0	1	0	1	0	0	1	1	0	1	1	0	0	1	1
7	1	1	1	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0	0	1	1	0	0	1	0	0	1
c_j	$4+2\beta$	$4+2\beta$	$4+2\beta$	$4+\beta$	4	4	$4+\beta$	$4+2\beta$	$4+2\beta$	$4+2\beta$	$4+\beta$	4	$4+\beta$	$4+\beta$	$4+\beta$	$4+\beta$	$4+2\beta$	$4+\beta$	4	$4+\beta$	$4+2\beta$	$4+\beta$	$4+\beta$	$4+\beta$	$4+\beta$	4	$4+\beta$	$4+2\beta$

Table II. Sets of 3 nonconsecutive off days t_i and corresponding 4 workdays s_i

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

The bound expressed by (10) can be considered as the cost of requiring at least two off days per week to be consecutive. If nonconsecutive days off were allowed, we could satisfy the demand on days 1, 2, 4, and 6 by assigning employees to days-off pattern t_1, i.e., off on days 3, 5, and 7. In that case, the number of employees would be equal to the maximum demand on these 4 days, i.e., $W \geqslant \max\{r_1, r_2, r_4, r_6\}$, which is dominated by the first bound ($W \geqslant r_{\max}$).

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

$$W = \max\left\{r_{\max}, \left\lceil \frac{1}{4}\sum_{i=1}^{7} r_i \right\rceil, \left\lceil \frac{S_{\max}}{3} \right\rceil \right\}, \tag{11}$$

where $S_{\max} = \max\{S_1, S_2, \ldots, S_7\}$,
$\lceil a \rceil =$ smallest integer $\geqslant a$

and

$$S_i = \sum_{j \in s_i} r_j, \quad i = 1, 2, \ldots, 7, \tag{12}$$

$$s_i = \{i, i+1, i+3, i+5\}, \quad i = 1, 2, \ldots, 7, \tag{13}$$

where s_i is circular set with a cycle $= 7$ (see Table II).

5. Days-Off Assignments

After applying (11) to determine the workforce size, W, we need to assign each of the W workers to a specific days-off pattern in order to minimize total cost. We assume that the cost of each days-off pattern depends on the number of premium-paid weekend workdays. Assuming that each worker is paid 1 unit per regular workday and $1 + \beta$ units ($\beta \geqslant 0$) per weekend workday, the weekly costs of the 28 days-off patterns $C = \{c_1, c_2, \ldots, c_{28}\}$ are shown in Table I. Changing the objective to the minimization of total cost, (1) is replaced by

$$\text{Minimize } Z = \sum_{j=1}^{28} c_j x_j, \tag{14}$$

where $c_j =$ weekly cost of days-off pattern j per worker, shown in Table I.

Introducing these costs in the IP model, the optimum solution with (14) naturally changes from that with (1). However, although the solution may slightly differ, the workforce size W does not change in all cases of Equation (11). This means that for the cost structure defined by C, for all $\beta \geqslant 0$, the minimum *cost* is always obtained with the minimum *number* of workers.

To illustrate the above point, we will use the dual solution under differential costs ($\beta \geqslant 0$) and complementary slackness primal-dual relationships. Let us consider for example the case when the maximum argument of (11) is $\lceil S_1/3 \rceil$. In the optimum dual solution, the basic variables are: y_1, y_2, y_4, y_6, and y_7, and the equations are dual constraints: 3, 5, 8, 11, 13, 15, 19, 23, 24, and 27. Since basic dual variables correspond to primal equations and dual equation corresponds to primal basic variables, the corresponding primal constraints (4) can be written as

$$\begin{array}{l} x_{13} + x_{27} = b_1 \\ x_8 + x_{15} = b_2 \\ x_3 + x_8 + x_{11} + x_{15} + x_{19} + x_{23} + x_{27} \leqslant b_3 \\ x_3 + x_{23} + x_{24} = b_4 \\ x_3 + x_5 + x_8 + x_{11} + x_{13} + x_{24} + x_{27} \leqslant b_5 \\ x_5 + x_{11} + x_{19} = b_6 \\ x_5 + x_{13} + x_{15} + x_{19} + x_{23} + x_{24} + x_{27} = b_7 \end{array} \tag{15}$$

The workforce size is obtained by summing the first, second, fourth, and sixth constraints in (15) as follows:

$$\begin{aligned} W &= (x_{13} + x_{27}) + (x_8 + x_{15}) + (x_3 + x_{23} + x_{24}) + (x_5 + x_{11} + x_{19}) \\ &= b_1 + b_2 + b_4 + b_6 \\ &= (W - r_1) + (W - r_2) + (W - r_4) + (W - r_6) \\ &= 4W - S_1. \end{aligned}$$

Therefore

$$W = \lceil S_1/3 \rceil.$$

This example shows that introducing the varying costs for different days-off patterns defined by C does not affect the workforce size. Similar results are obtained for all other cases. The assignment of workers to days-off work patterns will depend on which argument of (11) is maximum. There are three possible cases, which are discussed next.

Case 1. $r_{\max}$ *or* $\lceil \sum r_i/4 \rceil$ *is maximum*
In this case, the value of the minimum workforce size W is calculated first using (11), then the following constraint is added to the primal integer programming model defined by (14), (2), and (3):

$$\sum_{j=1}^{28} x_j = W. \tag{16}$$

The addition of this constraint has been found to drastically reduce integer programming computation times. In order to evaluate the effect of adding constraint (16), computational experiments have been carried out using 202 test problems. In all these problems, the value of β was set at 0.5 to indicate 50% higher pay for weekend work. The 202 problems, described by Alfares [1], are divided into 10 sets with different demand types, but all have an average demand of 50 workers per day. The first six sets: level, trend, concave, convex, unimodal, and sinusoidal, involve 17 problems each, with a demand range of 34 to 66. The last four sets involve 25 problems each, randomly distributed on the intervals: [34, 66], [0, 100], [20, 80], and [45, 55].

Microsoft Excel Solver®, run on an 866-MHz Pentium III PC with a 128 MB RAM, was used to solve the test problems by integer programming. The results of computational experiments are summarized in Table III. For all 10 sets of problems, adding constraint (16) has decreased the average, maximum, and variation of solution times. Overall, average solution time dropped by 98%, from 7.67 seconds to 0.16 seconds. Maximum solution time fell by 99.9%, from 310.17 seconds to only 0.28 seconds. Similarly, the standard deviation decreased from 32.84 seconds to only 0.02 seconds.

It is interesting to note that similar relative savings in computation times have been obtained by running Microsoft Excel Solver® on a 120-MHz Pentium; average solution also time dropped by 98%, from 56.52 seconds to 1.14 seconds. On both computers, the addition of constraint (12) has made the solution on average 50 times faster. Clearly, adding constraint (16) leads to a remarkable reduction in both the mean and variation of solution times.

Case 2. $\lceil S_{\max}/3 \rceil$ *is maximum*
If $\lceil S_{\max}/3 \rceil$ is maximum, the dual variables y_j, $j \in s_i \cup \{6, 7\}$ will be basic. In addition, 10 dual constraints will be equations. The corresponding primal constraints will have 10 basic variables and four or five equations. Optimum variable values satisfying these primal constraints can be found without integer programming. To minimize total cost, as many workers as possible are assigned to days-off patterns

Table III. Integer programming solution times in seconds (minimum, average, maximum, and standard deviations) with and without constraint (16)

Problem set	No. of problems	Without constraint min	ave	max	std dev	With constraint min	ave	max	std dev
1	17	0.16	17.53	250.13	60.33	0.11	0.16	0.28	0.04
2	17	0.16	2.54	17.41	4.69	0.16	0.17	0.17	0.01
3	17	0.16	3.59	29.33	7.53	0.11	0.16	0.21	0.02
4	17	0.16	8.57	110.01	26.67	0.16	0.16	0.17	0.01
5	17	0.16	21.55	310.17	74.81	0.16	0.17	0.22	0.01
6	17	0.16	9.71	154.12	37.22	0.11	0.16	0.17	0.02
7	25	0.16	0.58	5.00	1.07	0.11	0.17	0.22	0.02
8	25	0.16	4.20	44.55	12.20	0.11	0.17	0.22	0.02
9	25	0.16	5.86	89.86	18.10	0.11	0.17	0.22	0.02
10	25	0.16	8.14	124.46	25.33	0.11	0.16	0.22	0.02
Overall (866 MHz)	202	0.16	7.67	310.17	32.84	0.11	0.16	0.28	0.02
Overall (120 MHz)	202	0.93	56.52	1500*	223.1	0.98	1.14	1.87	0.139

*Some problems could not be solved within the 1500-second time limit.

with the lowest cost. For example, the primal constraints (15) corresponding to the case when $\lceil S_1/3 \rceil$ is maximum can be optimally satisfied by the following values:

$$\begin{aligned}
x_5 &= \min(b_5, b_6, b_7 - b_1),\\
x_{19} &= \min(b_3 - b_2, b_6 - x_5, b_7 - b_1 - x_5),\\
x_{11} &= b_6 - x_5 - x_{19},\\
x_{13} &= \min(b_1, b_5 - x_5 - x_{11}),\\
x_{27} &= b_1 - x_{13},\\
x_{15} &= \min(b_2, b_7 - b_1 - x_5 - x_{19}),\\
x_8 &= b_2 - x_{15},\\
x_3 &= b_1 + b_4 - b_7 + x_5 + x_{15} + x_{19},\\
x_{23} &= \min(b_3 - b_2 - x_3 - x_{11} - x_{19} - x_{27}, b_4 - x_3),\\
x_{24} &= b_4 - x_3 - x_{23}.
\end{aligned} \tag{17}$$

Similar optimum solutions have been derived for cases when $\lceil S_i/3 \rceil$ is maximum, $i = 2, \ldots, 7$, as shown in Table IV. Moreover, expressions for the values of days-off work assignments $\{x_1, x_2, \ldots, x_{28}]$ can also be derived without mathematical programming for the following special case.

Table IV. Values of nonzero days-off assignments, $x_1, \ldots, x_{28}$, for the case when $S_i/3$ is the maximum argument of (11)

i	10 nonzero days-off assignments corresponding to $S_i/3$ being maximum
1	$x_5 = \min(b_5, b_6, b_7 - b_1)$, $x_{i9} = \min(b_3 - b_2, b_6 - x_5, b_7 - b_1 - x_5)$, $x_{11} = b_6 - x_5 - x_{19}$, $x_{13} = \min(b_1, b_5 - x_5 - x_{11})$, $x_{27} = b_1 - x_{13}$, $x_{15} = \min(b_2, b_7 - b_1 - x_5 - x_{19})$, $x_8 = b_2 - x_{15}$, $x_3 = b_1 + b_4 - b_7 + x_5 + x_{15} + x_{19}$, $x_{23} = \min(b_3 - b_2 - x_3 - x_{11} - x_{19} - x_{27}, b_4 - x_3)$, $x_{24} = b_4 - x_3 - x_{23}$
2	$x_6 = \min(b_1 - b_2, b_6, b_7)$, $x_{12} = \min(b_4 - b_3, b_6 - x_6, b_7 - x_6)$, $x_{20} = b_7 - x_6 - x_{12}$, $x_{16} = \min(b_3, b_6 - x_6 - x_{12})$, $x_9 = b_3 - x_{16}$, $x_{14} = \min(b_2, b_6 - x_6 - x_{12} - x_{16})$, $x_{28} = b_2 - x_{14}$, $x_{17} = b_5 - b_6 + x_6 + x_{12} + x_{14} + x_{16}$, $x_4 = \min(b_4 - b_3 - x_{12} - x_{17} - x_{20} - x_{28}, b_5 - x_{17})$, $x_{25} = b_5 - x_4 - x_{17}$
3	$x_5 = \min(b_5 - b_4, b_6, b_7)$, $x_{26} = \min(b_2 - b_3, b_6 - x_5, b_7 - x_5)$, $x_{18} = b_6 - x_5 - x_{26}$, $x_{15} = \min(b_3, b_7 - x_5 - x_{18} - x_{26})$, $x_8 = b_3 - x_{15}$, $x_{24} = \min(b_4, b_7 - x_5 - x_{15} - x_{26})$, $x_{10} = b_4 - x_{24}$, $x_{21} = b_1 - b_7 + x_5 + x_{15} + x_{24} + x_{26}$, $x_7 = \min(b_1 - x_{21}, b_2 - b_3 - x_{10} - x_{18} - x_{21} - x_{26})$, $x_{13} = b_1 - x_7 - x_{21}$
4	$x_6 = \min(b - 1, b_6 - b_5, b_7)$, $x_{19} = \min(b_3 - b_4, b_6 - b_5 - x_6, b_7 - x_6)$, $x_{27} = b_7 - x_6 - x_{19}$, $x_{16} = \min(b_4, b_6 - b_5 - x_6 - x_{19})$, $x_9 = b_4 - x_{16}$, $x_{11} = \min(b_3 - b_4 - x_{19} - x_{27}, b_5)$, $x_{25} = b_5 - x_{11}$, $x_1 = b_2 - b_6 + b_5 + x_6 + x_{16} + x_{19}$, $x_{14} = \min(b_1 - x_1 - x_6 - x_9 - x_{25} - x_{27}, b_2 - x_1)$, $x_{22} = b_2 - x_1 - x_{14}$
5	$x_{12} = \min(b_4 - b_5, b_6)$, $x_{26} = b_6 - x_{12}$, $x_{24} = \min(b_5, b_7 - b_6)$, $x_{10} = b_5 - x_{24}$, $x_{15} = \min(b_2 - x_{10} - x_{26}, b_3, b_7 - b_6 - x_{24})$, $x_{23} = \min(b_3 - x_{15}, b_4 - b_5 - x_{12}, b_7 - b_6 - x_{15} - x_{24})$, $x_2 = b_3 - x_{15} - x_{23}$, $x_{28} = b_1 - b_7 + b_6 + x_{15} + x_{23} + x_{24}$, $x_7 = \min(b_1 - x_{28}, b_2 - x_2 - x_{10} - x_{15} - x_{26} - x_{28})$, $x_{20} = b_1 - x_7 - x_{28}$
6	$x_{11} = \min(b_3, b_6)$, $x_{25} = b_6 - x_{11}$, $x_{13} = \min(b_5 - b_6, b_7)$, $x_{27} = b_7 - x_{13}$, $x_1 = \min(b_1 - b_7 - x_{25}, b_2, b_3 - x_{11} - x_{27})$, $x_8 = \min(b_2 - x_1, b_3 - x_1 - x_{11} - x_{27}, b_5 - b_6 - x_{13} - x_{25})$, $x_{21} = b_2 - x_1 - x_8$, $x_9 = \min(b_1 - b_7 - x_1 - x_{21} - x_{25}, b_3 - x_1 - x_8 - x_{11} - x_{27}, b_4)$, $x_{17} = \min(b_1 - b_7 - x_1 - x_9 - x_{21} - x_{25}, b_4 - x_9)$, $x_3 = b_4 - x_9 - x_{17}$
7	$x_{12} = \min(b_4, b_7)$, $x_{26} = b_7 - x_{12}$, $x_{14} = \min(b_1, b_6 - b_7)$, $x_{28} = b_1 - x_{14}$, $x_4 = \min(b_4 - x_{12} - x_{28}, b_5, b_6 - b_7 - x_{14})$, $x_{18} = \min(b_2 - b_1 - x_{26}, b_5 - x_4, b_6 - b_7 - x_4 - x_{14})$, $x_{10} = b_5 - x_4 - x_{18}$, $x_2 = b_3 - b_6 + b_7 + x_4 + x_{14} + x_{18}$, $x_{16} = \min(b_3 - x_2, b_4 - x_2 - x_4 - x_{10} - x_{12} - x_{28})$, $x_{22} = b_3 - x_2 - x_{16}$

Case 3. Weekday and Weekend Demands D & E ($E \geqslant 2.5D$)
If weekday labor demands are uniform and denoted by D, and weekend labor demands are uniform and denoted by E, Equation (11) becomes

$$W = \max\{D, E, \lceil (5D + 2E)/4 \rceil, \lceil (3D + E)/3 \rceil, \lceil 2(D + E)/3 \rceil\}. \tag{18}$$

Defining

$$\alpha = E/D \tag{19}$$

then

$$W = \max\{1, \alpha, \lceil 1.25 + 0.5\alpha \rceil, \lceil 1 + \alpha/3 \rceil, \lceil 2/3 + 2\alpha/3 \rceil\} * D. \tag{20}$$

Since all arguments of (20) are linear functions of a, the optimum solution can be easily found as

$$W = \lceil 1.25 + 0.5\alpha \rceil D, \quad \text{for } 0 \leqslant \alpha \leqslant 2.5, \quad \text{corresponding to } W = \left\lceil \sum_{i=1}^{7} r_i/4 \right\rceil,$$

and

$$W = \alpha D = E, \quad \text{for } \alpha \geqslant 2.5, \quad \text{corresponding to } W = r_{\max} = E.$$

If $\alpha \leqslant 2.5$, the solution is obtained by integer programming, after appending constraint (16) which takes the form: $\sum x = \lceil (5D + 2E)/4 \rceil$. If $\alpha \geqslant 2.5$, the solution can be obtained without integer programming. Since $W = E = r_6 = r_7$, both $b_6 = b_7 = 0$, and primal constraints (4) are reduced to

$$\begin{aligned}
x_1 + x_9 + x_{17} + x_{21} + x_{28} &\leqslant E - D \\
x_1 + x_2 + x_8 + x_{10} + x_{21} + x_{28} &\leqslant E - D \\
x_1 + x_2 + x_3 + x_8 + x_9 &\leqslant E - D \\
x_2 + x_3 + x_9 + x_{10} + x_{17} + x_{28} &\leqslant E - D \\
x_3 + x_8 + x_{10} + x_{17} + x_{21} &\leqslant E - D
\end{aligned} \tag{21}$$

With $E \geqslant 2.5D$, the solution satisfying the above constraints is given by

$$x_1 = D, \qquad x_3 = \lceil D/2 \rceil, \qquad x_{10} = E - 2D, \qquad x_{17} = \lfloor D/2 \rfloor, \tag{22}$$

where

$$\lfloor a \rfloor = \text{largest integer} \leqslant a.$$

6. Steps of the Algorithm

The preceding development is summarized by the following algorithm description:

1. Determine the minimum workforce size W using Equation (11).
2. (a) If $\max\{r_{\max}, \sum r/4, S_{\max}/3\} = r_{\max}$ or $\sum r/4$, then
   - let $W = r_{\max}$ or $W = \lceil \sum r/4 \rceil$, respectively, then add constraint (16) to the integer programming model defined by (14), (2), and (3).
   - use integer programming to find $x_1, x_2, \ldots, x_{28}$.

   (b) If $\max\{r_{\max}, \sum r/4, S_{\max}/3\} = S_{\max}/3$, then
   - if $S_{\max}/3$ is not integer, increment $S_i = S_{\max}$ by 1 or 2 to make it a multiple of 3. Among the four daily labor demands that can be increased r_j, $j \in s_i$, increase the minimum non-weekend demand(s), $r_1, \ldots, r_5$.
   - let $W = \lceil S_{\max}/3 \rceil$, then determine $b_1, \ldots, b_7$ using Equation (6).
   - apply row i in Table IV to find $x_1, x_2, \ldots, x_{28}$.

   (c) If $D = r_1 = r_2 = \ldots = r_5$, $E = r_6 = r_7$, and $E \geqslant 2.5D$, then
   - apply system (22) to find the solution
3. In the case of ties, apply any system arbitrarily.

7. A Solved Example

Assume that the daily labor demands for a certain workweek are given as

$$r_1, r_2, \ldots, r_7 = 12, 14, 5, 10, 4, 11, 3.$$

Using (12), the following values are calculated:

$$S_1, S_2, \ldots, S_7 = 47, 26, 38, 31, 32, 38, 24.$$

Hence

$$r_{\max} = 14,$$

$$\sum_{i=1}^{7} r_i/4 = 59/4 = 14.75$$

and

$$S_{\max}/3 = S_1/3 = 47/3 = 15.67.$$

Using (11), the workforce size is

$$W = \lceil S_1/3 \rceil = \lceil 15.67 \rceil = 16.$$

Since $S_1/3 = 15.67$ is not integer, we must increment S_1 by 1 in order to make it divisible by 3. The set S_1 contains demands for days 1, 2, 4, and 6. Using the criteria specified in the algorithm, we add 1 to the demand for day 4 (thus $r_4 = 11$), and then use Equation (6): $b_i = 16 - r_i$, to obtain

$$b_1, b_2, \ldots, b_7 = 4, 2, 11, 5, 12, 5, 13.$$

Using row 1 in Table IV, corresponding to system (17), we obtain the following days-off assignments:

$$\begin{aligned}
x_5 &= \min(12, 5, 13 - 4) = 5,\\
x_{19} &= \min(11 - 2, 5 - 5, 13 - 4 - 5) = 0,\\
x_{11} &= 5 - 5 - 0 = 0,\\
x_{13} &= \min(4, 12 - 5 - 0) = 4,\\
x_{27} &= 4 - 4 = 0,\\
x_{15} &= \min(2, 13 - 4 - 5 - 0) = 2,\\
x_8 &= 2 - 2 = 0,\\
x_3 &= 4 + 5 - 13 + 5 + 2 + 0 = 3,\\
x_{23} &= \min(11 - 2 - 3 - 0 - 0 - 0, 5 - 3) = 2,\\
x_{24} &= 5 - 3 - 2 = 0.
\end{aligned}$$

8. Conclusions

An optimum days-off scheduling algorithm has been developed for the 4-day workweek problem in which workers are given two or three consecutive off days per week. A simple formula has been derived to determine the minimum workforce size as a function of the given labor demands. Since the costs of different days-off patterns are not assumed to be equal, the algorithm can be used to minimize either the total number or the total cost of workers assigned.

For two cases, formulas have been developed to produce the optimum solution without resorting to mathematical programming. For the remaining case, the addition of a workforce-size constraint to the integer programming model has been shown to greatly improve the computational efficiency. Extensive computational experiments have shown that the enhanced model is remarkably more efficient than the conventional integer programming model.

Acknowledgement

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

References

1. Alfares, H. K.: An efficient two-phase algorithm for cyclic days-off scheduling, *Comput. Oper. Res.* **25** (1998), 913–923.
2. Alfares, H. K.: Dual-based optimization of cyclic four-day workweek scheduling, *IMA J. Math. Appl. Bus. Indust.* **11** (2000), 269–283.

3. Billionnet, A.: Integer programming to schedule a hierarchical workforce with variable demands, *Eur. J. Oper. Res.* **114** (1999), 105–114.
4. Burns, R. and Narasimhan, R.: Multiple shift scheduling of workforce on four-day workweeks, *J. Oper. Res. Soc.* **50** (1999), 979–981.
5. Burns, R. N., Narasimhan, R. and Smith, L. D.: A set processing algorithm for scheduling staff on 4-day or 3-day work weeks, *Naval Res. Logist.* **45** (1998), 839–853.
6. Gould, C. H.: Rolling fours: Novel work schedule, *J. Construct. Engrg. Manag.* **114** (1988), 577–593.
7. Hung, R.: Single-shift scheduling under a compressed workweek, *OMEGA* **19** (1991), 494–497.
8. Hung, R.: Single-shift off-day scheduling of a hierarchical workforce with variable demands, *Eur. J. Oper. Res.* **78** (1994), 49–57.
9. Lankford, W. M.: Changing schedules: a case for alternative work schedules, *Career Devel. Internat.* **3** (1998), 161–163.
10. McCampbell, A. S.: Benefits achieved through alternative work schedules, *Human Res. Plann.* **19** (1996), 30–37.
11. Nanda, R. and Browne, J.: *Introduction to Employee Scheduling*, Van Nostrand Reinhold, New York, 1992.
12. Narasimhan, R.: Algorithm for a single shift scheduling of hierarchical workforce, *Eur. J. Oper. Res.* **96** (1996), 113–121.
13. Narasimhan, R.: Algorithm for multiple shift scheduling of hierarchical workforce on four-day or three-day workweeks, *INFOR* **38**(1) (2000), 14–32.