



Staff scheduling at the United States Postal Service

Jonathan F. Bard^{a,*}, Canan Binici^a, Anura H. deSilva^b

^a*Graduate Program in Operations Research and Industrial Engineering, University of Texas,
Austin, TX 78712-1063, USA*

^b*Planmatics, Inc., 15200 Shady Grove Road, Rockville, MD 20850, USA*

Received 1 October 2001; accepted 1 February 2002

Abstract

The purpose of this paper is to present a full-scale model of the tour scheduling problem as it arises in the United States Postal Service, and to examine several scenarios aimed at reducing the size of the workforce. The problem is formulated as a pure integer linear program and solved with CPLEX. The baseline model includes both full-time and part-time workers, as well as the principal constraints defined by the union contract. The scenarios include requirements for two days off in a row, variable daily start times, the use of part-time flexible workers, and a parametric analysis of full-time to part-time restrictions. The results indicate that problem instances of realistic size can be solved within 1 h, and that measurable savings can be achieved by departing from current practice.

Scope and purpose

Effective personnel scheduling has become one of the primary means by which service organizations remain competitive. Unlike manufacturing, where standard shifts and days off are the rule, the service industry often operates 24 h a day, 7 days a week and faces widely fluctuating demand. Poor personnel schedules can lead to an oversupply of workers with too much idle time, or an undersupply with an attendant loss of business. In this paper, we present an integrated model that can be used to find optimal schedules for a homogeneous workforce. The objective is to meet daily staffing requirements at minimum cost without violating labor agreements and government regulations. The approach requires the solution of a large-scale integer linear program to determine general staffing needs for both full-time and part-time employees. Weekly tours are then constructed, and each employee is assigned a lunch break. Extensions to the model include different days off policies, variable start times, and the use of part-time flexible workers. Test data were provided by the United States Postal Service for one of their processing and distribution centers. The main conclusions of the study are that problems of realistic size can be solved quickly, and that substantial savings can be achieved by considering less restrictive policies. While the latter is not surprising, many organizations adopt costly practices, such as providing weekly schedules with the same start times every day for each employee, without fully realizing their implications. © 2002 Elsevier Science Ltd. All rights reserved.

* Corresponding author. Tel.: +1-512-471-3076; fax: +1-512-232-1489.

E-mail addresses: jbard@mail.utexas.edu (J.F. Bard), adesilva@planitusa.com (A.H. deSilva).

Keywords: Staff scheduling; Shifts; Tours; Integer programming; Service industry

1. Introduction

Service organizations typically face demand that varies throughout the day. Examples include hospitals, mail processing facilities, airline reservation desks, and support hotlines. In each of these cases, management's goal is to find the best mix of hourly employees so that demand is satisfied at minimum cost. The problem is complicated by labor laws, union contracts and aperiodic fluctuations in demand.

The purpose of this paper is to describe a methodology for finding weekly schedules or tours for each employee in a given skill category. This means specifying the work days, their length, the daily start times, and the lunch breaks. Our focus is on tour scheduling at the United State Postal Service (USPS) mail processing and distribution centers where these specifications constitute a "bid job". The first component of the problem involves shift scheduling. The corresponding objective is to find the optimal crew size and daily work assignment for each member of the crew each day of the week. A *shift* is a set of consecutive time periods within a day and its *length* is the total amount of time it covers. In this study, a shift may vary in length depending on whether it is associated with a full-time or part-time employee, but cannot extend into the following day. This introduces the *discontinuity* property. The shift scheduling problem begins with the definition of permissible shifts, and concludes with the number of employees that should be assigned to each so as to satisfy daily demand period by period.

The second part of a weekly schedule requires the specification of the days off [1]. The number and characteristics of those days (weekend days, weekdays or combinations thereof) vary according to the organization and the type of industry in which it operates. A general consideration is that sufficient slack must be provided throughout the week so that the days off requirement is satisfied for every worker. The shift scheduling problem has to take this requirement into account when determining the optimal workforce size. An efficient way of doing this is by introducing lower bound constraints in the shift scheduling model, as described by Jarrah et al. [2].

The third component of a bid job is the lunch break. All shifts, except those shorter than a specific number of periods, require a break. Labor contracts determine the start time and the duration of this break. General practice is to create a *break window* for every shift—a set of consecutive periods during which a break may be given—and to assign a break within the window [3]. Depending on the nature of the work being performed, the breaks could be staggered or coincident.

Because an employee at lunch is off the clock, there should be sufficient resources to cover for him. Therefore, the schedule has to provide the necessary amount of over coverage, as well as make sure that everybody is taking his break within the prescribed window. The implicit modeling of break allowances for each employee is possible with the appropriate variables and constraints. However, this approach will only guarantee that there are a sufficient number of idle periods for every worker, i.e., the required slack is available. Determining who takes which period off is a transportation problem and is handled by a post-processing algorithm in our approach.

Malhotra and Ritzman [4] and Malhotra et al. [5] were among the first to consider the labyrinth of issues faced by the postal service in scheduling a flexible workforce. We begin with an analysis of a baseline model that reflects current practice at the USPS. The solution is stated in terms of the minimum number of full-time and part-time regular employees needed to satisfy the daily demand without violating the contractual labor ratio. Extensions are then considered and include the introduction of part-time flexible workers, allowance for two consecutive days off, and the relaxation of the constraint that a regular worker be given the same start time each day of the week. The full-time to part-time ratio is investigated parametrically.

In the next section, we review the personnel scheduling literature. This is followed by a description of the baseline model and solution approach. CPLEX [6] is used to solve the shift scheduling problem, and several post-processes written in visual basic for applications (VBA) are used for assigning breaks and days off. The models were tested on sample data provided by the USPS Oklahoma City processing and distribution center. The computational results indicate that problem instances of realistic size can be solved quickly, and that within the scope of the union contract considerable savings can be achieved by altering current practices.

2. Background and literature review

The accelerating growth of the service industry and the increasing cost of labor have led to renewed interest in personnel scheduling. Starting with Dantzig's [7] set covering formulation there have been numerous studies aimed at optimizing the workforce in the light of a variety of considerations. These include a non-homogeneous workforce in terms of skill, cost and efficiency measures, various labor requirements such as maximum work stretches, break definitions, days off and weekends off policies [1,3,8–12], as well as managerial issues such as queueing policies, maximum and minimum workforce constraints, start time restrictions, and objective function definitions [13–15].

Although there has been abundant research on personnel scheduling, almost all efforts have focused on a subset of the tour problem with the objective of minimizing total cost. Qualitative factors, however, have also been included in several studies. Bailey [9] added the cost of customer inconvenience due to understaffing. Beaumont [13] defined an objective function as a weighted sum of three criteria. The first criterion considered employee preferences in terms of longer or shorter work stretches in lieu of longer or shorter breaks. The second was designed to achieve an even yearly workload distribution, meaning that the number of days worked in all months should be approximately equal. The third was to keep staff on duty in numbers proportional to demand.

Segal [16] addressed a shift scheduling problem for telephone operators who were required to be given a lunch break and two relief breaks during their shift. He divided the day into ninety-six 15-min periods and used a network model to find solutions. Break assignments were made with a post-processing algorithm. Emmons [11], on the other hand, divided the week into two major parts—weekdays and weekend days. A constant workforce requirement was assumed for each part, and then master rotation schedules were created with days off and weekend allowances.

The derivation of daily or periodic workforce requirements has also been studied from different points of view. Mason [15] used both simulation and a heuristic to investigate the periodic case. Lin et al. [12] developed a decision support system for demand forecasting. Their study used regression and simulation to relate target abandonment rates to the required hourly workforce size.

Start time restrictions for the tour scheduling problem were recently addressed by Brusco and Jacobs [17] who developed a two-stage heuristic for constructing full-time and part-time schedules. Burns and Carter [1] were the first to provide a comprehensive solution to the days off assignment problem. They derived a set of lower bounds on the workforce size that took into account days off requirements as well as the requirement for A out of B weekends off. Their results assumed a maximum work stretch of six consecutive workdays for each employee.

Bartholdi et al. [10] used a circular 0–1 matrix for the general consecutive days off requirements associated with the (k, m) cyclic staffing problem in which employees are supposed to have k out of m days off. In the formulation, the circular matrix contains all possible patterns for k off-days with a “1” denoting the day when the worker is on, and a “0” the day when the worker is off. The solution provides the optimal number of employees for each shift type together with their off-day assignments. A number of solution techniques were considered including a linear nonsingular unimodular transformation of the decision variables, the conversion of the integer programming model into a series of network flow problems, and a linear relaxation followed by a round-off algorithm.

Alfares [18] provided an efficient algorithm for the tour scheduling problem that assigned two consecutive days off to employees without relying on a circular 0–1 matrix. Following the approach of Burns and Carter [1], he first developed lower bounds on the workforce size and then introduced them as additional constraints in a linear programming model. This was sufficient to assure integer solutions.

The implicit modeling of breaks was first proposed by Bechtold and Jacobs [3] who derived three constraints that collectively assured the feasibility of the break assignments. Aykin [8] used a similar objective function for the shift scheduling problem but extended the model to allow for multiple, rather than single, breaks and break windows. His approach called for a new set of decision variables to represent every possible combination of breaks. The resultant model was significantly smaller than its predecessors.

Beaumont [14] took a more expansive view and included worker available, the maximum number of workers who can start at the same time, the relative efficiency of a worker, the cost of making a customer wait, annual leave factors, the expected number of jobs an employee can complete in each period, the number of contractors, and the maximum number of jobs that can be done in each period, as parameters. His model also permitted a limited amount of queueing of customers, but ultimately was too large to be of practical value.

Because the number of variables in a model increases with the number of features included, the majority of recent research has focused on developing more sophisticated procedures for solving the underlying integer programs. Aykin [8], Bartholdi et al. [10], Bechtold and Jacobs [3], Brusco [19], Jarrah et al. [2] and Brusco and Jacobs [17] have all tried to reduce the number of variables in an effort to achieve faster convergence.

Looking at more advanced computational approaches, Brusco [19] evaluated the performance of dual all-integer cutting planes for solving the tour scheduling problem. He showed that a cutting plane enhanced by an LP objective cut and a sophisticated row selection rule improved solution times with respect to a commercial branch and bound code. Building on the work of Bailey [9], Jarrah et al. [2] were the first to address the days off scheduling problem and shift scheduling problem in an efficient manner. Their methodology used a set of aggregate variables and related cuts, along with a partial enumeration scheme and a feasibility heuristic, to find upper and lower bounds that

converged to near-optimal solutions. In the remainder of this paper we expand on these ideas and incorporate several new features in their formulation.

3. Model development

The baseline model for the shift scheduling problem was built in consultation with the Austin USPS processing and distribution center. This facility has three types for workers: full-time regulars (FTR), part-time regulars (PTR) and part-time flexibles (PTF). A regular employee has a constant start time for every working day. A flexible employee is not given a 5-day schedule but is called in when needed. He or she can have different start times on each day worked. Employees and managers generally prefer a constant start time for the entire week because of the consistency it offers. Because we wish to consider the simplest situation first, the entire workforce in the baseline model is assumed to be composed of regular employees only. PTFs will be included when various scenarios are examined.

A full-timer works $8\frac{1}{2}$ consecutive hours which includes a $\frac{1}{2}$ h allowance for a lunch break (in reality, he is off the clock for the $\frac{1}{2}$ lunch). A part-timer, on the other hand, may be assigned one of a variety of possible shift lengths. In this study, we consider five different lengths from 4 to $8\frac{1}{2}$ h (including the lunch breaks where applicable). All employees working 6 or more hours per day must be given a $\frac{1}{2}$ h lunch break.

The facility operates 24 h a day, 7 days a week and is driven by service standards for processing the mail within a fixed time of its arrival. The schedules for regular workers are determined through a bidding system but the schedule for the flexible part-timers can be changed weekly. In cases where the entire workforce is not sufficient to meet demand, temporary workers or overtime is used. The model discussed in this study is for the long-range planning problem rather than the weekly scheduling problem.

In the baseline model, the workday is divided into 48 periods, each 30 min long. For accounting purposes the first period starts at 7:00 a.m. The USPS uses three intervals in a day for managerial purposes. Shifts are evenly distributed in these intervals. A regular employee starts during one of these three intervals and works a shift of a predefined length.

The baseline model includes 9 different full-time shifts whose start times are listed in Table 1. A full-time shift is 17 periods ($8\frac{1}{2}$ h) long and includes the lunch break. There are 12 different start

Table 1
Full-time and part-time shift types included in the model

Shift type	Start times	Shift lengths
Full-time	7:00, 8:00, 9:00 (a.m.)	$8\frac{1}{2}$ h (17 periods)
	3:00, 4:00, 5:00 (p.m.)	
	8:30, 9:30, 10:30 (p.m.)	
Part-time	7:00, 8:00, 10:00, 11:00 (a.m.)	4, 5, $6\frac{1}{2}$, $7\frac{1}{2}$ and $8\frac{1}{2}$ h for each start time (8, 10, 13, 15 and 17 periods, respectively)
	1:00, 2:00, 4:00, 5:00 (p.m.)	
	7:00, 8:00, 9:30, 10:30 (p.m.)	

times for a part-time shift and 5 different lengths for each start time, making 60 different part-time shift types in all. Allowable part-time shift lengths are 8, 10, 13, 15 and 17 periods, including the breaks where applicable.

The breaks are typically assigned sometime between the 9th and the 12th period giving a break window of 4 periods or 2 h in length. Also, every worker must be given 2 days off a week. There are no restrictions in this regard but two consecutive days off, or at least one Saturday or Sunday, is preferable. The constraints needed to accommodate the first restriction are explained in the next section.

The following notation is used in the developments.

Indices

- d index for the days of the week; $d = 1, \dots, 7$
- t, s index for time periods during a day; $t = 1, \dots, 48$
- f index for the full-time shift types; $f = 1, \dots, n^F$
- p index for the part-time shift types; $p = 1, \dots, n^P$

Parameters

- c_f prorated weekly cost of full-time shift f
- c_p prorated weekly cost of part-time shift p
- G_{ft} 1 if full-time shift type f covers period t ; 0 otherwise
- P_{pt} 1 if part-time shift type p covers period t ; 0 otherwise
- D_{dt} demand for period t on day d
- k earliest period a break can begin for any of the permissible shifts
- q latest period a break can begin for any of the permissible shifts
- n^F number of full-time shifts
- n^P number of part-time shifts
- ρ full-time to part-time labor ratio

Sets

- B_s^F $\{j: \text{break window for full-time shift } j \text{ lies entirely between periods } s \text{ and } q\}$
- B_s^P $\{j: \text{break window for part-time shift } j \text{ lies entirely between periods } s \text{ and } q\}$
- F_s^F $\{j: \text{break window for full-time shift } j \text{ lies entirely between periods } k \text{ and } s\}$
- F_s^P $\{j: \text{break window for part-time shift } j \text{ lies entirely between periods } k \text{ and } s\}$
- T set of part-time shift types that have breaks
- M set of initial periods of the break windows, in ascending order
- N set of final periods of the break windows, in ascending order

Decision variables

- x_{fd} number of employees assigned to full-time shift type f on day d
- y_{pd} number of employees assigned to part-time shift type p on day d
- β_{dt} total number of breaks initiated in period t on day d
- w_f total number of full-time employees needed for shift type f
- v_p total number of part-time employees needed for shift type p

Model

$$\text{minimize } z = \sum_{f=1}^{n^F} c_f w_f + \sum_{p=1}^{n^P} c_p v_p \tag{1a}$$

subject to

$$\sum_{f=1}^{n^F} G_{ft} x_{fd} + \sum_{p=1}^{n^P} P_{pt} y_{pd} - \beta_{dt} \geq D_{dt}, \quad d = 1, \dots, 7, \quad t = 1, \dots, 48, \tag{1b}$$

$$\sum_{f=1}^{n^F} w_f \geq \rho \sum_{p=1}^{n^P} v_p, \tag{1c}$$

$$w_f \geq \frac{1}{5} \sum_{d=1}^7 x_{fd}, \quad f = 1, \dots, n^F, \tag{1d}$$

$$w_f \geq x_{fd}, \quad f = 1, \dots, n^F, \quad d = 1, \dots, 7, \tag{1e}$$

$$v_p \geq \frac{1}{5} \sum_{d=1}^7 y_{pd}, \quad p = 1, \dots, n^P, \tag{1f}$$

$$v_p \geq y_{pd}, \quad p = 1, \dots, n^P, \quad d = 1, \dots, 7, \tag{1g}$$

$$\sum_{t=k}^s \beta_{dt} - \sum_{f \in F_s^F} x_{fd} - \sum_{p \in F_s^P} y_{pd} \geq 0 \quad \forall s \in N, \quad d = 1, \dots, 7, \tag{1h}$$

$$\sum_{t=s}^q \beta_{dt} - \sum_{f \in B_s^F} x_{fd} - \sum_{p \in B_s^P} y_{pd} \geq 0 \quad \forall s \in M, \quad d = 1, \dots, 7, \tag{1i}$$

$$\sum_{f=1}^{n^F} x_{fd} + \sum_{p \in T} y_{pd} - \sum_{t=k}^q \beta_{dt} = 0, \quad d = 1, \dots, 7, \tag{1j}$$

$$w_f \geq 0, \quad v_p \geq 0, \quad \beta_{dt} \geq 0, \quad x_{fd} \geq 0, \quad y_{pd} \geq 0 \quad \forall t, k, p, d \text{ and all variables integer.} \tag{1k}$$

Objective function (1a) minimizes the total weekly cost of the workforce. For full-time shifts, the cost coefficients c_f are all the same because the shift lengths are constant. On the other hand, part-time shifts have a number of possible shift lengths, so c_p has a different value for each part-time shift p . If the cost parameters for any shift type differ according to the day (weekend days may cost more), then the variable definitions and objective function need to be modified slightly.

The first constraint, (1b), assures that the net workforce is sufficient to cover the demand for each period, every day. The net workforce is the total number of part-time and full-time employees whose shift definitions cover that specific period (i.e., the workers who are responsible for that period of the day), less those who have a break during that period. The 0–1 matrices (G and P) filter out the shifts that do not cover the period under consideration.

Constraint (1c), sometimes referred to as the ratio constraint, limits the number of part-time employees. In general, part-timers cost less than full-timers because they have less seniority and may not get some benefits. If there are no limits on the number of part-time employees, no full-time employees would be chosen by the model. This is an important issue given that almost every union contract has an upper bound on part-timers defined in various ways. In the case of the USPS, this bound is determined by head counts on the payroll. A ratio based on the total hours worked instead of the number of employees might be an alternative.

Constraints (1d)–(1g) were introduced by Burns and Carter [1] in a slightly different form, and are used to calculate lower bounds on the number of workers required to meet the daily demand. The first of these bounds, $L1$, is needed to assure that there is enough coverage so that every worker can take 2 days off a week. Constraints (1d) and (1f) correspond to $L1$ for the full-time and part-time workforce, respectively. The second lower bound, $L2$, is necessary to assure that there are a sufficient number of workers to cover the day with the highest demand. Constraints (1e) and (1g) correspond to $L2$ for the full-time and part-time workforce, respectively. Mathematically, the lower bounds can be represented as follows for each shift type f (similar constraints are applicable for shift type p):

$$L1 = \frac{1}{5} \sum_{d=1}^7 x_{fd}, \quad L2 = \max\{x_{fd}: d = 1, \dots, 7\}, \quad (2)$$

where x_{fd} denotes the number of workers assigned to shift type f on day d .

To account for breaks, three more constraints are needed. The first, (1h), is referred to as the *forward pass* constraint by Bechtold and Jacobs [3]. It assures that the total number of breaks initiated from period k up to a given period s exceeds the total number of employees who should have taken their breaks by that period. The employees included in the constraint are those whose break windows are fully covered through s , but not the ones who have the option of a break in some future period.

The second constraint (1i) is referred to as the *backward pass* constraint and assures that the total number of breaks that are initiated from some specific period s through the end of the day (or until the last period that can be taken as a break, which is q) exceeds the number of employees who are entitled to a break during this interval. In other words, there should be sufficient breaks in the future to satisfy the break requirement for the rest of the day.

These two constraints are needed to provide every employee with a break, but they are not sufficient to enforce the requirement that exactly one break be assigned to each worker entitled to one. Furthermore, they do not limit the break assignments to their respective ranges. Constraint (1j), which is known as the balance equation, is needed to assure that every worker is assigned a break and that it is within its permitted window.

4. Solution approach

The baseline model has five sets of decision variables, all of which are required to be integer, as indicated by (1k). Thus the formulation is a pure integer linear program and reflects the major constraints enumerated in the postal union contract. Additional features that reflect local practices are taken up in the next section.

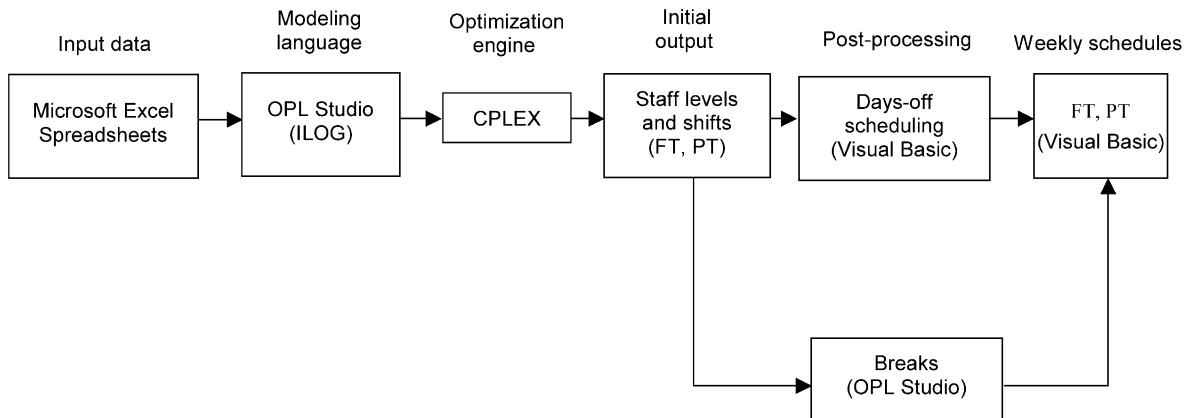


Fig. 1. Personnel scheduling software (computational flow).

The model was encoded using OPL Studio, a modeling language that allows the use of Microsoft Excel spreadsheets as an input source. It then calls CPLEX to solve the particular instance of (1a)–(1k). Fig. 1 depicts the computational flow of the solution approach.

The results from the optimization stage provide the number of each shift type required each day, and the number of breaks that should be initiated in each period. They do not specify the days off or when a break is to be taken during a shift. These issues are resolved with two post-processing algorithms, one that determines the days off assignments and the other the break assignments. The computational order is arbitrary.

The days off algorithm, which is explained below, is coded in VBA. The computations are triggered by a button on the Excel spreadsheet that contains the results of the baseline model. The days off assignment pattern for each worker is reported on the same spreadsheet as the optimization results.

The break assignment problem is solved as network model that is formulated in OPL Studio and solved with CPLEX. The days off and break assignments are retrieved simultaneously and placed on the same Excel spreadsheet. A code written in VBA is then called to build the final weekly schedules for each employee. The results are available through the OPL Studio window and in a user-specified text file. The various screens and scripts can be found in Binici [20].

4.1. Days off assignments

In explaining the post-processing algorithms, the word “enrolled” refers to employees whose names are listed on the payroll for a certain shift while the word “assigned” is used to identify those employees who are not only enrolled but also physically working on that shift on that day. The first post-processing task is to assign every worker 2 days off. Using a modified version of the Burns and Carter [1] algorithm, we begin with the workforce complement and shifts determined by the solution of optimization model (1), and then calculate the difference between the number of workers available and the number of required shifts each day. This value indicates the number of workers (by shift type) that can be given the day off. To make this more precise, let $Surplus_{fd}$ be the total number of people who are enrolled in a particular shift f but have not been assigned to work on

day d . These are the paid but idle workers for that shift. In particular,

$$\text{Surplus}_{fd} = w_f - x_{fd} \text{ for full-time shifts,}$$

$$\text{Surplus}_{pd} = v_p - y_{pd} \text{ for part-time shifts.}$$

From these definitions, we see that it is not possible to end up with a surplus value larger than the number of people enrolled in a shift. Moreover, the first lower bound on the size of the workforce guarantees that there will be sufficient surplus available the entire week for each shift.

The algorithm starts by finding the day with maximum surplus and assigns it as the first day off for the first employee on the current shift. Next, the neighboring days are searched for a positive surplus. If one is found, it is assigned as the second day off. If not, another day with a positive surplus is chosen arbitrarily and assigned as the second day off for the worker. Finally, both surplus values are reduced by one and the algorithm starts over for the second employee.

The constraints in the baseline model do not guarantee consecutive days off for every employee although many are likely to get them. Nevertheless, the only guarantee is the availability of 2 days off. An illustration of how the algorithm works for shift f follows.

Day, d	Mon	Tue	Wed	Thu	Fri	Sat	Sun	No. of employees
Surplus_{fd}	1	3	0	1	0	0	1	3

Iteration 1:

Assign worker 1 the day with the largest surplus

worker 1: {Tuesday, *}

check Monday and Wednesday, if available, assign

worker 1: {Tuesday, Monday}; after iteration the surplus is [0, 2, 0, 1, 0, 1]

Iteration 2:

Assign worker 2 the day with the largest surplus

worker 2: {Tuesday, *}

check Monday and Wednesday, if available, assign

if not find day with positive surplus (Thursday selected arbitrarily)

worker 2: {Tuesday, Thursday}; after iteration the surplus is [0, 1, 0, 0, 0, 0, 1]

Iteration 3:

Assign worker 3 the day with the largest surplus

worker 3: {Tuesday, *} (selected arbitrarily since both remaining days have excess of 1)

check Monday and Wednesday, if available, assign

if not, find day with positive surplus

worker 3: {Tuesday, Sunday}; after iteration the surplus is [0, 0, 0, 0, 0, 0, 0]

Stop.

The algorithm attempts to give as many workers as possible two consecutive days off but there is no guarantee that this number is maximized. For the USPS application, a job is associated with a

fixed schedule so employees do not rotate among assignments. Therefore, it is not possible to give weekends off with some frequency. If it were desirable to give workers at least one weekend day off, the logic could be adjusted accordingly. Note that for fixed f , because one of the lower bounds (1d)–(1e) is always tight in an optimal solution the algorithm always terminates with zero surplus.

4.2. Break assignments

The second post-processing operation involves the assignment of breaks to employees. Because this is essentially a transportation problem, a bi-partite network can be used to obtain a solution. In fact, all that is really needed is a feasible solution since no priorities are given for any of the periods with respect to a shift. Thus all arcs in the network have the same cost and bounds, implying that the objective function is constant. In some cases, though, it might be preferable to assign the break to the middle of a shift.

In the model, the supply nodes represent the shift types that are entitled to a break, and the demand nodes represent the periods that can be assigned a break. The fixed external flows associated with these nodes are the output of the baseline model. An arc connecting a shift node to a period node indicates that the period lies within the break window of that shift. The break window definitions determine the number of arcs that should terminate at any demand node. The shift definitions, on the other hand, determine the origins of these arcs.

To calculate the size of the network, we make use of the following data: n_F is the number of full-time shift types selected by the optimization model; n_P the number of part-time shifts types selected by the optimization model that require a break; and n_β the number of break periods in the optimization model.

The number of arcs needed to account for full-time break requirements is $n_\beta \times n_F$ and the number of arcs needed to account for part-time break requirements is $n_\beta \times n_P$. The number of demand nodes is $n_F + n_P$ and the number of supply nodes is $n_\beta = q - k + 1$. Further details are given in [20].

5. Extensions

Several extensions to the baseline model are possible by making small adjustments in either the constraints or parameter definitions. However, several of the proposed enhancements will have a significant impact on the quality of the solution.

5.1. Two consecutive days off

The first extension we consider is the enforcement of ‘two consecutive days off’ for all workers. Baker [21] and Tiberwala et al. [22] modeled this requirement explicitly for a single shift type. Their integer programming formulation is

$$\text{minimize } z = \sum_{j=1}^7 c_j x_j \quad (3a)$$

Table 2
Set of three nonadjacent numbers and its complement

<i>i</i>	<i>S_i</i>	<i>S_i[*]</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}

$$\text{subject to } \left(\sum_{j=1}^7 x_j \right) - x_d - x_{d-1} \geq r_d, \quad d = 1, \dots, 7, \tag{3b}$$

$$x_j \geq 0 \text{ and integer, } \quad j = 1, \dots, 7, \tag{3c}$$

where *d* is the index for the day of the week, *j* is the index for consecutive days off patterns (only seven are possible), *c_j* is the cost of working days off pattern *j*, *x_j* is the number of workers assigned to days off pattern *j*, *r_d* is the demand on day *d*, and *x₀* = *x₇*. To incorporate (3) into the baseline model (1), it would be necessary to create seven different versions of every shift, one for each pattern, so the number of shift variables would increase by a factor of 7. However, Alfares [18] used a simplified approach to provide a lower bound on the size of the workforce. Once the lower bound is found an additional constraint is introduced and the model is solved as a simple linear program. The additional constraint is as follows:

$$\sum_{j=1}^7 x_j = W,$$

where *W* is a lower bound on the workforce.

Alfares proved that for a (5, 7)-cyclic scheduling problem with consecutive days off requirements, the following equation yields the minimum workforce for shift type *f*:

$$w_f = \max \left\{ \left(\max_d x_{fd} \right), \left\lceil \frac{1}{5} \sum_{d=1}^7 x_{fd} \right\rceil, \left\lceil \frac{R_{\max}}{3} \right\rceil \right\}, \tag{4}$$

where

$$R_i = \sum_{d \in S_i^*} x_{fd}, \quad i = 1, \dots, 7,$$

$$R_{\max} = \max_i R_i$$

and *S_i^{*}* is the complement of the set of three nonadjacent numbers given in Table 2.

In fact, the first two terms in (4) are the first and second lower bounds (1d)–(1g) in the baseline model. So, the addition of the third lower bound will provide a schedule in which every employee

can be given two consecutive days off. The constraints that should be added to the baseline model are

$$w_f \geq \frac{1}{3} \sum_{d \in S_i^*} x_{fd}, \quad f = 1, \dots, n^F, \quad i = 1, \dots, 7,$$

$$v_p \geq \frac{1}{3} \sum_{d \in S_i^*} y_{pd}, \quad p = 1, \dots, n^P, \quad i = 1, \dots, 7.$$

5.2. Part-timers with 6 h/day, 6 days/week schedules

The second extension is concerned with a third type of employee who works 6 h/day, 6 days a week. Although these workers simply correspond to another shift definition, their inclusion in the baseline model requires the introduction of another set of parameters and variables. The resultant constraints would be the same as (1f) and (1g) except that the right-hand side of the equivalent of (1f) would be divided by 6 instead of 5. The objective function and the remaining constraints would have to be modified accordingly to account for the new part-time shift. The details are given in [20].

5.3. Variable start times

The third extension is to relax the ‘same start time’ requirement for shifts. To permit flexible start times, the total number of employees, w , enrolled in a full-time shift should be constrained as follows:

$$w \geq \frac{1}{5} \sum_{f=1}^{n^F} \sum_{d=1}^7 x_{fd}. \tag{1d'}$$

Neither the objective function nor the remaining constraints would be affected. Jarrah et al. [2] used (1d') in place of (1d) in their baseline model.

Variable start times allow an employee to work one shift one day and another shift (of the same length) another day. Complete freedom in making such assignments, though, can lead to impractical schedules as Jarrah et al. observed. In practice, a workday can be divided roughly into three intervals, I_a ($a = 1, 2, 3$), such that periods 1–16 belong to I_1 , periods 17–32 belong to I_2 , and periods 33–48 belong to I_3 . To conform with current USPS operations, the shift definitions were made so that an equal number of shifts start in each interval. Because a shift might cover periods in two consecutive intervals due to its length, the grouping of shifts into intervals is made according to the start time of the shift. An employee is permitted to work another shift in the same interval and of the same length. This relaxation does not provide any interchangeability between full-time and part-time shifts.

In the new formulation, because employees are not limited to a specific shift, the lower bound definitions and the objective function have to be changed. In the baseline model, the total number of employees enrolled for a specific full-time shift f is denoted by w_f . However, if variable start times are allowed then an employee enrolled for one shift can work another, which means that a separate decision variable is not necessary to track the enrollments. Instead, a variable accounting for

the total workforce in each interval is satisfactory. Therefore, let w_a be the total number of full-time employees enrolled in interval a ; $a = 1, 2, 3$, and $E_a = \{f: \text{shift } f \text{ starts in interval } a\}$; $a = 1, 2, 3$.

Consequently, the lower bound definitions (1d) and (1e) have to be replaced with the following:

$$w_a \geq \frac{1}{5} \sum_{f \in E_a} \sum_{d=1}^7 x_{fd}, \quad a = 1, 2, 3 \quad (\text{lower bound 1}), \quad (1d')$$

$$w_a \geq \sum_{f \in E_a} x_{fd}, \quad a = 1, 2, 3, \quad d = 1, \dots, 7 \quad (\text{lower bound 2}). \quad (1e')$$

The variable start time relaxation for part-timers is slightly different. In the case of full-timers, the shift lengths and thus the shift costs are constant which allows 100% interchangeability among the full-time workforce as long as the day-to-day assignments are within the same interval. However, there are 5 different shift lengths for the part-timers. If an employee is enrolled in a part-time shift then he or she can only work another part-time shift of the same length and cost. Therefore, to allow for variable start times for the part-time employees a separate variable for every possible shift length needs to be used. Let l be the index for part-time shifts of the same length, $l = 1, \dots, 5$ and define the following sets: $R_l = \{\text{part-time shifts of same length corresponding to index } l\}$, $l = 1, \dots, 5$, and $E_a = \{p: \text{shift-type } p \text{ is in interval } a\}$, $a = 1, 2, 3$.

The decision variable v_{la} is the total number of part-time employees enrolled in part-time shift of same length corresponding to index l that starts in interval a , and the lower bound equations (1f) and (1g) become

$$4v_{la} \geq \frac{1}{5} \sum_{p \in (R_l \cap E_a)} \sum_{d=1}^7 y_{pd}, \quad l = 1, \dots, 5, \quad a = 1, 2, 3 \quad (\text{lower bound 1}), \quad (1f')$$

$$v_{la} \geq \sum_{p \in (R_l \cap E_a)} \sum_{d=1}^7 y_{pd}, \quad l = 1, \dots, 5, \quad a = 1, 2, 3 \quad (\text{lower bound 2}). \quad (1g')$$

Objective function (1a) also needs to be changed in the following manner:

$$\text{minimize } z = \sum_{a=1}^3 \left(c_f w_a + \sum_{l=1}^5 c_l v_{la} \right), \quad (1a')$$

where c_f is the weekly cost of a full-time shift, and c_l the weekly cost of a part-time shift with length l .

The objective function value for this formulation is expected to be less than or equal to the optimal objective value for the baseline model because the flexible start times provide a relaxation of the shift assignment problem. Results are presented in the next section.

5.4. Ratio of full-timers to part-timers

The effect of the ratio constant, which prevents the number of part-time employees from exceeding some proportion of the total workforce, is another aspect of the problem that we wish to examine.

The baseline model calls for a minimum ratio of 4:1 between full-time and part-time employees. As might be expected, the output of this model indicates that ratio constraint (1c) is always binding, so altering the parameter ρ might significantly affect the total weekly cost. That is, the hourly cost of a part-time shift is lower than that of a full-time shift, so the total objective function is likely to decrease as the ratio decreases.

6. Results

Data supplied by the USPS Oklahoma City processing and distribution center were used to validate the model. The facility operates 24 h a day, 7 days a week. Daily staffing requirements for the automation equipment (OCRs, BCSs, and flat sorters) are listed in Table 7 in Appendix A.

Recall from Table 1 that the baseline model includes 9 full-time shift types, each $8\frac{1}{2}$ h in length, and 60 part-time shifts with 12 different start times and 5 different lengths. The shift definitions are listed in Table 8.

Both the part-timers and full-timers are treated as regular employees. This means that they have the same start times each day and get 2 days off a week if they are scheduled to work 6 or more hours per day. The full-time to part-time ratio is set to 4 implying that at least 80% of the workforce must be assigned full-time shifts.

The week starts on Saturday, and each day is divided into 48 half-hour periods with the first period beginning at 7:00 a.m. The hourly cost of a full-time employee is \$21, giving a weekly cost of \$840. The hourly cost of a part-time employee is \$16 but the weekly cost depends on the length of the shift worked.

The baseline model and the various extensions mentioned above were coded in OPL Studio and solved with CPLEX 6.5 on a Dell Dimension with a Pentium III processor and 128 MB RAM. In the computations, the relative MIP gap tolerance in CPLEX was set to 10^{-4} and the absolute MIP gap tolerance was set to 10^{-6} . The node limit was set to 15,000, and the frequency with which the built-in heuristic tried to find a feasible solution was set to 10. The choice of these parameters was somewhat arbitrary but seemed to produce good results. A node limit rather than a time limit was chosen to accommodate the fact that problem sizes varied considerably from one scenario to the next. Of course, any restrictions on run time, the size of the search tree, or memory usage mean that there is no guarantee of obtaining an optimal solution.

Tables 3 and 4 report input and output statistics. From these tables, we see that the baseline model contains 1092 constraints and 888 integer variables. The solution of \$96,280 was obtained in about 46 min and calls for 101 full-timers and 25 part-timers. It is worth noting that about 69% of the workforce is assigned two days off in a row and that the weekly schedule has 980 surplus (idle) hours built in. This conforms with USPS experience and is unavoidable given the worker requirements in Table 7.

The first scenario we report on involved a parametric analysis of ρ . The current value is 4 and was arrived at during the most recent round of USPS labor–management negotiations. Because the union would like this ratio to be as high as possible, it is critical for management to know the consequences of both increasing and decreasing ρ . The results indicate that when $\rho = 3$ (at least 75% of workforce must consist of full-timers), the objective function drops to \$95,040 (1.3%), although the size of the workforce increases by 2 to 128. Also, the number of surplus hours drops

Table 3
Model sizes and staffing results

	Number of constraints	Number of variables	Total cost per week (\$)	Number of full-timers	Number of part-timers	% 2 days off in a row
Baseline model	1092	888	96,280	101	25	68.9
Ratio 3:1	1092	888	95,040	96	32	65.6
Ratio 5:1	1092	888	97,880	105	21	63.5
Consecutive off-days	2127	1440	103,600	108	27	100
6 h/6 day workers	1140	936	95,952	100	25	72.4
Variable start time	684	837	95,800	101	25	62.1
Part-time flexible	1092	1308	94,976	100	N/A ^a	67.8

^aTotal number of part-timers is not explicit as there is no condition for a 5-day workweek for the PTF employees.

Table 4
Computational results

	Solution time (s)	No. of simplex iterations	Solution at node 0 (\$)	% Gap between LP relaxation and MIP	Total hours available	Total surplus (h)	Memory occupied
Baseline model	2755	590,600	94,316.84	2.081	4755	980	3,375,588
Ratio 3:1	2828	473,813	92,352.89	3.949	4740	965	3,64,2368
Ratio 5:1	2950	510,600	96,037.00	1.919	4805	1030	3,638,348
Consecutive off-days	4435	643,468	99,703.36	3.908	5075	1300	4,609,318
6 h/6 day workers	1524	651,465	99,703.36	0.794	4747	972	3,990,648
Variable start time	1180	642,100	94,313.44	1.576	4725	950	3,637,724
Part-time flexible	1684	473,301	94,228.00	2.390	4686	911	3,634,782

to 965 from the original 980. The baseline model uses 4041 h of full-time workers and 715 h of part-time workers per week. The modified model uses 3840 h of full-time workers (200 h less) but 900 h part-time workers (185 h more). The reduction in total idle time is only 15 h, but the objective function drops by \$1240. This indicates that there is a measurably advantage in decreasing ρ .

The opposite effect is observed when the ratio is increased to 5:1. This forces the percentage of full-timers to go up, inducing a \$1600 (1.67%) increase in the optimal objective value. The total number of full-timers increases to 105 from 101 and the total number of part-timers drops to 21 from 25. The total size of the workforce, however, remains at the original level of 126, but with total surplus moving up to 1030 h. The model size and computational effort remain about the same for either case.

The third scenario is concerned with giving everyone two consecutive days off. The impact is dramatic; the number of constraints in the model doubles and system performance deteriorates. In particular, the numbers of full-timers and part-timers increase to 108 and 27, respectively, and the weekly cost rises to \$103,600, a 7.6% increase. Moreover, computation time goes up by 61% and the number of surplus hours increases to 1300, indicating that a heavy price must be paid for adopting this policy.

The fourth scenario evaluates the use of a third type of employee who works 6 h a day, 6 days a week. Recall that the corresponding model requires additional variables and constraints but the increase is not significant. When the model is run with these workers, the objective function goes down to \$95,952, not much lower than the baseline value. Somewhat surprisingly, the computation time decreased by 45% to 1524 CPU s, implying that smaller problem instances may be harder to solve than larger ones.

The fifth scenario relaxes the constraint that an employee must start at the same time every day by permitting daily variations within an 8-h interval. We assigned 3 different full-time shifts and 20 different part-time shifts to each interval. The resultant model has 51 fewer variables and 408 fewer constraints than the baseline. Looking at Table 3 we see that the solution calls for the same number of full-time and part-time workers in either case but that the total cost decreased slightly to \$95,800. This is attributed to a greater use of shorter part-time shifts. The computation time decreased to 1180 CPU s from its original of 2755 CPU s, down 57%.

The final scenario investigates the use of part-time flexible workers (PTFs) along with the part-time regular workers (PTRs). PTFs are not given 5-day a week schedules but are called in when needed. That is, they do not necessarily work 5 days a week and there is no requirement that they start at the same time every day, so constraints (1f) and (1g) do not apply. To account for this type of employee, we must define a new set of decision variables that will appear in all but the lower bound constraints. A new term must also be added to the objective function equal to the total number of shifts worked by PTFs multiplied by the part-time shift cost. PTFs are treated as PTRs in the ratio constraint; however, the headcount of PTFs is not explicit because 5 shifts could be worked by the same person or 5 different people during a week. Therefore an approximation is needed to keep the number of part-time employees in proportion to the number of full-time employees. The following ratio constraint was used in the model:

$$\sum_{f=1}^{n^F} w_f \geq \rho \left(\sum_{p=1}^{n^P} \left(v_p + \frac{1}{5} \sum_{d=1}^7 ptf_{dp} \right) \right),$$

where ptf_{dp} denotes the number of PTFs assigned to shift p on day d .

The resultant model essentially requires twice as many part-time variables because each part-time shift must be duplicated. The number of constraints remains the same. Because we wanted to favor PTRs over PTFs the shift cost coefficients of the former were set marginally below those of the latter. The optimal objective function value was \$94,976, 1.35% below the baseline. Among the 725 part-time hours worked, 420 of them were worked by PTRs.

Fig. 2 plots the optimal objective function values of the scenarios investigated. The graph dramatizes the cost of giving each regular employee two consecutive days off in a row.

For the computational results presented in Table 4, CPLEX always ran to the 15,000 node limit. At termination, the percent gap was invariably $< 0.5\%$. This success was in part due to the fact that the LP relaxations at node 0 were very tight. As can be seen, the largest gap (difference between the final MIP solution and the first LP solution) was $< 4\%$. Much of the computational effort went into proving that one of the incumbents found in the first few minutes of the branch and bound process was indeed the best solution that was to be found.

The terminal values of the decision variables in the baseline model are given in Tables 5 and 6 for the full-time and part-time shifts, respectively. The days off algorithm and the break assignment

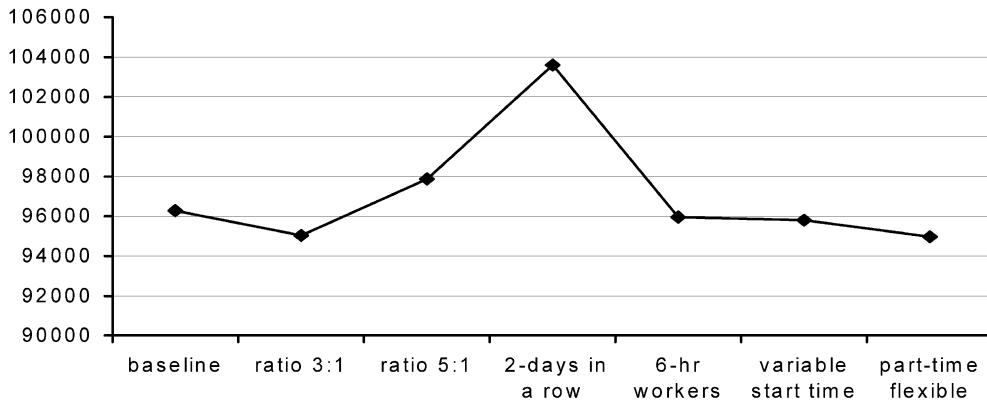


Fig. 2. Total cost for various scenarios.

Table 5

Optimal full-time shift assignments, x_{fd}

Shift type	Total enrolled	Number of employees working on each weekday (d)							
		Sat	Sun	Mon	Tue	Wed	Thu	Fri	
1	12	7	6	9	9	10	10	9	
2	0	0	0	0	0	0	0	0	
3	2	1	1	1	0	1	1	1	
4	14	9	6	11	10	12	11	11	
5	6	0	5	5	4	6	6	4	
6	0	0	0	0	0	0	0	0	
7	21	14	0	17	18	19	20	17	
8	8	4	3	7	6	8	7	5	
9	38	23	7	31	30	34	33	32	

algorithm were applied to these results to obtain the 5-day a week schedules given in Appendix B. In comparison to current workforce schedules at the Oklahoma City facility, our findings suggests that a 10% reduction in labor could be achieved without sacrificing performance. However, before reaching any conclusions it would be necessary to undertake an in-depth study of that facility to gain a better understanding of current staffing patterns, local operating policies, and weekly fluctuations in demand.

7. Discussion

Staff scheduling in the service industry is a multi-faced assignment problem with many variations that depend on union contracts, company policies and government regulation. Inclusion of breaks, weekend off requirements, interchangeable labor categories, ratio constraints, and a range of shift types, add both reality and complexity to the formulation. In this study, we investigated staff scheduling at a USPS facility. The problem involves the assignment of shifts, breaks and days off to each

Table 6
Optimal part-time shift assignments^a, y_{pd}

Shift type	Total enrolled	Number of employees working on each weekday (d)						
		Sat	Sun	Mon	Tue	Wed	Thu	Fri
16	1	0	0	1	1	1	1	1
19	1	0	0	1	1	1	1	1
20	1	1	1	0	1	1	1	0
24	6	5	0	4	6	4	5	6
28	2	0	0	2	2	2	2	2
29	2	2	0	2	1	2	2	1
31	1	1	0	1	1	0	1	1
32	3	3	0	2	3	3	1	3
36	6	3	0	6	4	6	6	5
37	1	1	0	0	1	1	1	1
50	1	1	1	0	1	0	1	1

^aOnly shifts with a positive number of employees are shown.

member of the regular workforce which are then processed into weekly tours. Given the 15,000 node limit, solutions were obtained in less than an hour for the sample data in all but one case. Although this time is reasonable for planning purposes, because we are solving a general integer linear program, the computational effort is, in theory, an exponential function of the number of variables in the problem. Adding more shift types to the model, for example, could have a dramatic effect on solution times, although this was not the case here. In fact, the solutions reported were always uncovered within the first few hundred nodes (about 1%) of the branch and bound tree. More telling, though, when the baseline model was run with CPLEX 7.1, the optimal solution was obtained within 3 min.

To make the model more versatile, it might be desirable to include several additional features. The first is the ability to handle a cross-functional workforce. In many organizations, higher skilled employees (with higher pay) can be assigned to jobs (at the higher pay) that require only a subset of those skills, and hence could be performed by a lower paid employee. This is sometimes referred to as downgrading [23]. When two skill levels are considered concurrently, the number of shift and workforce variables, and all but the break assignment constraints, must be duplicated in the model. Each additional skill level adds a corresponding number of variables and constraints.

A second feature involves the overlap of a shift from one day to the next. This is referred to as the continuous tour scheduling problem. Brusco and Jacobs [24] give two formulations that implicitly account for breaks and explicitly account for days off. When a shift extends to the next day but its break window remains in the previous day, as is the case with USPS facility operations, it is possible to modify Eq. (1b) to account for this situation. In particular, for each time period t in which there is overlap, the shifts from the previous day would be included in the constraint. The new formulation is almost identical to the original so solution times should not be affected.

A third feature takes the form of a post-processor and involves the assignment of workers to actual machines or tasks. We call this the *back assignment problem* and is the final step in the construction of a bid job. In the development of staffing requirements, the demand for workers with the same skills is often aggregated into a single number. The data in Table 7, for example, represent

the demand for all machine operators, and do not distinguish between the type of machine or its location in the facility. The back assignment problem can be formulated as a minimum cost network flow problem with side constraints, where the objective is to minimize the total number of switches an employee must make each day. Ideally, an employee would be assigned to a specific machine or location for an entire shift, but the demand data rarely permits such solutions, at least in the case of USPS facilities.

A fourth feature involves the selection of an optimal subset of starting times from a set of potential starting times. One implementation of this idea would limit the selection of starting time to one within a band of, say, 2 h. Each band could be modeled as an SOS type 1 constraint. The inclusion of this feature reduces the options available to the scheduler but may be desirable because it facilitates managerial oversight. Similarly, it may be desirable to include a minimum number of shifts of a particular type if at least one is selected. This condition can be modeled with semi-continuous variables but is likely to lead to a weakening of the LP relaxation.

Efforts are now underway to implement the baseline model as well as the variations highlighted in Section 5 in a prototype system that will run over the web. The Dallas Texas USPS facility is serving as the test site. The enhancements discussed above will be added once the prototype is validated.

Appendix A

Input data for model are given in Tables 7 and 8.

Table 7
Number of workers required in each period

	Period	Sat	Sun	Mon	Tues	Wed	Thu	Fri
7:00–7:30 a.m.	1	4	4	7	7	7	7	6
7:30–8:00	2	7	4	9	9	10	10	9
8:00–8:30	3	6	4	8	8	9	9	8
8:30–9:00	4	6	4	8	8	9	9	8
9:00–9:30	5	5	4	6	6	7	7	6
9:30–10:00	6	5	4	6	6	7	7	6
10:00–10:30	7	5	4	6	6	7	7	6
10:30–11:00	8	5	4	6	6	7	7	6
11:00–11:30	9	5	4	6	6	7	7	6
11:30–0:00 p.m.	10	5	4	6	6	7	7	6
0:00–0:30	11	9	7	12	12	13	13	12
0:30–1:00	12	9	7	12	12	13	13	12
1:00–1:30	13	9	7	12	12	13	13	12
1:30–2:00	14	8	7	11	11	12	12	11
2:00–2:30	15	11	7	15	15	17	17	15
2:30–3:00	16	15	7	20	20	22	22	20
3:00–3:30	17	15	7	20	20	22	22	20

Table 7 (continued)

	Period	Sat	Sun	Mon	Tues	Wed	Thu	Fri
3:30–4:00	18	15	7	20	20	22	22	20
4:00–4:30	19	22	7	29	29	32	32	29
4:30–5:00	20	22	7	29	29	32	32	29
5:00–5:30	21	22	7	29	29	32	32	29
5:30–6:00	22	23	7	30	30	33	33	30
6:00–6:30	23	24	10	32	32	35	35	32
6:30–7:00	24	24	10	32	32	35	35	32
7:00–7:30	25	24	10	32	32	35	35	32
7:30–8:00	26	24	10	32	32	35	35	32
8:00–8:30	27	24	10	32	32	35	35	32
8:30–9:00	28	25	10	33	33	36	36	33
9:00–9:30	29	26	10	35	35	39	39	35
9:30–10:00	30	26	10	35	35	39	39	35
10:00–10:30	31	26	15	35	35	39	39	35
10:30–11:00	32	42	15	56	56	62	62	56
11:00–11:30	33	32	15	42	42	46	46	42
11:30–0:00 a.m.	34	36	15	48	48	53	53	48
0:00–0:30	35	36	15	48	48	53	53	48
0:30–1:00	36	36	10	48	48	53	53	48
1:00–1:30	37	36	10	48	48	53	53	48
1:30–2:00	38	35	10	47	47	52	52	47
2:00–2:30	39	35	10	47	47	52	52	47
2:30–3:00	40	35	7	47	47	52	52	47
3:00–3:30	41	35	7	47	47	52	52	47
3:30–4:00	42	35	7	47	47	52	52	47
4:00–4:30	43	35	7	47	47	52	52	47
4:30–5:00	44	33	7	44	44	48	48	44
5:00–5:30	45	27	7	36	36	40	40	36
5:30–6:00	46	25	7	33	33	36	36	33
6:00–6:30	47	23	7	30	30	33	33	30
6:30–7:00 a.m.	48	20	7	27	27	30	30	27

Table 8
Full-time and part-time shift types included in the model

Shift type	Shift no.	Shift length (periods)	Starting period
Full-time	1	17	1
	2	17	3
	3	17	5
	4	17	17
	5	17	19
	6	17	21
	7	17	28
	8	17	30
	9	17	32

Table 8 (continued)

Shift type	Shift no.	Shift length (periods)	Starting period
Part-time	1	8	1
	2	10	
	3	13	
	4	15	
	5	17	
	6	8	3
	7	10	
	8	13	
	9	15	
	10	17	
	11	8	7
	12	10	
	13	13	
	14	15	
	15	17	
	16	8	9
	17	10	
	18	13	
	19	15	
	20	17	
	21	8	13
	22	10	
	23	13	
	24	15	
	25	17	
	26	8	15
	27	10	
	28	13	
	29	15	
	30	17	
	31	8	19
	32	10	
	33	13	
	34	15	
	35	17	
	36	8	21
	37	10	
	38	13	
	39	15	
	40	17	

Table 8 (continued)

Shift type	Shift no.	Shift length (periods)	Starting period
	41	8	
	42	10	
	43	13	25
	44	15	
	45	17	
	46	8	
	47	10	
	48	13	27
	49	15	
	50	17	
	51	8	
	52	10	
	53	13	30
	54	15	
	55	17	
	56	8	
	57	10	
	58	13	32
	59	15	
	60	17	

Appendix B

Results for baseline model are given in Tables 9 and 10.

Table 9
Complete weekly schedules for full-time employees

Worker no.	Off-day 1	Off-day 2	Break periods for the days worked						
			Sat	Sun	Mon	Tue	Wed	Thu	Fri
1	Sun	Sat	x	x	9	9	9	9	9
2	Sun	Sat	x	x	9	9	9	9	9
3	Sun	Sat	x	x	9	9	9	9	9
4	Sun	Sat	x	x	9	10	9	9	10
5	Mon	Sun	9	x	x	10	9	9	10
6	Tue	Mon	9	9	x	x	9	9	10
7	Fri	Sat	x	9	9	10	9	9	x
8	Tue	Mon	9	9	x	x	10	10	10
9	Wed	Tue	10	9	9	x	x	10	10
10	Thu	Wed	10	10	10	10	x	x	10

Table 9 (continued)

Worker no.	Off-day 1	Off-day 2	Break periods for the days worked						
			Sat	Sun	Mon	Tue	Wed	Thu	Fri
11	Fri	Thu	10	10	10	10	10	x	x
12	Sun	Fri	10	x	10	10	10	10	x
13	Tue	Mon	16	14	x	x	13	15	13
14	Sat	Fri	x	13	13	14	14	13	x
15	Sun	Sat	x	x	26	25	28	28	25
16	Sun	Sat	x	x	28	28	28	28	26
17	Sun	Sat	x	x	28	28	28	28	27
18	Sun	Sat	x	x	28	28	28	28	28
19	Sun	Sat	x	x	28	28	28	28	28
20	Tue	Mon	25	25	x	x	28	28	28
21	Sun	Mon	28	x	x	28	28	28	28
22	Tue	Mon	28	25	x	x	28	28	28
23	Thu	Wed	28	26	28	28	x	x	28
24	Fri	Thu	28	27	28	28	28	x	x
25	Sun	Fri	28	x	28	28	28	28	x
26	Tue	Wed	28	27	28	x	x	28	28
27	Sun	Fri	28	x	28	28	28	28	x
28	Tue	Thu	28	28	28	x	28	x	28
29	Sat	Fri	x	28	30	30	29	29	x
30	Sat	Fri	x	29	30	30	30	29	x
31	Sat	Sun	x	x	30	30	30	30	30
32	Sat	Tue	x	29	30	x	30	30	30
33	Sat	Tue	x	30	30	x	30	30	30
34	Sat	Mon	x	30	x	30	30	30	30
35	Sun	Sat	x	x	36	36	36	36	36
36	Sun	Sat	x	x	36	36	36	36	36
37	Sun	Sat	x	x	36	36	36	36	36
38	Sun	Sat	x	x	36	36	36	36	36
39	Sun	Sat	x	x	36	36	36	36	36
40	Sun	Sat	x	x	36	36	36	36	36
41	Sun	Sat	x	x	36	36	36	36	36
42	Sun	Mon	36	x	x	37	36	37	37
43	Sun	Mon	36	x	x	37	37	37	37
44	Sun	Mon	36	x	x	37	37	37	37
45	Sun	Mon	36	x	x	37	37	37	37
46	Sun	Fri	36	x	38	37	37	37	x
47	Sun	Fri	36	x	38	37	37	37	x
48	Sun	Fri	37	x	39	37	37	37	x
49	Sun	Fri	37	x	39	39	37	37	x
50	Sun	Thu	37	x	39	39	37	x	37
51	Sun	Wed	37	x	39	39	x	39	37
52	Sun	Wed	37	x	39	39	x	39	37
53	Sun	Tue	37	x	39	x	38	39	39
54	Sun	Tue	38	x	39	x	39	39	39
55	Sun	Tue	38	x	39	x	39	39	39
56	Sun	Sat	x	x	38	38	38	39	39
57	Sun	Sat	x	x	38	38	38	39	39

Table 9 (continued)

Worker no.	Off-day 1	Off-day 2	Break periods for the days worked						
			Sat	Sun	Mon	Tue	Wed	Thu	Fri
58	Sun	Sat	x	x	38	39	38	39	39
59	Fri	Sat	x	39	38	39	38	39	x
60	Sun	Mon	38	x	x	39	38	40	39
61	Tue	Fri	38	41	38	x	38	40	x
62	Sun	Fri	38	x	38	39	38	40	x
63	Tue	Thu	38	41	40	x	38	x	39
64	Sun	Sat	x	x	40	40	40	40	40
65	Sun	Sat	x	x	40	40	40	40	40
66	Sun	Sat	x	x	40	40	40	40	40
67	Sun	Sat	x	x	40	40	40	40	40
68	Sun	Sat	x	x	40	40	40	40	40
69	Sun	Sat	x	x	40	40	40	40	40
70	Sun	Sat	x	x	40	41	40	41	40
71	Sun	Sat	x	x	41	41	41	41	40
72	Sun	Sat	x	x	41	41	41	41	41
73	Sun	Sat	x	x	41	41	41	41	41
74	Sun	Sat	x	x	41	41	41	41	41
75	Sun	Sat	x	x	41	41	41	41	41
76	Sun	Sat	x	x	41	41	41	41	41
77	Sun	Sat	x	x	41	41	41	41	41
78	Sun	Sat	x	x	41	42	41	41	41
79	Sun	Mon	40	x	x	42	41	42	41
80	Sun	Mon	40	x	x	42	42	42	42
81	Sun	Mon	40	x	x	42	42	42	42
82	Sun	Mon	40	x	x	42	42	42	42
83	Sun	Mon	40	x	x	42	42	42	42
84	Sun	Mon	41	x	x	42	42	42	42
85	Sun	Mon	41	x	x	42	42	42	42
86	Sun	Fri	41	x	42	43	42	42	x
87	Sun	Fri	41	x	42	43	42	42	x
88	Tue	Wed	41	40	42	x	x	43	42
89	Sun	Fri	41	x	42	43	43	43	x
90	Tue	Wed	42	41	42	x	x	43	42
91	Sun	Fri	42	x	42	43	43	43	x
92	Tue	Wed	42	41	42	x	x	43	43
93	Sun	Fri	42	x	42	43	43	43	x
94	Tue	Wed	42	42	43	x	x	43	43
95	Thu	Fri	42	42	43	43	43	x	x
96	Sun	Thu	43	x	43	43	43	x	43
97	Tue	Thu	43	42	43	x	43	x	43
98	Sun	Thu	43	x	43	43	43	x	43
99	Tue	Thu	43	42	43	x	43	x	43
100	Sun	Tue	43	x	43	x	43	43	43
101	Sun	Tue	43	x	43	x	43	43	43

Table 10
Complete weekly schedules for part-time employees

Worker no.	Off-day1	Off-day2	Break periods for the days worked						
			Sat	Sun	Mon	Tue	Wed	Thu	Fri
102 ^a	Sat	Sun	x	x	-	-	-	-	-
103	Sat	Sun	x	x	18	18	17	17	18
104	Mon	Fri	17	19	x	17	17	17	x
105	Sun	Sat	x	x	21	21	21	21	21
106	Sun	Mon	21	x	x	21	21	22	22
107	Sun	Mon	21	x	x	22	21	22	22
108	Sun	Fri	21	x	21	22	21	22	x
109	Sun	Wed	22	x	22	22	x	22	22
110	Sun	Wed	22	x	22	22	x	22	22
111	Sat	Sun	x	x	24	23	24	24	23
112	Sat	Sun	x	x	25	24	25	25	24
113	Sun	Fri	23	x	23	23	23	23	x
114	Sun	Tue	24	x	23	x	23	23	23
115 ^a	Sun	Wed	-	x	-	-	x	-	-
116 ^a	Sun	Mon	-	x	x	-	-	-	-
117 ^a	Sun	Thu	-	x	-	-	-	x	-
118 ^a	Sun	Thu	-	x	-	-	-	x	-
119 ^a	Sun	Sat	x	x	-	-	-	-	-
120 ^a	Sun	Sat	x	x	-	-	-	-	-
121 ^a	Sun	Sat	x	x	-	-	-	-	-
122 ^a	Sun	Fri	-	x	-	-	-	-	x
123 ^a	Sun	Tue	-	x	-	x	-	-	-
124 ^a	Sun	Tue	-	x	-	x	-	-	-
125 ^a	Sun	Mon	-	x	x	-	-	-	-
126	Thu	Mon	35	37	x	35	x	37	37

^aThese shifts are shorter than 6 h so they are not entitled to a break.

References

- [1] Burns RN, Carter MW. Work force size and single shift schedules with variable demands. *Management Science* 1985;31(5):599–607.
- [2] Jarrah AIZ, Bard JF, deSilva AH. Solving large-scale tour scheduling problems. *Management Science* 1994;40(9):1124–45.
- [3] Bechtold SE, Jacobs LW. Implicit modeling of flexible break assignments in optimal shift scheduling. *Management Science* 1990;36(11):1339–51.
- [4] Malhotra MK, Ritzman LP. Scheduling flexibility in the service sector: a postal case study. *Production and Operations Management* 1994;3:100–17.
- [5] Malhotra MK, Ritzman LP, Benton WC, Leong GK. A model for scheduling postal distribution employees. *European Journal of Operational Research* 1992;58:374–85.
- [6] CPLEX 6.5 User's Manual, ILOG, Inc., <http://www.ilog.com>, Mountain View, CA, 1999.
- [7] Dantzig GB. A comment on Edie's traffic delays at toll booths. *Operations Research* 1954;2(3):339–41.
- [8] Aykin T. Optimal shift scheduling with multiple break windows. *Management Science* 1996;42(4):591–602.
- [9] Bailey J. Integrated days off and shift personnel scheduling. *Computers & Industrial Engineering* 1985;9(4):395–404.
- [10] Bartholdi JJ, Orlin JB, Ratliff HD. Cyclic scheduling via integer programs with circular ones. *Operations Research* 1980;28:1074–85.

- [11] Emmons H. Work-force scheduling with cyclic requirements and constraints on days off, weekends off, and work stretch. *IIE Transactions* 1985;17(1):8–15.
- [12] Lin CKY, Lai KF, Hung SL. Development of a workforce management system for a customer hotline service. *Computers & Operations Research* 2000;27:987–1004.
- [13] Beaumont N. Using mixed integer programming to design employee rosters. *Journal of the Operational Research Society* 1997;48:585–90.
- [14] Beaumont N. Scheduling staff using mixed integer programming. *European Journal of Operational Research* 1997;98(3):473–84.
- [15] Mason AJ, Ryan DM, Panton DM. Integrated simulation, heuristic and optimisation approaches to staff scheduling. *Operations Research* 1998;46(2):161–75.
- [16] Segal M. The operator scheduling problem : a network flow approach. *Operations Research* 1974;22:803–24.
- [17] Brusco M, Jacobs LW. Personnel tour scheduling when starting time restrictions are present. *Management Science* 1998;44(4):534–47.
- [18] Alfares HK. An efficient two-phase algorithm for cyclic days-off scheduling. *Computers & Operations Research* 1997;25:913–23.
- [19] Brusco MJ. Solving personnel tour scheduling problems using the dual all-integer cutting plane. *IIE Transactions on Operations Engineering* 1998;30(9):835–44.
- [20] Binici C. Staff scheduling in the service industry: application to the United States Postal Service processing and distribution centers. Master's thesis, Graduate program in operations research and industrial engineering. The University of Texas, Austin, 2000.
- [21] Baker KR. Workforce allocation in cyclical scheduling problems: a survey. *Operations Research* 1976;22(1):155–67.
- [22] Tiberwala R, Philippe D, Browne J. Optimal scheduling of two consecutive idle periods. *Management Science* 1972;19(1):71–5.
- [23] Dawid H, Konig J, Strauss C. An enhanced rostering model for airline crews. *Computers & Operations Research* 2001;28:671–88.
- [24] Brusco M, Jacobs LW. Optimal models for meal-break and start-time flexibility in continuous tour scheduling. *Management Science* 2000;46(12):1630–41.

Jonathan F. Bard is a Professor of Operations Research & Industrial Engineering in the Mechanical Engineering Department at the University of Texas at Austin. He holds the *Industrial Properties Corporation Endowed Faculty Fellowship* and serves as the Associate Director of the Center for the Management of Operations and Logistics, and as the Area Coordinator for the OR/IE Program. His research interests are in airline operations, manufacturing systems, hierarchical optimization, and vehicle routing. He is currently the Editor of *IIE Transactions on Operations Engineering* and an Associate Editor for *Management Science*.

Canan Binici received a B.S. in Industrial Engineering from The Middle East Technical University in Ankara, Turkey. In 2000, she received an M.S. in Operations Research and Industrial Engineering from The University of Texas at Austin. She is now working as a systems analyst in the service sector.

Anura deSilva is the CEO of Planmatics, Inc., of Rockville Maryland. He specializes in technology planning and the use of decision support systems to enhance the efficacy of technology. Under his leadership, his company has introduced technology planning and process improvement systems to the United States Postal Service, the US Employment and Training Administration and DeutschePost (DPWN).