Personnel Shift Assignment: Existence Conditions and Network Models

Yeroon van den Berg

Department of Mechanical Engineering, University of Twente, The Netherlands

David M. Panton

Centre for Mathematical Applications, University of South Australia, The Levels
Pooraka 5095, South Australia

The personnel scheduling problem is known to be a five-stage process in which the final stage involves the assignment of shifts to the days worked in the schedule. This paper discusses the existence conditions for both continuous and forward rotating shift assignments and heuristic network algorithms for the determination of such assignments. Results generated for a number of test problems demonstrate, first, that the network devised to search for continuous solutions produces these solutions in a high proportion of cases where such solutions are known to exist. Second, for more general problems, the algorithm is shown to be efficient in its ability to generate either continuous or rotating solutions. *© 1994 John Wiley & Sons, Inc.*

1. INTRODUCTION

Personnel scheduling problems have been the source of considerable study over many years. In the most recent survey of work done in this area [7], the personnel scheduling problem has been described as a five-stage process. Stage 1 determines the temporal personnel requirements and results in a *Shift Requirements Matrix* (SRM), $req[s, d]$, specifying how many persons are required during shift s on day d. A *shift* is a period of work with a specified starting and ending time. It will be assumed that each person does not work more than one shift per day. We will let $\mathcal{S}$ be the set of all shift types and will assume that days d belong to a weekly planning period given by $\mathcal{D} = \{M, T, W, Th, F, Sa, Su\}$. In the context of this paper, we will allow $\mathcal{D}$ to be a cyclic set where $Su + 1 = M$. Stage 2 involves the determination of the total workforce requirement, given off-weekend and other recreational constraints, and is often solved concurrently with Stage 1. Stage 3 involves the identification of recreation blocks, and along with Stage 4, which involves the placement of these blocks in the schedule, creates a *Single Shift Schedule* (SSS) specifying which days are worked and which days are recreation days in the entire schedule. An example of an SRM and an SSS are given in Table I. The schedule shown in Table I can be viewed as either *fixed* or *cyclic*.

In a fixed schedule, each of the five personnel are associated with a line that they repeat from week to week, whereas in a cyclic schedule, personnel rotate from one line (week) to the next, completing the cycle in 5 weeks. In this paper, it is assumed that we are dealing with cyclic schedules. Within the SSS, we can identify contiguous sequences of days worked, which we call *workstretches.* An example of a workstretch is the sequence $W \rightarrow Sa$ shown in week 3 of the SSS in Table I.

The final stage (5) for the completion of a personnel schedule is the subject of this paper, namely, the assignment of shifts to the single-shift schedule. Input for this problem consists of the SRM and the SSS, together with any constraints that might be placed on the nature of the solution. The most common requirement in the workforce is for a *continuous shift assignment* (CoSA) in which there is only one shift type associated with each workstretch.

TABLE I. Examples of a shift requirements matrix for two shift types and a single-shift schedule

Shift Requirements Matrix (SRM)							
Shift Type	*M*	*T*	*W*	*Th*	*F*	*Sa*	*Su*
*s*1	2	1	2	2	2	2	2
*s*2	2	1	2	2	1	2	2

Single-Shift Schedule (SSS)							
Week no.	*M*	*T*	*W*	*Th*	*F*	*Sa*	*Su*
1	1	1	1	1	1	0	1
2	1	0	1	1	0	1	1
3	1	0	1	1	1	1	0
4	1	1	0	0	1	1	1
5	0	0	1	1	0	1	1

TABLE II. A continuous shift assignment (CoSA) for the problem given in Table I

Week No.	*M*	*T*	*W*	*Th*	*F*	*Sa*	*Su*
1	*s*1	*s*1	*s*1	*s*1	*s*1	0	*s*1
2	*s*1	0	*s*2	*s*2	0	*s*2	*s*2
3	*s*2	0	*s*1	*s*1	*s*1	*s*1	0
4	*s*2	*s*2	0	0	*s*2	*s*2	*s*2
5	0	0	*s*2	*s*2	0	*s*1	*s*1

An example of a CoSA solution for the problem given in Table I is shown in Table II.

Should shift changes be necessary within a workstretch, then this is referred to as shift rotation. When rotations occur from an earlier shift starting time to a later shift starting time on consecutive days, this is referred to as a *forward rotating shift assignment* (FoRSA). An example of a forward rotating shift assignment is shown in week 3 of the schedule in Table III, where *s*1 is a shift with an earlier starting time than that of shifts of type *s*2. Rotations of this type are desirable for two reasons: First, union conditions may require that a minimum period of time is allowed between the completion of a shift started on one day and the commencement of a shift started on the next day. For instance, an 8 hour shift starting at 5 P.M. will end at 1 A.M. on the next day, which excludes a 6 A.M. start on that day if a minimum of 8 hours is required between shifts. In this situation, the problem arises as a result of the second shift starting at an earlier time than that of the preceding one. Second, recent studies involving sleep patterns of personnel on shift work [3] indicate that shift rotations of this type are undesirable. Consecutive shifts of this type are called *backward rotations,* an example of which is shown between weeks 2 and 3 of the schedule in Table III.

Methods for shift assignment have received little attention in the literature. Conditions for the existence of CoSA and FoRSA solutions appear to have received *none.* Early approaches to the assignment problem [4, 5] use a two-part strategy. First, they identify the personnel requirements for each shift type and then solve stages 3 and 4 separately for each shift. Second, they sequence the shift-specific schedules and select appropriate shift schedules using a lexicographic procedure. Mathematical programming methods for shift assignment have also been discussed in [1, 2, 6]; however, no satisfactory approach detailing efficient methods for seeking CoSA solutions and what to do should these not be attainable has been published.

In Section 2 of this paper, we discuss the computational complexity of CoSA and FoRSA and general conditions under which SRM/SSS combinations give rise to either CoSA or FoRSA solutions. In Section 3, we describe decomposition network algorithms that provide CoSA solutions, in a high percentage of test cases or, alternatively, generate rotating solutions should CoSA solutions not exist. In Section 4, the results of experiments using these algorithms are discussed. Conclusions are presented in Section 5.

2. COMPLEXITY AND EXISTENCE CONDITIONS

2.1. Complexity of CoSA and FoRSA

Theorem 2.1 shows the *NP*-completeness of CoSA. First, we define the decision problem COSA.

COSA

Instance: A single shift schedule (SSS) and a shift requirements matrix (SRM).

Question: Is there an assignment of shifts to the workstretches in the SSS that satisfies the SRM?

Theorem 2.1. COSA *is NP-complete.*

Proof. Whether a given solution satisfies the SRM can be checked easily, by summing the occurrences of shifts on every day of the week; hence, COSA $\in NP$.

TABLE III. An example of a shift assignment containing both a forward rotation and a backward rotation

Week No.	*M*	*T*	*W*	*Th*	*F*	*Sa*	*Su*
1	0	0	*s*1	*s*1	*s*1	*s*1	*s*1
2	*s*1	0	0	*s*2	*s*2	*s*2	*s*2
3	*s*1	*s*1	0	0	*s*1	*s*1	*s*2
4	*s*2	*s*2	*s*2	0	0	*s*2	*s*2

We show that the following known *NP*-complete problem PARTITION is a special case of COSA.

PARTITION
Instance: A finite set A and a "size" $s(a) \in \mathrm{IN}^+$ for each $a \in A$.
Question: Is there a subset $A' \subset A$ such that $\sum_{a \in A'} s(a) = \sum_{A \setminus A'} s(a)$?

Define for every $a \in A$ a workstretch of $7 \cdot s(a)$ days and put them in the SSS with 1 or more days off in between. Assume that $S = \sum_{a \in A} s(a)$ is even; otherwise, no solution exists. Let there be two shift types ($s1$, $s2$), and let all SRM elements be equal to $S/2$. The elements of A' correspond to workstretches to which shift type $s1$ is assigned. ■

We conjecture that the problem of finding a forward rotating assignment with at most k rotations is *NP*-complete ($k \in \mathrm{IN}^+$).

2.2. Conditions for the Existence of CoSA

Consider the problem of finding a continuous assignment initially for the single shift type $s1$ whose requirements are specified in Table I. It should be noted that in any workstretch a particular day can occur at most once more than another day. Therefore, when looking at days M, T, to satisfy the requirements on these days, there should be at least one workstretch in which M occurs once more than does T. This must be a workstretch that ends on M whose length is not a multiple of 7. For the same reason, when looking at days T, W, there should be one workstretch available that starts on W to be able to satisfy the requirements for these days. The above observations can be formalized in the following definitions:

Definition 2.1. *For all $s \in \mathcal{S}$, $d \in \mathcal{D}$, the starting requirements are given by $sr[s, d] = \max\{0, req[s, d] - req[s, d-1]\}$ and the ending requirements are given by $er[s, d] = \max\{0, req[s, d] - req[s, d+1]\}$. For all $d \in \mathcal{D}$, the total starting requirements are $sr[d] = \sum_{s \in \mathcal{S}} sr[s, d]$ and the total ending requirements are $er[d] = \sum_{s \in \mathcal{S}} er[s, d]$.*

Definition 2.2. *For all $d \in \mathcal{D}$, $sa[d]$ (starts available) denotes the number of workstretches in the SSS that start with day d. Likewise, $ea[d]$ (ends available) denotes the number of workstretches that end on day d. $sa^*[d]$ and $ea^*[d]$ are equal to $sa[d]$ or $ea[d]$ less the number of workstretches starting or ending on day d having length $7k$, $k \in \mathrm{IN}^+$.*

Example. Consider the two-shift SRM given in Table I. Table IV shows the *total* starting and ending requirements from the SRM.

TABLE IV. Starting and ending requirements associated with the shifts in Table I

	M	*T*	*W*	*Th*	*F*	*Sa*	*Su*
sr[*d*]	0	0	2	0	0	1	0
er[*d*]	2	0	0	1	0	0	0

The available starting and ending days in the SSS, shown in Table V, can be seen to be sufficient to satisfy these requirements.

Relationships between starting and ending requirements from the SRM given in the lemma below, and the definition that follows, will be useful in establishing a general set of conditions for the existence of a continuous assignment.

Lemma 2.1. $sr[s, d] - er[s, d-1] = req[s, d] - req[s, d-1]$.

Proof. Since by definition $sr[s, d]$ represents the positive part of $req[s, d] - req[s, d-1]$ and $-er[s, d-1]$ represents the negative part of $req[s, d] - req[s, d-1]$, then, clearly, $sr[s, d] - er[s, d-1] = req[s, d] - req[s, d-1]$. ■

Definition 2.3. *A chain is a set of workstretches within the SSS that can be concatenated in such a way that a following workstretch starts on the day after the ending day of the previous workstretch in the chain. A cyclic chain is a chain that contains a total number of days that is a multiple of 7.*

Example. Workstretches $M \rightarrow Sa$; $Su \rightarrow Th$ form an example of a chain, and $M \rightarrow Sa$; $Su \rightarrow Th$; $F \rightarrow Su$ form an example of a cyclic chain of length 14 days. Note that in this case the ending day for the last workstretch precedes the starting day for the first workstretch.

The following theorem gives the necessary and sufficient conditions under which a CoSA can be obtained.

Theorem 2.2. *CoSA* $\Leftrightarrow$

(*i*) *Workstretches with shift-type s, $s \in \mathcal{S}$, can be concatenated into chains that are either cyclic or go from*

TABLE V. Available starting and ending days

	M	*T*	*W*	*Th*	*F*	*Sa*	*Su*
sa[*d*]	1	0	3	0	1	2	1
ea[*d*]	2	1	0	2	1	1	1

a starting day d_1 with $sr[s, d_1] > 0$ to an ending day d_2 with $er[s, d_2] > 0$.

(ii) There are $sr[s, d]$ noncyclic chains starting and $er[s, d]$ noncyclic chains ending on day d with shift s.

(iii) For each shift type, the number of days in the chains equals the demand.

Proof. ($\Leftarrow$): If we assign the corresponding shift type to the workstretches in the noncyclic chains, then the starting ($sr[s, d]$) and ending ($er[s, d]$) requirements are satisfied, so that we obtain equal remaining requirements for each shift type. When the shift types are also assigned to the workstretches in the cyclic chains, then the remaining requirements are still equal, since each day is represented the same number of times in the cycles. Since precisely enough days are assigned for each shift type, the remaining requirements must be zero, whereupon a CoSA is found.

($\Rightarrow$): Since a CoSA exists, we can assign a shift type to every workstretch so that the shift requirements are satisfied. Accordingly, the number of workstretches with shift s that start on day d less the number that end on day $d - 1$ is equal to $sr[s, d] - er[s, d - 1] = req[s, \mathrm{d}] - \mathrm{req}[\mathrm{s}, d - 1]$ from Lemma 2.1. We pairwise concatenate as many workstretches with the same shift type that end on day $d - 1$ with those that start on day d. Consequently, if $req[s, d] \geq req[s, d - 1]$, then $sr[s, d]$ workstretches remain without predecessors, and if $req[s, d] \leq req[s, d - 1]$, then $er[s, d - 1]$ workstretches remain without successors. After doing this for all s and d, the workstretches form chains with a certain shift type that are either cyclic or go from a day with $sr[s, d] > 0$ to a day with $er[s, d] > 0$. For all s and d, $sr[s, d]$ noncyclic chains start and $er[s, d]$ noncyclic chains end on day d. Clearly, in a CoSA, the number of assigned days equals the demand for each shift type. ■

Theorem 2.2 does not provide us with a clear way of identifying SRM/SSS combinations that will be guaranteed to provide CoSA solutions. However, it is possible to deduce from this result a property that the SSS must possess that is easy to test for and that is necessary for a CoSA to occur. First, consider the following definition:

Definition 2.4. *An SSS is said to have sufficient starting days (S), if for every $d \in \mathcal{D}$ $sa^*[d] \geq sr[d]$.*

Corollary 2.1. *A necessary condition for a continuous assignment of shifts from an SRM to an SSS is that there are sufficient starting days in the SSS to satisfy these requirements in the SRM, i.e., CoSA $\Rightarrow$ S.*

Proof. This result follows immediately from Theorem 2.2 parts (i) and (ii), since in order to provide $sr[s, d]$ noncyclic chains starting on day d for shift type s, the SSS must contain at least that number of workstretches that start on that day. ■

Note that an equivalent condition involving ending day availability and requirements can be deduced as a consequence of the following definition and lemma:

Definition 2.5. *By $p_{d-1,d}$, $d \in \mathcal{D}$, we denote the number of times that days $d - 1$ and d appear consecutively in workstretches.*

Lemma 2.2. $sa[d] - ea[d - 1] = sr[d] - er[d - 1]$.

Proof. Since

$$p_{d-1,d} + sa[d] = \sum_{s \in \mathcal{S}} req[s, d]$$

$$p_{d-1,d} + ea[d - 1] = \sum_{s \in \mathcal{S}} req[s, d - 1].$$

Subtracting gives

$$\begin{aligned} sa[d] - ea[d - 1] &= \sum_{s \in \mathcal{S}} (req[s,d] - req[s, d - 1]) \\ &= sr[d] - er[d - 1] \end{aligned}$$

from Lemma 2.1. ■

Corollary 2.2. *If for every $d \in \mathcal{D}$, $sa^*[d] \geq sr[d]$ (S); then for every $d \in \mathcal{D}$, $ea^*[d] \geq er[d]$, i.e., the SSS has sufficient ending days (E).*

Proof. From Lemma 2.2, $sa^*[d] - sr[d] = ea^*[d - 1] - er[d - 1]$, since after deleting workstretches starting on day d of length $7k$ ($k \in \mathrm{IN}^+$), both $sa[d]$ and $ea[d - 1]$ decrease by the same amount, giving $sa^*[d]$ and $ea^*[d - 1]$, respectively. It now follows that if $sa^*[d] \geq sr[d]$ for all $d \in \mathcal{D}$, then $ea^*[d - 1] \geq er[d - 1]$ for all d
$\in \mathcal{D}$. ■

Example. To illustrate that satisfaction of the required number of starting or ending days in the SSS is *not sufficient* to obtain a CoSA, consider the following example: Suppose that we have an SSS in which workstretches are no shorter than 2 days in length and consider the requirements for a particular shift as given in Table VI.

TABLE VI. Requirements for a single shift

M	*T*	*W*	*Th*	*F*	*Sa*	*Su*
12	8	4	4	5	7	0

Even if an adequate number of workstretches are provided in the SSS with the required number of starting and ending days, a continuous assignment is not possible in this case. Observe that we need four workstretches ending on Monday for this shift type. However, without workstretches of length 1 day, this is impossible in a continuous assignment, since each such workstretch would contain a superfluous Sunday. We note that the detection of impossible shift strings can be used as a screening test for eliminating problems that cannot provide a continuous solution. This will be further discussed in Section 4.

As described in Theorem 2.2, a continuous assignment will contain a combination of noncyclic and cyclic chains in general. Once noncyclic chains have been provided, for a particular shift type, the remaining requirements will be equal, since the satisfaction of starting and ending requirements will have removed any differences that existed between adjacent shift requirements.

Definition 2.6. *After assigning shifts to days in the SSS, the remaining requirements* $rem[s, d]$ *are equal to* $req[s, d]$ *less the number of times shift s is assigned to day d. The remaining requirements are said to be equalized for shift type s if* $rem[s, d_1] = rem[s, d_2]$ *for all* $d_1, d_2 \in \mathcal{D}$. *If workstretches in the SSS exist that allow every shift type in the SRM to be equalized, the SRM is said to be equalizable.*

The following corollary can now be stated.

Corollary 2.3. *A necessary condition for the existence of a continuous assignment of shifts from an SRM to an SSS is that the SRM is equalizable.*

Proof. This result is now an immediate consequence of ($\Rightarrow$) in Theorem 2.2. ■

The equalization of the SRM is shown to be a necessary condition for CoSA. In addition, it is shown that sufficient starting (or ending) days are required to achieve this equalization. The strategy to establish equalization and the test for sufficient starting (or ending) days will be incorporated in network algorithms discussed in Section 3.

2.3. Conditions for the Existence of FoRSA

Corollary 2.1 shows that when insufficient starting days are available in the SSS then a CoSA cannot be obtained. In this situation, it is hoped to achieve an assignment of shifts that is as "continuous" as possible, but which must necessarily contain some rotations. As explained earlier, *forward rotations* are desirable for several reasons, and we will now examine the conditions under which such solutions can be obtained.

Example. To gain a better understanding of the requirements for forward rotation, consider the situation involving two shift-types on adjacent days $d - 1$ and d (Table VII). As has been our usual convention, the shift types are listed in increasing order of starting times. Starting with $s1$, we note that assigning these shifts on consecutive days $d - 1$, d in the SSS leaves one $s1$ on day d that would require the start of a workstretch on that day in order to avoid backward rotation. This illustrates the underlying principle regarding a necessary condition for FoRSA.

Theorem 2.3. *FoRSA* $\Rightarrow$ *for all* $r \in \mathcal{S}$, $d \in \mathcal{D}$,

$$\sum_{s\in\mathcal{S}, s\le r} req[s, d] \le \sum_{s\in\mathcal{S}, s\le r} req[s, d-1] + sa[d],$$

where $s \le r$ *indicates that shift s has a starting time that is not later than that of shift r.*

Proof. Assume that we have a forward rotating shift assignment in which for some r and d, $\sum_{s\le r} req[s, d] > \sum_{s\le r} req[s, d-1] + sa[d]$.

Then, $\sum_{s\le r} req[s, d] - (\sum_{s\le r} req[s, d-1] + sa[d])$ surplus shifts of type $s \le r$ cannot be either placed at the beginning of a workstretch commencing on day d or paired with shifts on day $d - 1$ of type $t < s \le r$. These surplus shifts on day d must therefore be paired with shift types on day $d - 1$ that have a later starting time. Thus, the assignment cannot be forward rotating. ■

We note that an *equivalent* result involving availability of ending days on day $d - 1$ can be derived.

Corollary 2.4. *FoRSA* $\Rightarrow$ *for all* $r \in \mathcal{S}$, $d \in \mathcal{D}$, $\sum_{s\in\mathcal{S}, s>r} req[s, d-1] \le \sum_{s\in\mathcal{S}, s>r} req[s, d] + ea[d-1]$.

Proof. From Theorem 2.3, we have, for all $r \in \mathcal{S}$,

$$\sum_{s\le r} (req[s, d] - req[s, d-1]) \le sa[d],$$

and from Lemma 2.2 since $sa[d] = ea[d-1] + (sr[d] - er[d-1])$, then

$$\sum_{s\le r} (req[s, d] - req[s, d-1])$$
$$\le ea[d-1] + \sum_{s\in\mathcal{S}} (req[s, d] - req[s, d-1])$$

TABLE VII. Two shift requirements on adjacent days

Shift Type	. . . . $d-1$	d . . . .
$s1$	3	4
$s2$	4	3

from Lemma 2.1. The result follows by a rearrangement of the sums. ■

The backward rotation shown in Table III is an example of one that cannot be removed, since the requirements of Theorem 2.3 are not satisfied. Theorem 2.3 can be used to determine a lower bound on the number of backward rotations that can occur in an assignment should this condition be violated. This is calculated by accumulating surplus shifts, as described in the proof of the theorem, over all shift types and adjacent pairs of days in the SRM. This lower bound will be used in the network solution method discussed in Section 3.2.

3. NETWORK MODELS FOR SHIFT ASSIGNMENT

3.1. Continuous Shift Assignment Network

We will initially consider a network model that searches for a continuous assignment. In Section 3.2, this model will be extended and incorporated in a more general network that seeks to determine rotating solutions when a CoSA cannot be found. Since much of the strategy used in the more general model is based on techniques employed for CoSA, it is instructive to examine the CoSA model first. Initially, the SSS and SRM are tested to ensure that sufficient starting days exist, since Corollary 2.1 specifies this as a necessary condition for a CoSA solution. Once this test is passed, a three-step algorithm for seeking a CoSA solution is adopted. A general outline of this algorithm is as follows:

Algorithm Search CoSA

1. **(Equalization)** Attempt to equalize the SRM by satisfying starting and ending requirements using workstretches in the SSS. If an equalization can be found for each shift type, assign shifts to the workstretches used to equalize and adjust the SRM and SSS; Else STOP.
2. **(Cycle Searching)** For the remaining workstretches in the SSS, find cyclic chains of workstretches. Each cyclic chain will contain a multiple of 7 days.
3. **(Cycle Assignment)** For each shift type whose equalized remaining requirements are greater than 0, assign cycles found in Cycle Searching to the remaining shift types.

This algorithm describes a decomposition approach to seeking CoSA solutions, in which Equalization is attempted for each shift type at a time. Since neither the equalization of the SRM nor the cycles found in the remaining SSS are necessarily unique, it may be necessary to use this strategy repetitively before a solution is found. Networks are used to implement Equalization and Cycle Searching, whereas a simple greedy heuristic is used to find Cycle Assignments.

The network used to carry out *Equalization* is shown in Figure 1. There are three node sets comprising 15 nodes in total. First, a set of seven nodes representing possible workstretch starting days; second, a set of seven nodes representing possible workstretch ending days; and, finally, a source/sink node.

There are four arc sets comprising a total of $21 + tws$ arcs altogether, where tws is the total number of workstretches in the SSS. Each workstretch in the SSS is represented by an arc drawn from its starting day node to its ending day node, having a lower bound $l = 0$, an upper bound $u = 1$, and a cost equal to the length of the workstretch. Figure 1 shows an example of a workstretch $T \rightarrow Su$. A set of seven arcs is drawn from the source/sink node to each starting day node and starting requirements are imposed by setting $l = u = sr[s, d]$ for these arcs. Likewise, a set of seven arcs is drawn from each ending day node to the source/sink node. Ending requirements are imposed by setting $l = u = er[s, d]$ on these arcs. For both sets of arcs, the cost $c = 0$. Finally, there is a set of backward arcs for each ending day $d - 1$ to a starting day d. These are required to allow circulation of flow through the network and to encourage the creation of cyclic chains linking up workstretches that end on day $d - 1$ with workstretches that start on day d.

For Equalization, a network is constructed for each shift type s and the Out of Kilter algorithm is used to seek a minimum-cost circulation. Since the only arcs with nonzero costs are those representing workstretches, a minimum cost solution is one containing as few days as

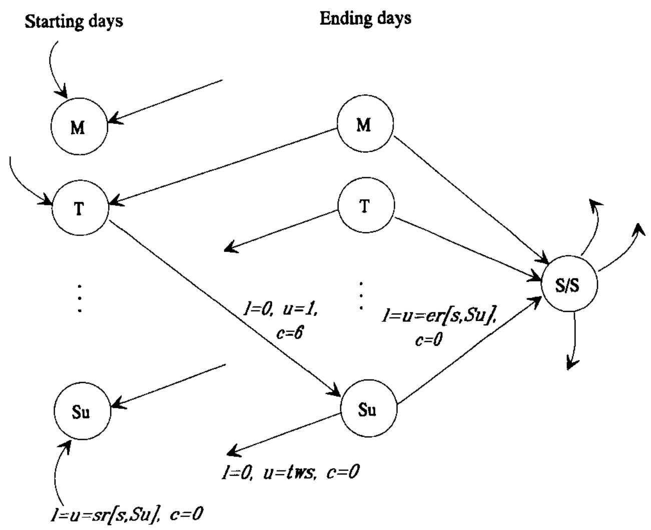

Fig. 1. Network for equalization of shifts showing lower and upper bounds and costs for each arc set.

possible in order to satisfy starting and ending requirements. Three outcomes are possible: First, a solution is found where the remaining shift requirements are $rem[s, d] = k \geq 0$ for every $d \in \mathcal{D}$, in which case equalization has been achieved. Second, a solution is possible, but too many days are used, i.e., $rem[s, d] = k < 0$ for every $d \in \mathcal{D}$. Third, no feasible solution can be found, in which case no equalization is achieved for shift type s. When equalization is achieved for a shift, then workstretches that have been used in the equalization are removed from the network when the next shift type is considered. This process is repeated for each shift type. If for some shift type equalization is not achieved, either this shift type is moved to the top of the list, the network is reinitialized, and the process is started afresh or, for the case where this shift has previously failed to equalize, the equalization process is terminated. Table VIII illustrates the order of processing should two shift types fail to equalize initially. Should either $s2$ or $s4$ fail to equalize again at iteration 3, then equalization of the SRM is terminated. At the conclusion of Equalization for all shift types, we have $rem[s, d] = k_s \geq 0$ for every $d \in \mathcal{D}$ and $s \in \mathcal{S}$ and the SSS is adjusted by removal of all workstretches used in the equalization process.

Cycle Searching involves a process of searching for cyclic chains in the remaining SSS. Note that Theorem 2.2 guarantees that cycles can be found in the remaining SSS to which the remaining shifts can be assigned. A shortest path network is used to find these cycles. The network used is the same as described in Figure 1, but without the source/sink node and its associated arcs. In addition, arcs associated with workstretches used in the equalization process have been removed. As before, the costs on the backward arcs are all zero, so that contributions to the length of paths found in this network come only from the workstretches. Dijkstra's shortest path algorithm is used to find cyclic chains, commencing at the first starting day where a remaining workstretch begins. For example, if an $M \twoheadrightarrow W$ occurs, we commence at the starting day node for Monday and seek the shortest path from this node to the ending day node for Sunday. If a $Th \twoheadrightarrow Su$ workstretch is available, then we have a cyclic chain $M \twoheadrightarrow W$; $Th \twoheadrightarrow Su$ of length 7. Workstretches are removed from the network as cycles are found. When all workstretches commencing on Monday are removed, we move to Tuesday and so on, until all workstretches are concatenated into cycles.

In *Cycle Assignment*, remaining shifts must be matched with suitable cycles to complete the assignment process. We are able to solve this task effectively using a simple greedy heuristic since the numbers involved are small. Examination of all cases for remaining shifts less than six showed that a cycle assignment can always be found by assigning the largest cycle to the largest remaining shift requirement. In the event that remaining shifts are greater

TABLE VIII. Order of processing for shift equalization; A "*" indicates failure to equalize

Processing Order	Iteration Number	1	2	3
1		$s1$	$s2$	$s4$
2		$s2$*	$s1$	$s2$
3			$s3$	$s1$
4			$s4$*	$s3$
5				$s5$

than or equal to six, cycles are found using a simple exhaustive search routine until remaining shifts fall below six, whereupon the greedy routine is used. Whereas the total length of cycles must equal the total of all remaining shift requirements, a given set of cycles may not match the required remaining shifts. For example, if after equalization the remaining requirements for four shifts are (1, 0, 2, 1), while a set of cycles consists of (2, 2), then clearly shift types $s1$ and $s4$ cannot be satisfied. In this situation, alternative cycles (if any) are generated in Cycle Searching. Should all possible cycle sets fail to allow the completion of the cycle assignment, we return to Equalization and seek alternative equalizations.

3.2. Rotating Shift Assignment Network

Shift assignment problems for which the CoSA algorithm fails to find a solution are candidates for forward rotating shift assignments (FoRSA). Although we can no longer obtain solutions with every workstretch associated with a single shift type, it is desirable to seek solutions with as few rotations as possible. Since forward rotating solutions must necessarily contain workstretches in which more than one shift type occurs, it is natural to consider portions of workstretches rather than complete ones, within which assignments are continuous. It will then be useful to define the following terms:

Definition 3.1. *A partial workstretch is a subset of contiguous days taken from a workstretch. A front partial workstretch is one that contains the starting day for the original workstretch, whereas a back partial workstretch is one that contains the ending day for the original workstretch.*

Example. For the workstretch $M \twoheadrightarrow F$, $T \twoheadrightarrow Th$ is a partial workstretch; $M \twoheadrightarrow W$, a front partial workstretch; and $Th \twoheadrightarrow F$, a back partial workstretch.

A network that is capable of producing forward rotating shift assignments must necessarily contain partial workstretches. The extended network shown in Figure 2 can

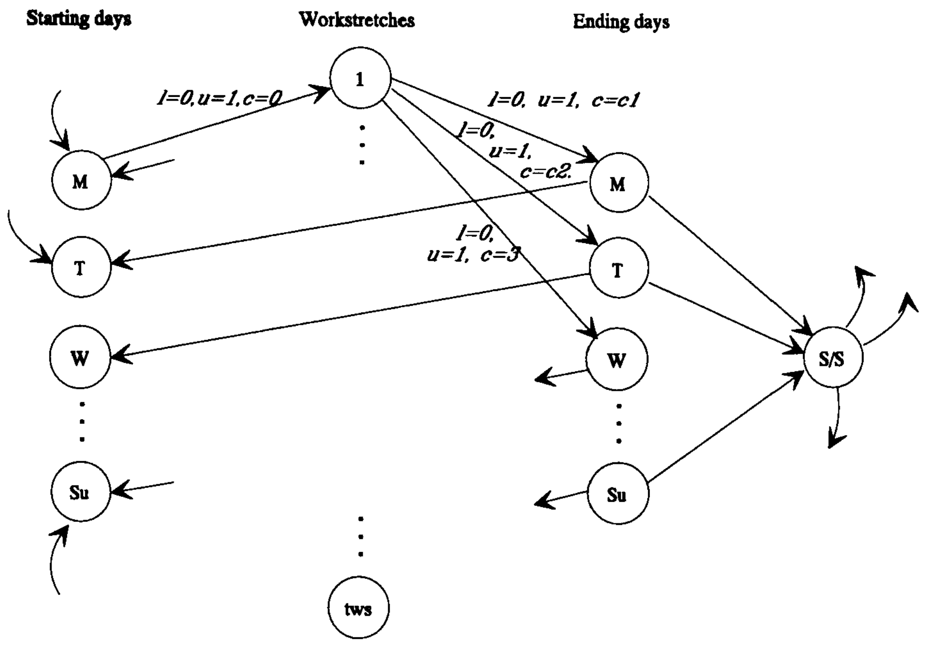

Fig. 2. The extended network for FoRSA.

be used for this purpose. In addition to the node sets described in the original network of Figure 1, we have a node for each workstretch in the SSS, making a total of 15 + *tws* in all. The original workstretches and a set of front partial workstretches are incorporated into the network as follows: The common starting day for both an original workstretch and its front partials is represented by an arc drawn from a starting day node to the workstretch node. Bounds and costs on these arcs are given by $l = 0$, $u = 1$, and $c = 0$. Workstretches and their front partials are represented by arcs drawn from the workstretch node to the corresponding ending days. Figure 2 shows a $M \rightarrow W$ workstretch and its front partials $M \rightarrow T$ and $M \rightarrow M$. Bounds on these arcs are set at $l = 0$, $u = 1$, whereas arc costs $c1$ and $c2$ are modified in the course of finding solutions and will be discussed shortly.

The general strategy used is as described for CoSA, namely, a shift equalization procedure, but this time involving an additional set of partial workstretches. The original workstretch arcs are given costs c = "workstretch length," whereas the costs on the partial workstretch arcs are set equal to their length plus a fixed high penalty. In this way, we seek equalizations that discourage the use of partial workstretches. For problems in which starting and ending day shortages occur, assignments of front partial workstretches expose starting days, thus facilitating the equalization process.

Backward rotations may arise as a result of employing partial workstretches. In Equalization, if a front partial workstretch is used to equalize a certain shift type, this will expose a corresponding back partial workstretch that may be used to equalize another shift type. In this situation, it is important to ensure that forward rotations occur at this interface, if possible, or at least that the number of backward rotations obtained are as few as possible. A similar problem arises when workstretches used to create cyclic chains are used to satisfy remaining shift requirements, in Cycle Assignment, since, in general, these cyclic chains will contain some back partial workstretches that must be interfaced with front partial workstretches already used in the equalization process. It should be noted that Theorem 2.3 does not guarantee FoRSA in the event that no backward rotations are determined, since this is merely a *necessary* condition.

In a small percentage of cases, all Equalization/Cycle Assignment alternatives fail to give an assignment. In this event, cycles within the current set of cycles are split in order to complete the solution. Thus, for example in the case where only (2, 2) cycles can be found for remaining shift requirements of (1, 0, 2, 1), one of the 2 cycles is split to give two 1 cycles. Since splitting a cycle necessarily divides a workstretch in two, this will result in additional forward rotations. Cycle splitting is performed so as to avoid additional backward rotations where possible.

4. RESULTS

To test the effectiveness of the CoSA network, a set of experiments was devised in which for *given* single-shift schedules continuous solutions were created. This was done by randomly assigning a shift type continuously to each workstretch in the SSS. The corresponding SRM was then *derived* from the continuous assignment thus

TABLE IX. Test problems used in experiments, showing the size of the workforce, and the no. shift types; the last column shows the percentage of successful continuous assignments when CoSA solutions are known to exist, for 1000 randomly constructed continuous SRM/SSS combinations

Test Problem No.	Size of Workforce	No. of Shift Types	% of Continuous Assignments
1	28	4	99.8
2	25	4	100.0
3	37	3	100.0
4	40	6	98.5

constructed, giving an SSS/SRM combination for which it was *known* that a continuous solution could be found. Four test problem sets were generated using this technique. The basis for each test problem set was an SSS of given length and a specified number of shifts required in the assignment. Table IX displays the workforce size and number of shift types in each case. For each of these four SSS/shift number combinations, 1000 randomly constructed continuous SSS/SRM problems were generated and presented to the CoSA network. Results are shown in Table IX, demonstrating a high average success rate of 99.6%.

Two strategies were adopted to improve the efficiency of the network algorithms. First, all problems were screened to determine whether the SSS had sufficient starting days to satisfy the starting requirements in the SRM, and, in addition, all problems had their SRMs tested for impossible shift strings. The satisfaction of this condition is dependent on the length of workstretches allowed in the solution. Any problem for which there were both sufficient starting days and no impossible shift strings were passed to the CoSA network.

Second, the *smoothness* of the SRM was found to be a major factor in determining the speed of solution or the tendency for the heuristic to fail at Equalization or Cycle Assignment. For this reason, an initial smoothing routine was incorporated prior to using the extended network for finding a FoRSA solution. In this context, smoothness is a measure of the difference between shift requirements on adjacent days. In the ideal case when requirements on adjacent days are identical for a particular shift type, no equalization at all is required for that shift type. When smoothing the SRM, the total number of shifts for each day must be left unchanged in order to satisfy the SSS, since this specifies a fixed total work requirement for each day. The requirements for each *shift type* on a particular day, however, were changed so that differences between requirements on adjacent days for each shift were as small as possible. The smoothed problem was then passed through the extended network, after which assignments

TABLE X. A comparison of times and failures based on 100 randomized versions of test problem 4[a]

Test Problem 4	Average Time (sec)	Max Time Over 100 Cases (sec)	Failures to Equalize	Failures to Assign
Unsmoothed	5.35	88.43	3	1
Smoothed	0.32	1.92	—	—

[a] In each randomized version, a variation of the SRM was created. The same random problems were used for the smoothed and unsmoothed cases.

were reconstituted to conform to the *original* shift requirements. Table X displays a comparison of results obtained for test problem 4. As well as the removal of a small number of failures to either equalize or assign cycles, these results display a dramatic improvement in time, and for this reason, the smoothing strategy was adopted in all cases prior to using the extended network.

Results are now presented for the same four test problem sets defined in Table IX, using the strategies discussed above. In each problem set, a fixed SSS was used and 1000 randomly generated SRMs were created by varying the original shift distributions on each day. Test problems were generated on an IBM 486DX 66 Mhz machine and are shown in Table XI. Note that the proportion of either continuous (C), forward rotating (F), or backward rotating (B) solutions is intrinsic to the particular problem considered. There were no cases where a problem either failed to equalize or assign cycles.

5. CONCLUSIONS

This paper discussed conditions for the existence of continuous (CoSA) and forward rotating (FoRSA) shift assignments. Necessary and sufficient conditions for the existence of CoSA have been derived. However, the necessary and sufficient conditions do not provide an easily recognizable set of properties required of the SSS/SRM that will guarantee CoSA solutions. Nevertheless, we can readily deduce from this result a necessary condition that

TABLE XI. Results obtained for 1000 instances of the four test problem sets, showing the average time taken to solve, the maximum time in each case, and the type of solution found

Test problem	Average Time (sec)	Max Time 1000 Cases (sec)	C	F	B
1	0.039	0.06	333	301	366
2	0.069	0.27	30	917	53
3	0.057	0.11	757	243	—
4	0.455	26.75	278	471	251

can be used as a test for the existence of CoSA solutions before executing the network algorithm.

In addition, necessary conditions for the existence of FoRSA solutions are derived. These conditions are readily implemented as a test for the existence of FoRSA before executing the network algorithm and are adapted in the algorithm to provide a lower bound on the number of backward rotations when FoRSA solutions cannot occur.

The network algorithms either seek CoSA solutions when sufficient starting days are available and impossible shift strings do not occur or seek FoRSA solutions in the more general case. When CoSA solutions are known to exist, the CoSA network has been demonstrated to provide such solutions in a high proportion of cases. The more general extended network, in association with smoothing prior to its application, has been demonstrated to produce assignments efficiently in all cases.

We would like to thank Geoff Jarrad for his thoughtful contribution to the coding and development of the network algorithms.

REFERENCES

[1] N. Balakrishnan and R. T. Wong, A network model for the rotating workforce scheduling problem. *Networks* **20** (1990) 25–42.

[2] L. D. Bodin, Towards a general model for manpower scheduling: Parts 1 and 2. *J. Urban Anal.* (1973) 191–245.

[3] C. A. Czeiler, M. C. Moore-Ede, and R. M. Coleman, Report—Rotating shift work schedules that disrupt sleep are improved by applying circadian principles. *Science* **217** (1982) 460–463.

[4] N. B. Heller, J. T. McEwen, and W. W. Stenzel, Computerized scheduling of police manpower. *St. Louis Police Department,* St. Louis, MO (1973).

[5] N. B. Heller et al., *Work Schedule Design Handbook: Methods for Assigning Employees' Work Shifts and Days Off.* U.S. Department of Housing and Urban Development, Washington, DC (1978).

[6] D. M. Panton, On the creation of multiple shift continuous operation rosters under general workforce conditions. *APJOR* **8** (1991) 189–201.

[7] J. M. Tien and A. Kamiyama, On manpower scheduling algorithms. *SIAM Rev.* **24** (1982) 275–287.

Received February 22, 1993
Accepted February 23, 1994