M&SOM MANUFACTURING & SERVICE OPERATIONS MANAGEMENT
Vol. 10, No. 3, Summer 2008, pp. 411–420
ISSN 1523-4614 | EISSN 1526-5498 | 08 | 1003 | 0411

informs®
DOI 10.1287/msom.1070.0172

Simple Methods for Shift Scheduling in Multiskill Call Centers

Sandjai Bhulai, Ger Koole, Auke Pot
Department of Mathematics, Vrije Universiteit, 1081HV Amsterdam, The Netherlands
{sbhulai@few.vu.nl, koole@few.vu.nl, sapot@few.vu.nl}

This paper introduces a new method for shift scheduling in multiskill call centers. The method consists of two steps. First, staffing levels are determined, and next, in the second step, the outcomes are used as input for the scheduling problem. The scheduling problem relies on a linear programming model that is easy to implement and has short computation times, i.e., a fraction of a second. Therefore, it is useful for different purposes and it can be part of an iterative procedure: for example, one that combines shifts into rosters.

Key words: contact centers; multiskill call centers; shift scheduling; skill-based routing; staffing; workforce management
History: Received: December 1, 2005; accepted: March 1, 2007. Published online in *Articles in Advance* January 4, 2008.

1. Introduction

This paper deals with the allocation of labor resources over time, which is an integral part of workforce management (WFM). Labor allocation is typically an operational problem with a time horizon of only a few weeks. It is common to distinguish four phases in the process of labor allocation:

(1) workload prediction,
(2) staffing,
(3) shift scheduling, and
(4) rostering.

Workload prediction is concerned with the prediction of the future amount of work offered to the call center. Staffing translates this amount of work in numbers of required agents such that a prespecified service level is met. Shift scheduling then generates shifts such that these staffing levels are met. Finally, rostering refers to the pairing of shifts into rosters and the assignment of employees to the rosters.

It is important to find a good match between the predicted workload and the scheduled workforce. An inadequately sized workforce can lead to low service levels, such as long waiting times. This can be avoided by scheduling a sufficiently large number of employees. However, it is undesirable to schedule too many employees because, besides service levels, contact centers also have to meet economical objectives, in particular, minimizing costs because of employee salaries. Minimizing the number of employees is an important subject because labor is expensive; approximately 80% of operating costs in call centers are because of personnel; see Gans et al. (2003). Therefore, the cost reductions obtained with good scheduling algorithms can be substantial.

Optimal labor allocation in single-skill call centers is a complex issue, and the integration of the four phases described above results in intractable models. Multiskill call centers come with additional complexity, because agents need to handle jobs that require different skills. Regarding labor allocation, the predicted workload is often specified per job/skill type in a multiskill setting. Hence, to determine optimal staffing levels is much more complicated as compared to single-skill call centers where the workload is specified by a single number.

1.1. Contribution

In this paper, we deal with Phase 3 of the labor allocation process: shift scheduling. Our main contribution is that we develop a method to determine schedules in multiskill call centers such that a rough match between the predicted workload and labor capacity is realized, taking the randomness of the arrival process into account. Our method iterates between Phases 2 and 3 of the labor allocation process. The incentive to solve these steps separately is computational, because an integrative approach yields calculations that are very time consuming to execute. In practice, obtaining good rosters often requires several iterations between the different phases. In these cases, it is important to have a scheduling and rostering method with short computation times. The possible drawbacks of solving both steps separately are discussed in §4.

To solve the shift-scheduling problem of Phase 3, we develop a model that generates a set of feasible solutions such that integer programming techniques can be used to obtain the optimal shifts. Feasible solutions are generated by using a fast and accurate heuristic that solves the Phase 2 staffing problem (Pot et al. 2008). The integer programming model then encapsulates the flexibility of multiskilled agents to work in different groups that may use various subsets of their skills in different periods of the day. Both phases have small computational requirements such that rostering (Phase 4) can be performed much more quickly than is currently possible using methods from the literature (Cezik and L'Ecuyer 2008).

1.2. Literature

The literature offers different models and algorithms for shift scheduling in single-skill call centers. However, not much literature is devoted to scheduling in multiskill call centers. The most relevant papers on scheduling in call centers are discussed next.

Most models that deal with shift scheduling in a multiperiod and single-skill environment are based on the standard set covering model presented in Dantzig (1954). The model of Dantzig finds an optimal set of shifts, while obeying the service-level constraint in each period. A cost is associated with each shift and the objective is to select the shifts that minimize the total costs.

Keith (1979) extended the set covering model with slack and surplus variables. His model allows for deviations from the predicted staffing levels to be penalized by costs. This creates a balance between the costs of deviating from the staffing levels and the reduction in the number of scheduled shifts, while satisfying the service level.

Thompson (1997) introduces two models for shift scheduling. He distinguishes between minimum acceptable service levels per period and a constraint on the average service level over the planning horizon. An integer programming model is described that includes both types of service-level constraints. It solves the staffing problem and the shift-scheduling problem in an integrated fashion. Thompson (1997) also gives an extensive overview of the literature on scheduling and makes a classification of the different shift-scheduling models. In §1 of the appendix, available online, [1] we give a short description of one of the models that can be used to obtain lower bounds for more complex models.

Ingolfsson and Cabral (2007) focus on cases in which the planning intervals are not assumed to be independent. Most staffing methods perform badly in this case because of transient effects between intervals. This is typically the case when long service times are present, because they create dependency between consecutive periods. In addition, the paper introduces a method for staffing and scheduling in single-skill call centers.

In Atlason et al. (2004), the model of Thompson (1997) is adjusted for cases in which the service level is only obtainable via simulation. The benefit of simulation is that it allows for the calculation of the service level in a transient setting. Because simulation is a very time-consuming operation, the method deals with the conditions on the service level differently and more efficiently by means of cutting-plane techniques; see Gomory (1958).

We next discuss two methods from the literature that are used in this paper. Cezik and L'Ecuyer (2008) describe a generalization of the method from Atlason et al. (2004) in the context of multiskill call centers. The method reduces the solution space by means of cutting-plane methods that were developed to solve large-scale linear integer programs. The computation time of this algorithm is relatively long because each cut requires the multiskill call center to be simulated multiple times and very accurately. Hence, they are not able to solve the shift-scheduling problem, but only are able to determine the staffing levels that are constant over the day. For this purpose, it is used only in Step 1 in this paper. Note that the computation time is important because Phase 4 (rostering)

[1] An online appendix to this paper is available on the *Manufacturing & Service Operations Management* website (http://msom.pubs.informs.org/ecompanion.html).

is usually an iterative procedure in which Phase 3 is executed several times with small adjustments. Thus, for practical purposes, it is desirable to have an algorithm that executes Phases 2 and 3 relatively quickly. Pot et al. (2008) solve the same problem by means of a Lagrangean relaxation.

1.3. Outline

This paper is organized as follows. The main contribution of the paper is in §2, presenting an efficient method for shift scheduling in a multiskill environment when considering a service-level constraint in each planning period. The method consists of two steps: (1) methods for the determination of staffing levels, discussed in §2.1, and (2) the determination of an optimal set of shifts, which is the subject of §2.2. The new methods for scheduling in multiskill call centers are numerically evaluated by a case study in §3. We show that the method yields nearly optimal results. Finally, a summary of the results is given in §4, which also discusses directions for future research. Additional numerical examples can be found in the appendix, which is online.

2. Multiskill Environment

This section introduces methods for shift scheduling in multiskill call centers for two types of service-level constraints. The methods consist of Phases 2 and 3 of the labor allocation process. The first method executes both steps separately. Because this method cannot deal with service-level conditions that are specified as an average over the day, the second method describes a heuristic that iterates between both steps.

The major difference with a single-skill environment is the presence of multiple agent groups with different skills. We assume that agents from the same group have an equal set of skills. Our objective is to meet the service-level constraint against minimal costs.

In the first step, a minimal staffing level is determined such that the service-level constraints are satisfied, i.e., the fraction of calls (over all types) that have a waiting time of less than 20 seconds (the AWT) is greater than or equal to α. The staffing levels denote the required number of agents in each agent group for each period. This scheduling problem is significantly more difficult in comparison to scheduling in single-skill call centers. We solve this difficult problem by using the heuristic developed in Pot et al. (2008). We discuss the heuristic in §2.1.

In the second step, a set of shifts has to be composed that minimizes the costs and satisfies the required staffing levels. This step is also more complex than in a single-skill environment. In a multiskill environment, an agent with a specific set of skills can be assigned to different agent groups with potentially fewer skills in each period. Modeling this in a straightforward way leads to many decision variables, which easily results in intractable models.

Before presenting the two methods, we define the multiskill model as follows. We consider a call center that handles calls that require a skill from the set $\mathcal{M} := \{1, 2, \ldots, M\}$. Calls of type $m \in \mathcal{M}$ arrive in period $t \in \mathcal{T} = \{1, 2, \ldots, T\}$, according to a Poisson process with rate $\lambda_{m,t}$. Moreover, we assume that the arrival rate is constant in each period. Every agent in the call center belongs to an agent group, which can be different in each period, from the set $\mathcal{G} = \{1, 2, \ldots, G\}$. The service times are assumed to be exponential with rates that are skill and group dependent, denoted by rate $\mu_{m,g}$ for skill $m \in \mathcal{M}$ and group $g \in \mathcal{G}$. We assume that a control policy π is given that defines a call selection and agent selection rule. Call assignment occurs according to the agent selection rule. If a call is not assigned to an agent group, it is queued, after which it is served according to the call selection rule.

A shift is defined by a subset of the working hours from the set $\mathcal{T}$ and a subset of skills from $\mathcal{M}$. The number of shift types, i.e., the number of different shifts, is fixed and denoted by K. Each shift type has an index, and the corresponding indices are enclosed in the set $\mathcal{K} = \{1, 2, \ldots, K\}$. Each shift has an offset, which is denoted as the index of the starting period and a length. However, additional characteristics, e.g., breaks and split shifts, are also easy to include.

Let S_g be the set of skills of group g. We assume that for each shift there is a group of agents that has exactly the skills to work that shift. Hence, for notational convenience, we can denote the skill set of shift k with f_k, i.e., the index of the corresponding agent group for shift k. In this context, a shift k is workable if there is a group g such that $f_k = g$, and agents who work shift k can work in all groups g' that satisfy $S_{f_k} \subseteq S_{g'}$. To meet the service-level constraints, we suppose that for every agent group there is a set of workable shifts such that for some agent configuration the requirements are met. The cost of shift k is denoted as c_k, and the working hours are defined by $a_{k,t}$:

$$a_{k,t} = \begin{cases} 1, & \text{if an agent assigned to shift } k \text{ works during period } t \\ 0, & \text{otherwise.} \end{cases}$$

2.1. Step 1: Staffing Levels

In this part, we describe methods to compute the staffing levels of the agent groups for each interval of the day. To this end, we consider two existing methods from the literature that are described in Cezik and L'Ecuyer (2008) and Pot et al. (2008). For a summary of these papers, we refer to §1.

Both methods require several input parameters. The main parameters are the arrival rates $\lambda_{m,t}$, the service rates $\mu_{m,g}$, the routing policy, and the staffing costs as a function of the group sizes $K^g(s_g)$. The arrival rates can be specified for each job type in each interval. The service rates and staffing costs need to be specified for each agent group, at each point in time. We let the class of routing policies be limited to priority routing policies. See, for example, Franx et al. (2006) for an explanation.

Staffing costs require additional attention because these are not always directly available in call centers. The reason is that an agent can sometimes work in an agent group requiring a subset of his or her skills. Hence, the staffing costs depend on the costs of the shifts. To this end, we suggest deriving staffing costs from the costs of the shifts in the following way:

$$K^g(s_g) := \frac{s_g}{|\{k \in \mathcal{K}\colon f_k = g\}|} \sum_{k \in \mathcal{K}\colon f_k = g} \frac{c_k}{a_k e},$$

with e the unity vector. It is the average cost of the possible shifts the agents from the group can work, normalized by the shift lengths.

Each of the methods proposed by Cezik and L'Ecuyer (2008) and Pot et al. (2008) has at least one advantage and one disadvantage, and the advantage of the one is the disadvantage of the other. A disadvantage of the first method is the longer computation times and the lower accuracy. A disadvantage of the second method is that the service-level constraints can not be specified per job type, but only as an average over all job types.

In our opinion, requiring service-level constraints to be uniform across call types is not a big restriction for the following reason. Schedules are most often generated at least a few weeks ahead of time based on predictions. As a result, call centers often have to reschedule during the day when the real workload deviates from the predictions. Thus, service levels often can be and need to be adjusted during operations.

In the numerical experiments of this paper, we decided to use the method of Pot et al. (2008), because we only consider service levels that are an average over all job types.

2.2. Step 2: Shift Scheduling

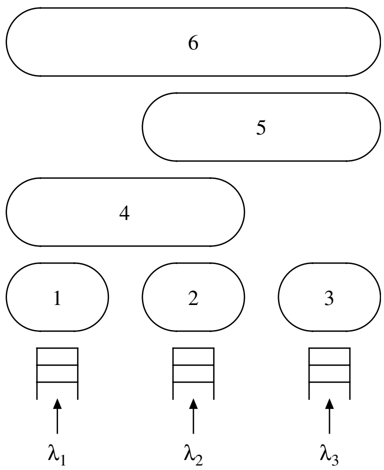
This section describes the second step of the two-step algorithm. A solution is found to the question of how to determine the optimal number of shifts of each type. We also answer the question of how to allocate agents to agent groups in each period.

The main feature of this method is that agents can work in different groups during the same shift. The skill set of the shift determines if an agent with a specific type of shift is allowed to work in a certain agent group. An agent with skill set X is allowed to work in a group with skills X' if $X' \subseteq X$. The objective of the integer programming model is to minimize personnel costs, while meeting the staffing requirements for each group in each period.

2.2.1. Introduction. We introduce the integer programming model by means of an example. Consider a call center with three skills $\mathcal{M} = \{1, 2, 3\}$ and six agent groups $S_1 = \{1\}$, $S_2 = \{2\}$, $S_3 = \{3\}$, $S_4 = \{1, 2\}$, $S_5 = \{2, 3\}$, and $S_6 = \{1, 2, 3\}$. Information about the arrival streams, control policy, and service time distributions is not relevant for shift scheduling. They are only needed to determine the required number of agents $s_{t,g}$ in Step 1. The example is depicted in Figure 1, showing the agent groups and the arrival streams.

We are interested in an integer programming model that determines the cheapest set of shifts such that requirements concerning minimum numbers of agents are met. To get more insight, we assume that

Figure 1 **Example of a Three-Skill Call Center**



the optimal values x_k and the number of agents working shift k are given. Then, the assignment of the available agent numbers x_k to the agent groups can be modeled as a linear assignment problem. This is depicted as a graph in Figure 2. The nodes on the left side represent the scheduled number of agents for each skill set, which is determined by the variables x_k. These numbers represent the sizes of the flows from the source. Note that the number of scheduled agents with skills $\mathcal{M}' \subseteq \mathcal{M}$ in period t is equal to

$$\sum_{k:\, S_g = \mathcal{M}',\, a_{k,t} > 0} x_k,$$

with $g \equiv f_k$. The nodes on the right side denote the required number of agents per agent group in period t. These represent the capacities of the arcs that are connected to the sink. The agents scheduled on the left side need to be assigned to the agent groups on the right. A feasible solution of the linear assignment problem gives a feasible assignment of the scheduled agents to the different agent groups. However, the assignment from Figure 2 of available agents to agent groups is not explicitly modeled in the integer program, because a reduction of decision

Figure 2 **Linear Assignment Problem**

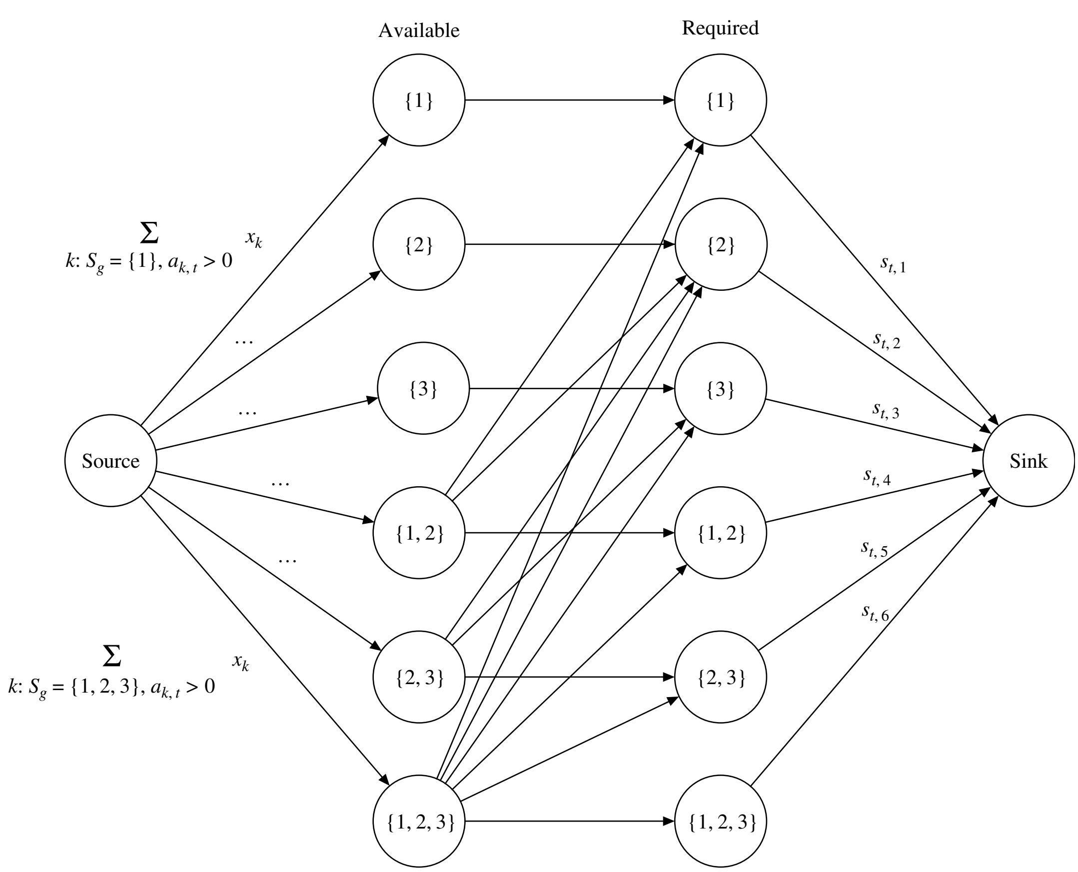
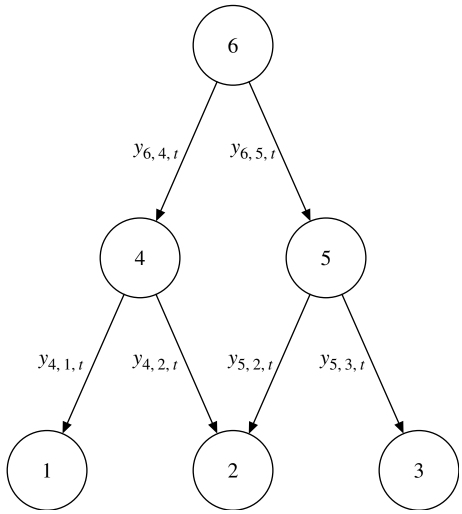


Figure 3 Simplified Assignment Model



variables is possible. The reduction is obtained by introducing dummy variables $y_{g',g,t}$ for each group g', g such that $S_g \subseteq S_{g'}$. The variable $y_{g',g,t}$ denotes the number of agents that are removed from group g' and work in group g, which has fewer skills in period t. Note that any subset $\mathscr{X}'$ of $\mathscr{X}$ can be obtained by removing elements successively, assuming that all types of agent groups are present in the call center. Therefore, the dummy variables $y_{g',g,t}$ make all feasible assignments possible. For example, an agent from Group 6 can operate as a specialist in Group 1 by setting the two dummy variables $y_{6,4,t}$ and $y_{4,1,t}$ to one. We can depict the dummy variables by arcs between groups that have one skill less, as shown in Figure 3. The introduction of the dummy variables leads to a significant reduction in decision variables. Suppose that we have a call center with groups having all possible combinations of skills. The linear assignment model has $\sum_{k=1}^{M}\binom{M}{k}(M^k-1)=3^M-2^M$ variables. The simplified model has $\sum_{k=2}^{M}\binom{M}{k}k = M(2^{M-1}-1)$ variables when $y_{g',g,t}$ has the additional constraint $|S_g|=|S_{g'}|-1$. If not all combinations of skills are represented by a group, then the number of decision variables can be reduced further, as we will explain in the following section.

Note that any subset $\mathscr{X}'$ of $\mathscr{X}$ can be obtained by removing elements successively only if all types of agent groups are present. However, call centers often have a limited number of groups in practice, which we also allow in our model formulation. Hence, we will choose the dummy variables more carefully in the integer programming model that we formulate in the next section.

2.2.2. Model. For the model, the necessary dummy variables are determined as follows. We consider each period t, $t \in \mathscr{T}$. For notational convenience we define the set $\mathscr{G}_t$ as a subset of the agent group indices that are required at time t. Group g is included in $\mathscr{G}_t$ if

- $s_{t,g} > 0$, or
- $a_{k,t} > 0$ for some $k \in \mathscr{K}$ and $f_k = g$.

Thus, it contains the indices of agent groups with a positive number of required agents or a potential positive number of scheduled agents (by having a shift with the same skills). Next, we define two sets of decision variables: $\mathscr{I}_{t,g}$ and $\mathscr{J}_{t,g}$. Set $\mathscr{I}_{t,g}$ contains the decision variables associated with agents moving from higher-level groups to agent group g, and set $\mathscr{J}_{t,g}$ contains the decision variables associated with agents moving from group g to lower-level groups. Variable $g' \in \mathscr{I}_{t,g}$ is included in the model if

- $g', g \in \mathscr{G}_t$,
- $S_g \subset S_{g'}$, and
- there exists no $g^* \in \mathscr{G}_t$ such that $S_{g'} \supset S_{g^*} \supset S_g$,

and variable $g' \in \mathscr{J}_{t,g}$ is included if

- $g', g \in \mathscr{G}_t$,
- $S_{g'} \subset S_g$, and
- there exists no $g^* \in \mathscr{G}_t$ such that $S_{g'} \subset S_{g^*} \subset S_g$.

Note that we require that no strict subset S_{g^*} exists between $S_{g'}$ and S_g. By using this notation, we can describe the integer programming model as

$$\begin{aligned}
\min\quad & \sum c_k x_k \\
\text{subject to}\quad & \sum_{k\in\mathscr{K}: f_k=g} a_{k,t}x_k + \sum_{g'\in\mathscr{I}_{t,g}} y_{g',g,t} - \sum_{g'\in\mathscr{J}_{t,g}} y_{g,g',t} \ge s_{t,g}, \\
& \qquad \forall t\in\mathscr{T}, g\in\mathscr{G}_t, \\
& x_k, y_{g',g,t} \ge 0 \text{ and integer}, \\
& \qquad \forall k\in\mathscr{K}, t\in\mathscr{T}, g, g'\in\mathscr{G}_t.
\end{aligned}$$

2.2.3. Shift Composition. Having obtained a solution (x, y), we can compose a schedule that specifies, for each shift, the agent groups in which the associated agent works during the different periods. This is done according to the following algorithm.

Shift Composition ( )
(1) Choose k such that $x_k \geq 1$. Set agent group $g = f_k$.
(2) For each period t with $a_{k,t} = 1$,
(3) Initialize $\bar{g} \leftarrow g$.
(4) Repeat:
(5) If variable $y_{g,g',t}$ exists and $y_{g,g',t} > 0$ for some g',
(6) $\bar{g} \leftarrow g'$ and $y_{g,g',t} \leftarrow y_{g,g',t} - 1$;
(7) else, stop and assign the shift to group $\bar{g}$ at time t.
(8) End for
(9) Decrease x_k by one and go to line 1 unless $x_k = 0$ for all k.

2.2.4. Note on Numerical Complexity. As we will show in §3.1, the problem in the cases of two, three, or five skills is numerically tractable. According to the literature, we can expect that this also holds for cases with a much larger number of different skills and many different types of shifts. The literature shows that set-covering problems are relatively easy to solve. There is a large number of papers available on crew scheduling on trains and airplanes. In particular, we would like to mention the shift-scheduling problems in which tasks are paired to shifts. Studies show that problems of more than 30,000 tasks are solved within reasonable time, e.g., hours, with shifts including breaks and many other features. In these problems, each task corresponds with a constraint, similar to a staffing level in our problem. The largest problems are solved close to optimality using column generation in conjunction with a Lagrangean relaxation. See, for example, Caprara et al. (1999), which also is applicable to our integer programming problem.

2.2.5. Note on Suboptimality. A possible drawback of our determining staffing levels and generating shifts separately, in two steps, is suboptimality. This is, to some extent, prevented by the condition from §2.2, i.e., there should be at least one shift available for each group. However, one should be careful in certain cases.

When there are many different agent groups with only a few skills and all shifts require only a small number of skills, the algorithm schedules more shifts than necessary, leaving agents with a lot of idle time. Perhaps the idle time could be reduced by choosing the agent groups more carefully. However, we expect that this situation is not likely to occur in practice. First, in many call centers, the number of different skills is limited, or the dependency between certain skills is low, as if there are several smaller multiskill call centers. Second, our experience is that if there are at least some agent groups with more than two skills, the results of the algorithm are nearly optimal. The reason is that solutions obtained by the algorithm prescribe in realistic cases the usage of relatively many specialists, because specialists are cheaper and work faster, and solutions often require relatively few cross-trained agents with two skills and hardly any agents with more than two skills. By including some agent groups with more than two skills, the time that agents are idle is expected to be low. A disadvantage is that solutions can require more agents with additional skills than an optimal solution would require. This is undesirable if agents with more skills are significantly more expensive. However, call centers often prefer a sufficient flexibility of agents in case the actual workload deviates from the predictions such that agents can be rescheduled. Then, it is desirable to have agents with additional skills available. Indeed, call centers often have a sufficient number of agents with more than two skills.

3. Numerical Experiments

In this section, we discuss a realistic example.

The example considers infinite waiting queues, customers having infinite patience for service, and service according to a first-in-first-out service discipline per job type. At their arrival, calls are assigned to employees according to overflow policies—see, for example, Franx et al. (2006)—and in such a way that specialists have the highest priority, agents with two skills the second-highest priority, and agents with three skills the third-highest priority. At a service completion, the job that arrived earliest is served among the queues with jobs for which the agent has the right skill.

Additional examples, with three and five skills, can be found in the appendix, available online.

3.1. Case Study

This study is based on the statistics of a Dutch call center, having two groups of specialists and one

Table 1 **Arrival Rates per Minute, $\lambda_{m,t}$**

	Time t													
m	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0.37	1.15	1.43	1.47	1.33	1.55	1.47	1.48	1.37	1.02	0.72	0.68	0.60	0.37
2	0.80	1.68	2.03	2.28	2.22	2.38	2.27	2.15	2.13	1.60	1.35	1.32	0.98	0.82

group of generalists. Two types of jobs, denoted by 1 and 2, arrive at the call center. The arrival rates during a particular day are given in Table 1, where the first row denotes the index t of each interval, $t \in \{1, 2, \ldots, 14\}$. Each column shows the average rates of both types during one hour. Three agent groups are distinguished, having indices 1, 2 (the specialists), and 3 (the generalists). The service rates for each group and call type are $\mu_{1,1} = 0.186$, $\mu_{2,2} = 0.577$, $\mu_{3,1} = 0.169$, and $\mu_{3,2} = 0.526$. We consider shifts with a length of five and six hours. The costs of a five-hour shift is 5, 4.5, and 4 for generalists, specialists of Type 1, and specialists of Type 2, respectively. The costs of a six-hour shift is 6, 5.5, and 5 for generalists, specialists of Type 1, and specialists of Type 2, respectively. The objective is to compute schedules such that 80% of the callers waits less than 20 seconds, i.e., AWT is 20 seconds and $\alpha = 0.8$, against minimal personnel costs.

We apply the two-step method from §2. Solving the mathematical programming model requires an integer programming solver. We used SA-OPT,[2] which was written by one of the authors. The result of Step 1 from §2.1 is presented in Table 2. The table shows, for each period and agent group, the minimum number of agents to meet the service level. The optimal set of shifts according to the model from §2.2 is presented in Table 3, having an objective value of 167. The columns represent the different periods and each row represents a shift, consisting of the group indices in which the corresponding agent works. The solution consists of eight shifts requiring Skills 1 and 2 (six of length 5 and two of length 2), 14 shifts requiring Skill 1 (eight of length 5 and six of length 6), and 13 shifts requiring Skill 2 (nine of length 5 and four of length 6). The value three indicates that the agents work in agent group 3, i.e., the group of generalists. The value zero denotes idleness, meaning that conditions concerning the service level are already satisfied, such that the employee is redundant in that period.

We note that a generalist sometimes works as a specialist. This is beneficial because specialists have a higher service rate. A second observation is that an agent sometimes idles during a shift. These idle periods can be used for serving other contact channels (such as e-mails and faxes, see §3.1 of the appendix, available online, for training, and for administrative tasks, without compromising the service level.

To check the optimality of the methods from §§2.1 and 2.2, we evaluated the results of Table 3. Two methods were considered for obtaining lower bounds of the objective function. First, we extended the integer programming model from §1 of the appendix, available online, to a multiskill call center. Unfortunately, the number of decision variables turned out to be very large (hundreds of thousands), and, given the fact that all variables must be integer, we were not able to obtain a feasible solution. However, without satisfying the integer requirement, we did succeed in finding an optimal solution, yielding a lower bound of 150. This result was not satisfying because the gap between 150 and 167 is relatively large. Hence, we considered a second approach for obtaining a lower bound. We determined a lower bound for the costs of an agent working during one time interval in a certain group. This can be easily derived from the costs of the shifts, yielding 0.8, 0.9, and 1.0 for specialists of Types 1 and 2 and generalists, respectively. Next, for each period, we calculated the cheapest

Table 2 **Required Group Sizes, $s_{t,g}$ (with Two Skills)**

	Time t													
g	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	6	8	7	8	10	6	8	9	5	5	5	3	2
2	3	5	6	6	7	8	5	6	7	4	5	5	3	2
3	1	2	2	3	1	0	5	2	0	3	0	0	2	2

[2] See http://www.math.vu.nl/~sapot/software/sa-opt for technical details.

Table 3 Optimal Shifts

	Time t													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1		1	1	3	3	1	3							
2		3	3	3	0	1	3							
3	3	3	3	3	0									
4						1	3	1	1	3				
5						2	3	3	1	3				
6							3	3	1	3	2			
7										0	2	2	3	3
8										0	0	2	3	3
9			1	1	1	1	1	1						
10			1	1	1	1	1	1						
11							1	1	1	1	1	1		
12							1	1	1	1	1	1		
13								1	1	1	1	1	1	
14									1	1	1	1	1	1
15	1	1	1	1	1									
16	1	1	1	1	1									
17		1	1	1	1	1								
18		1	1	1	1	1								
19		1	1	1	1	1								
20					1	1	1	1	1					
21					0	1	1	1	1					
22										1	1	1	1	1
23			2	2	2	2	2	2						
24								2	2	2	2	2	2	
25									2	2	2	2	2	2
26									2	2	2	2	2	2
27	2	2	2	2	2									
28	2	2	2	2	2									
29	2	2	2	2	2									
30		2	2	2	2	2								
31		2	2	2	2	2								
32					2	2	2	2	2					
33					0	2	2	2	2					
34					0	2	2	2	2					
35						2	2	2	2	2				

agent configuration that satisfies the service-level constraint. Because there are only three agent groups, this is doable by enumerating all possible configurations and simulations. The lower bound of the total costs for the whole day is calculated by multiplying the group sizes by the costs and by summing over the intervals, yielding 155. However, the optimal solution could be higher than 155, because it is likely that the optimal set of shifts exceeds the staffing levels at certain periods, resulting in idle times, as we saw in Table 3. We calculated a tighter lower bound by determining the idle time that is minimally required. To achieve this, we calculated the minimum number of required agents for each time interval (by summation of the number of specialists and generalists) in a single-skill call center. We solved[3] Dantzig's (1954) model and concluded that the minimum idle time is eight periods, which is equal to the number of idle periods in the solution from Table 3. Then, the lower bound becomes $155 + 0.8 * 8 = 161.4$. This shows that the solution from Table 3 is less than 3% from the optimal objective value.

4. Concluding Remarks

The contribution of this paper is a method of shift scheduling in multiskill call centers. This is among the first methods in the literature that are numerically capable of efficiently generating shifts for multiskill call centers (see also Cezik and L'Ecuyer 2008). An advantage is short computation time. Although our experiments deal with two, three, and five skills, computations are still tractable for call centers with more skills. It was our experience that the computation times are in the order of minutes for extremely large call centers, in favor of the optimization procedures from §2.1, which are of logarithmic order in size. Another advantage is that the methods are easy to implement.

In this paper, the integer programming model of the shift-scheduling method is developed for call centers. However, it is also applicable to service systems other than call centers. In general, it can solve shift-scheduling problems in organizations that

- distinguish multiple skills,
- allow employees to work consecutively on different tasks, and
- have employees with identical productivity within the same skill group.

An example is the scheduling of nurses in hospitals. It is likely that staffing levels are expressed similarly to those in call centers—for example, by choosing the staffing levels in each period in such a way that the workload is covered as accurately as possible. It is realistic that some nurses use only one skill to obtain high productivity, while others have several skills to minimize the total number of nurses. Also, the physical location of the different tasks can play a role. If the distance between the location of two tasks is large,

[3] An online tool is available at http://www.math.vu.nl/~sapot/software/shift-scheduling.

it is undesirable to schedule the same employee on these tasks.

As a possible extension, it might be necessary or beneficial to perform Phases 2 and 3 of the labor allocation process several times, and iteratively. This is desired, for example, if scheduled agents become ill and agents are rescheduled, or if workload predictions of a certain job type change. For that reason, it is likely that call center managers prefer fast methods for each separate phase so that they can iterate between the four phases within a short time.

There are different possibilities for future research. It is straightforward to use the model from §2.2 to perform multiskill rostering, i.e., combining daily shifts with weekly rosters. The main difference is that the rows represent the shifts, instead of required group sizes, and each column represents the weekly schedule of an agent, instead of shifts. Then, the schedules can be assigned to the available employees afterward. To handle the large number of possible schedules, column generation (a well-known method from linear programming) can be used. These problems are numerically tractable and have short computation times. This even has the potential to solve Phases 3 and 4 simultaneously.

Another promising method for shift scheduling is the method of Cezik and L'Ecuyer (2008). The advantage of their method is that it takes the transient behavior into account and can solve Phases 2 and 3 simultaneously. However, the computation times become extremely long as the size of the call center increases. If the efficiency of the algorithm can be increased, it would be interesting to combine their method with the integer problem from §2.2.

Note that although it is out of the scope of this paper, suboptimality can be significant in Phase 4 of the labor allocation process. This phase is about the assignment of shifts to employees. Suboptimality can occur if insufficient employees are available to satisfy the requirements for a type of shift requiring a specific set of skills. There are several ways to avoid this, for example, by creating agent groups only with skill sets that occur among the agents. Additionally, the staffing algorithm can be extended by adding constraints on the group sizes or on sums of several group sizes. Afterwards, by studying the results from the shift scheduling step and changing the staffing levels, it is likely that improvements are also possible.

Acknowledgments
The authors first thank the referees for their very useful comments and suggestions. Their contribution was especially crucial for the note of suboptimality in §2.2. The authors also thank Marco Bijvank for improving the readability of an earlier version of the manuscript.

References
Atlason, J., M. A. Epelman, S. G. Henderson. 2004. Call center staffing with simulation and cutting plane methods. *Ann. Oper. Res.* **127** 333–358.

Caprara, A., M. Fischetti, P. Toth. 1999. A heuristic method for the set covering problem. *Oper. Res.* **47** 730–743.

Cezik, M. T., P. L'Ecuyer. 2008. Staffing multiskill call centers via linear programming and simulation. *Management Sci.* **54**(2) 310–323.

Dantzig, G. B. 1954. A comment on Edie's "traffic delays at toll booths." *Oper. Res.* **2**(3) 339–341.

Franx, G. J., G. M. Koole, S. A. Pot. 2006. Approximating multi-skill blocking systems by hyperexponential decomposition. *Performance Eval.* **63** 799–824.

Gans, N., G. M. Koole, A. Mandelbaum. 2003. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing Service Oper. Management* **5** 79–141.

Gomory, R. E. 1958. Outline of an algorithm for integer solutions to linear programs. *Bull. Amer. Math. Soc.* **64** 275–278.

Ingolfsson, A., E. Cabral, X. Wu. 2007. Combining integer programming and the randomization method to schedule employees. Technical report, School of Business, University of Alberta, Edmonton, Alberta, Canada.

Keith, E. G. 1979. Operator scheduling. *AIIE Trans.* **11**(1) 37–41.

Pot, S. A., S. Bhulai, G. M. Koole. 2008. A simple staffing method for multiskill call centers. *Manufacturing Service Oper. Management* **10**(3) 421–428.

Thompson, G. M. 1997. Labor staffing and scheduling models for controlling service levels. *Naval Res. Logist.* **44**(8) 719–740.