A Simulation-based Decomposition Approach for Two-stage Staffing Optimization in Call Centers under Arrival Rate Uncertainty

Thuy Anh Ta[1], Wyean Chan[2], Fabian Bastin[3], and Pierre L'Ecuyer[4,5]

[1] *DIRO, Université de Montréal, member of GERAD and CIRRELT, Pavillon André-Aisenstadt, CP 6128 Succursale Centre-Ville, Montréal QC, Canada, H3C 3J7, tathuyanh1989@gmail.com.*

[2] *DIRO, Université de Montréal, Pavillon André-Aisenstadt, CP 6128 Succursale Centre-Ville, Montréal QC, Canada, H3C 3J7, chwyean@gmail.com.*

[3] *DIRO, Université de Montréal, member of CIRRELT, Pavillon André-Aisenstadt, CP 6128 Succursale Centre-Ville, Montréal QC, Canada, H3C 3J7, bastin@iro.umontreal.ca.*

[4] *DIRO, Université de Montréal, member of GERAD and CIRRELT, Pavillon André-Aisenstadt, CP 6128 Succursale Centre-Ville, Montréal QC, Canada, H3C 3J7.*

[5] *Correponding author, lecuyer@iro.umontreal.ca.*

January 2, 2021

A Simulation-based Decomposition Approach for Two-stage Staffing Optimization in Call Centers under Arrival Rate Uncertainty

Abstract

We study a solution approach for a staffing problem in multi-skill call centers. The objective is to find a minimal-cost staffing solution while meeting a target level for the quality of service to customers. We consider a common situation in which the arrival rates are unobserved random variables for which preliminary forecasts are available in a first stage when making the initial staffing decision. In a second stage, more accurate forecasts are obtained and the staffing may have to be modified at a cost, to meet the constraints. This leads to a challenging two-stage stochastic optimization problem in which the quantities involved in the (nonlinear) constraints can only be estimated via simulation, so several independent simulations are required for each first-level scenario. We propose a solution approach that combines sample average approximation with a decomposition method. We provide numerical illustrations to show the practical efficiency of our approach. The proposed method could be adapted to several other staffing problems with uncertain demand, e.g., in retail stores, restaurants, healthcare facilities, and other types of service systems.

Keywords: stochastic programming, simulation, stochastic optimization, sample average approximation, L-shaped decomposition

1 Introduction

Call centers play a major role in businesses and in public service systems. They are used to provide information and assistance, order food, taxis, or other products or services, receive emergency calls, etc. In multi-skill centers, calls are categorized by the type of service requested. Each call type requires a specific skill and each agent has a subset of all the skills. The agents are partitioned into groups in which all have the same skill set. See Gans et al. (2003) and Koole (2013) for more details.

The *quality of service* (QoS) is often measured by the *service level* (SL), defined as the fraction of calls answered within a given time limit, called the *acceptable wait threshold* (AWT). Selecting a *staffing* means choosing how many agents of each skill set to have in the center. Each agent has a cost that depends on its skill set. The staffing problem consists in finding a staffing that minimizes the total cost, under a set of constraints on the QoS. In applications, the day is usually divided into periods of 15 to 60 minutes and a staffing is selected for each period, based on distributional forecasts of arrival rates or call volumes (Cezik and L'Ecuyer, 2008, Oreshkin et al., 2016). A closely related problem not considered here is the *scheduling* problem

(Avramidis et al., 2010, Bodur and Luedtke, 2017), in which a set of admissible shift schedules is first specified, and the decision variables are the number of agents of each group in each shift.

There are two important issues with most staffing methods proposed in the literature: (i) the arrival rates are often assumed perfectly known, and (ii) the QoS targets (constraints) are usually defined with respect to the long-term expected value, which is an average over an infinite number of days. Perfect knowledge of the arrival rates leads to simpler optimization problems, but arrival rates in real-life call centers are uncertain and their distributions depend on multiple factors, such as the day of the week, time of the day, level of busyness, holidays and special events, etc.; see for instance Channouf et al. (2007), Ibrahim et al. (2016b), and Oreshkin et al. (2016), where actual data can be found. The QoS for a given day should then be modeled as a random variable. A manager who wants to meet the QoS targets for a given proportion of the days, or with a given probability, should impose distributional or chance constraints (Charnes and Cooper, 1959). This is especially true if the distribution of the random variable is unbounded or if the distribution is bounded but the upper bound would lead to far too conservative solutions. In other terms, a solution that satisfies the QoS constraints for all scenarios would be much too expensive. This motivates the use of chance constraints.

Often, the uncertainty in the arrival rates is reduced when we get closer to the target period. For instance, the arrival rate forecasts are usually more accurate in the morning of the target day, or in the evening before, than a week or two earlier when the staffing and work schedules are made. Then it makes sense (and it can be necessary, e.g., for emergency services) to readjust the staffing to account for the updated information. This leads to a two-stage stochastic optimization problem with recourse. In the first-stage, typically several days in advance, one selects an initial staffing based on the currently available forecast of the arrival rates. In the second-stage, closer to the target period, a more accurate forecast becomes available, and recourse actions may be applied to correct the initial staffing by adding or removing agents, at the price of penalty costs.

In this paper, we consider a two-stage model from Chan et al. (2016), in which chance constraints are imposed on the QoS of the day: The recourse must (always) be chosen so that the QoS meets its target with a minimum probability threshold, given the updated forecast. Solving this problem is quite challenging, due to the nonlinearity of the chance constraints, the large number of integer variables, and the fact that the QoS can only be estimated by Monte Carlo simulation, separately for each scenario. Chan et al. (2016) propose to solve this problem via a *sample average approximation* (SAA) approach, in which one generates a number of independent scenarios for the realization of the updated forecast. For each scenario, the chance constraints are approximated by linear cuts obtained by simulating the system for a number of replications to estimate subgradients at several tentative solutions specified by a cutting-plane algorithm, under distributions conditional on the updated forecast. They propose to solve the resulting two-stage linear integer program directly with a standard mixed-integer program (MIP) solver such as CPLEX. This works fine for small problem instances, but the computation time becomes excessive when one increases the size of the problem.

The aim of the present paper is to study a more effective simulation-based decomposition method to solve this SAA. For this, we incorporate a way to strengthen the linear cuts by mixed-integer rounding (MIR) inequalities (Nemhauser and Wolsey, 1990) and we decompose the mixed-integer linear problems by using the L-shaped method (van Slyke and Wets, 1969). With this method, instead of solving the complete MIP directly, we decompose it and iteratively solve a master program that is enriched by linear cuts at each iteration. More linear cuts are added whenever a solution does not satisfy the chance constraints. The problem is formulated for a single time period, but it is not hard to extend the formulation to multiple periods.

We report numerical experiments for staffing problems over a single time period. Our objective of this paper is not to solve realistic problems based on real data, but to explore the efficiency of the decomposition approach. We solve problems ranging from a small example with 2 call types and 2 agent groups to an example of moderate size with 15 call types and 20 agent groups. In these examples, our simulation-based decomposition approach returns good staffing solutions significantly faster than the deterministic equivalent approach examined in Chan et al. (2016). The gain in speed is also larger for larger problem instances, which means that it can make a real difference for large realistic problems.

In Section 2, we review some relevant literature on the staffing and scheduling of multi-skill call centers. In Section 3, we define the two-stage staffing optimization problem and its SAA formulation. We present our decomposition algorithm in Section 4. In Section 5, we compare the performance of the proposed algorithm and the deterministic equivalent approach in multiple numerical experiments. Conclusions are given in Section 6.

2 A Quick Review of Other Related Work

Much of the research on call centers has focused traditionally on single-skill centers, with a single call type (Gans et al., 2003, Green et al., 2003). Multi-skill centers involve routing rules, priorities, etc., and are analytically much more complex than a single queue with a single type of customer. There are no known accurate approximation formulas for QoS measures for them, so these measures must be estimated by computationally-costly simulation. For a multi-skill staffing problem with known arrival rates, Cezik and L'Ecuyer (2008) developed a simulation-based MIP optimization method where linear cuts are added iteratively using estimated subgradients of the SL function. Avramidis et al. (2010) extended this method to solve a shift scheduling problem with multiple periods. These methods are in fact adaptions and generalizations of the method of Atlason et al. (2004) for agent scheduling in single-skill call centers with constraints on the expected SL over an infinite time horizon. The latter method combines simulation with integer programming and cut generation, based on the concavity property of the SL function in the Erlang C model, when the queue is in steady state. The concavity property does not necessarily hold in the multi-skill context, so the methods are heuristic, but they work well empirically. Avramidis et al. (2009), Pot et al. (2008), Wallace

and Whitt (2005) proposed other algorithms for the single-period staffing problem that use crude approximation formulas, search methods, and corrections by simulation. Call routing is also an important aspect that interplays with staffing and scheduling in multi-skill centers: changing the priorities and call routing rules often changes the optimal staffing solution, and vice-versa; see Chan et al. (2014) and the many references given there.

Exact arrival rates in call centers are usually unknown, and it is now well recognized that the rates should be modeled as random variables or stochastic processes (Avramidis et al., 2004, Ibrahim et al., 2012, 2016a, Oreshkin et al., 2016, Steckley et al., 2009, Whitt, 1999). In the simplest cases where a good approximating formula is available for the QoS of interest as a function of the (constant) arrival rate, one can replace the QoS by this approximating formula in the problem formulation and then solve the resulting problem. This requires no simulation. Many have used this type of approach, often in the setting of two-stage stochastic programming; see Bodur and Luedtke (2017), Gans et al. (2015), Green et al. (2001), Gurvich et al. (2010), Harrison and Zeevi (2005), Pot et al. (2008), Robbins and Harrison (2010), for example. But reliable QoS approximations are not always available, and in many cases one has to rely on simulation to estimate the QoS for any given realization of the vector of arrival rates. In these complicated settings where simulation is required, it suffices to generate the random arrival rates at the beginning of the simulation (assuming we know how to generate those random rates), so the fact that the arrival rates are random adds no serious difficulty to the simulation. Some authors model the distributions of *arrival counts* instead of the rates, but this can make the simulation more difficult (Ibrahim et al., 2012, Oreshkin et al., 2016).

Several authors consider stochastic optimization to capture this arrival-rate uncertainty. In the single-skill setting, Liao et al. (2012, 2013) model the uncertain arrival rates by discrete probability distributions, while Gurvich et al. (2010), Helber and Henken (2010), and Robbins and Harrison (2010) use continuous distributions. Robbins and Harrison (2010) consider a stochastic scheduling problem with a penalty cost for missing the SL target. Gans et al. (2015) investigate a two-stage scheduling problem with recourse for a single-skill call center and use Gaussian quadrature for scenarios generation. The forecast is updated during the day, and the schedules can be corrected by adding or removing agents for the latter part of the day. Both Robbins and Harrison (2010) and Gans et al. (2015) construct a MIP in which a set of constraints are generated beforehand by the linearization of the SL and the abandonment ratio, which are taken as the steady-state values given by the analytic formulas for an M/M/s queue with abandonments, and then use a standard MIP solver. It is unclear if and how this type of approach can be generalized to the multi-skill case, for which no analytic formula is available.

For multi-skill call centers with random arrival rates, Harrison and Zeevi (2005), Bassamboo et al. (2006), and later Bodur and Luedtke (2017), approximate the level of *abandonment* by a fluid system, and solve a two-stage staffing or scheduling problem. They seek to minimize the scheduling cost function with a penalty cost on the abandonment ratio. The first-stage decision variables are the schedules, and the second-stage ones control the work assignment of each agent. Bodur and Luedtke (2017) also use a Benders decomposition with MIR inequalities

at the first stage, but no simulation. They assume that the number of arrivals is perfectly known in the second stage (which is never the case in call centers and in our setting, but is a by-product of the fluid approximation). A major drawback when optimizing a fluid system is that it assumes implicitly some kind of idealistic fluid routing policy, which is impossible to implement in a call center. Moreover, no good fluid approximation is available for the type of QoS considered in this paper, and more generally for the SL in multiskill call centers. In a similar vein, Gurvich et al. (2010) optimize a two-stage staffing problem with chance constraints on the steady-state abandonment ratio, for stochastic arrival rates. The requirement is that the QoS can be violated on at most a fraction δ of the arrival rate realizations, where δ represents the level of risk tolerance.

Chan et al. (2016) propose an extension of the cutting-plane method presented in Cezik and L'Ecuyer (2008) to solve a two-stage staffing problem with chance constraints on the SL over the day (not the long term SL). The second-stage decision variables are recourse actions to add or remove agents. They construct an SAA, approximate its constraints by linear cuts, and solve the resulting MIP with a standard solver. Ta et al. (2020) study the convergence properties of the solutions of this SAA to those of the true problem. In this paper, we consider the same model and we also use linear cuts, but we propose a better way to solve the SAA, based on a decomposition approach.

3 Problem Formulation and Sample Average Approximation

We now formulate the two-stage optimization problem considered in this paper. We also give an SAA formulation in which the constraints are approximated by sampling averages. These formulations are similar to those in Chan et al. (2016) and Ta et al. (2020).

3.1 Call Center Model

Consider a call center with K call types indexed from 1 to K, and I agent groups numbered from 1 to I. Agents in group i have the skill set $\mathcal{S}_i \subseteq \{1, \ldots, K\}$, which is the set of call types they can serve. Conversely, $\mathcal{G}_k = \{i : k \in \mathcal{S}_i\}$ is the set of groups that can handle calls of type k. Let $z = (z_1, \ldots, z_I)^{\mathrm{T}}$ be the staffing vector, which gives the number of agents in each group. We assume that calls of type k arrive from a time-homogeneous Poisson process with unknown rate Λ_k over the entire period, where the Λ_k are independent random variables with bounded support but otherwise arbitrary distributions. This models the forecasting uncertainty.

Agents in the same group are homogeneous and when an agent in group i serves a call of type k, the service time has a known distribution, for each pair (i, k). A call abandons the queue (and the call center) when its waiting time exceeds its patience time, which is a random variable with known distribution that may depend on the call type k. Calls are assigned to agents by an arbitrary routing policy whose details are not discussed here. In this paper, we do not optimize

the routing policy; we assume it is fixed. One advantage of simulation-based optimization is that there is no need to impose a specific form of routing policy or specific family of probability distributions in the model, as long as it can be simulated easily. For example, the service time can be exponential, lognormal, gamma, etc. As in Wallace and Whitt (2005), Ceşik and L'Ecuyer (2008), and many others, we optimize the staffing for a single time period.

3.2 Service Level Constraints

We measure the QoS by the SL introduced in Section 1, defined as the proportion of callers who wait less than an *acceptable waiting time* (AWT) parameter τ over a given finite time period. This SL is a random variable and the constraints will be probabilistic: the SL must reach a certain target $l \in [0,1]$ with probability at least $1-\delta$ for a given $\delta > 0$. The SL can actually be defined in various ways, depending on how we count abandons, the calls that overlap two or more periods, etc. Here we use a popular definition, implemented (among others) in ContactCenters (Buist and L'Ecuyer, 2005). For a staffing vector z and AWT threshold τ, we define

$$\mathrm{SL} = S(z) = \frac{A(z)}{T - L(z)} \tag{1}$$

where T is the total number of calls that arrived in the period, $A(z)$ is the number of those calls served after waiting at most τ, and $L(z)$ is the number of them that abandoned after waiting less than τ. For other definitions, see Jouini et al. (2013). Several authors replace T, A, and L by their (transient or steady-state) expectations; see for example Atlason et al. (2004), Avramidis et al. (2010), Ceşik and L'Ecuyer (2008). Then the SL is a constant, defined as a ratio of expectations, instead of a random variable, and the constraints are no longer probabilistic.

We define our chance constraints as follows. For each call type k, we select an AWT τ_k and denote $S_k(z)$ the SL for call type k during the selected period, given the staffing vector z. Let $S_0(z)$ denote the aggregate SL for all calls, with AWT τ_0, over the period. The random variables $S_0(z), \ldots, S_K(z)$ have distributions that depend on z. The constraints are:

$$\mathbb{P}[S_k(z) \geq l_k] \geq 1 - \delta_k, \qquad k = 0, 1, \ldots, K,$$

where the l_k are SL targets and the $\delta_k \in (0,1)$ are given risk thresholds.

3.3 Staffing Problem with Recourse

We now describe the two-stage staffing problem. In the first stage, based on an initial forecast that provides a prior distribution for the random arrival rate Λ_k for each call type k, the manager must select an initial staffing $x = (x_1, \ldots, x_I)^{\mathrm{T}}$ at the corresponding cost per agent of $c = (c_1, \ldots, c_I)^{\mathrm{T}}$. Stage 1 can be days or weeks in advance of the target date. Close to the target period (e.g., the previous day or a few hours before), additional information becomes available

that can improve the forecast of the arrival rates Λ_k. Let $\xi \in \Xi$ denote this new information. It could be related to weather conditions, the observed number of arrivals in the preceding period, etc. Let $\mathbb{E}_\xi$ denotes the expectation with respect to ξ and $\mathbb{P}[\cdot \mid \xi]$ be the probability distribution conditional on ξ. In particular, the distribution of Λ_k conditional on ξ is not the same as the unconditional one. It usually has a smaller variance.

In the second stage, the manager observes the realization of ξ and based on that, the initial staffing can be modified by adding or removing agents at some penalty costs (by calling them to work at the last minute or offering them to go back home, or canceling meetings, etc.). Note that even in the extreme case where the Λ_k are known exactly conditional on ξ, there is still uncertainty in the second stage and the SL is still a random variable. The recourse in Stage 2 consists in modifying the initial staffing by adding $r_i^+(\xi)$ extra agents to group i at a cost of $c_i^+ > c_i$ per agent, or removing $r_i^-(\xi) \le x_i$ agents in group i to save c_i^- per agent, where $0 \le c_i^- < c_i$. After the recourse, the new number of agents in group i is $z_i(\xi) = x_i + r_i^+(\xi) - r_i^-(\xi)$. Let c, c^+, c^-, and $z(\xi)$ be the vectors with components c_i, c_i^+, c_i^-, and $z_i(\xi)$, respectively. We define the recourse vectors as $r^+(\xi) = (r_1^+(\xi), \ldots, r_I^+(\xi))^{\mathrm{T}}$, and $r^-(\xi) = (r_1^-(\xi), \ldots, r_I^-\xi))^{\mathrm{T}}$. Given a staffing $z(\xi)$, the SL for call type k and the aggregate SL are random variables $S_k(z(\xi))$ for $k = 1, \ldots, K$, and $S_0(z(\xi))$, respectively. Let $g_k(z;\xi) = \mathbb{P}[S_k(z) \ge l_k \mid \xi]$ for $k = 0, \ldots, K$. With this, we have the following chance-constrained staffing problem with recourse:

$$
(\mathbf{P1}) \qquad \begin{cases} \min\limits_{x \in X} \quad c^{\mathrm{T}} x + \mathbb{E}_\xi \left[Q(x, \xi) \right], \\ \text{where} \quad Q(x,\xi) = \min\limits_{r^+, r^-} \quad \left\{ (c^+)^{\mathrm{T}} r^+(\xi) - (c^-)^{\mathrm{T}} r^-(\xi) \right\} \\ \qquad\qquad \text{subject to} \quad x + r^+(\xi) - r^-(\xi) = z(\xi), \\ \qquad\qquad\qquad\qquad\quad g_k(z;\xi) \ge 1 - \delta_k, \quad k = 0, \ldots, K, \\ \qquad\qquad\qquad\qquad\quad r^+(\xi), r^-(\xi) \ge 0 \text{ and integer}, \end{cases}
$$

in which $X \subset \mathbb{N}^I$ is the support set of first-stage solutions. The constraints in this formulation are on the SL, but they could also be on other QoS measures such as average waiting times, abandonment ratios, etc. We emphasize that in this formulation, for *every realization* of ξ, the recourse must be selected so that the probabilistic constraints are satisfied. Without this assumption, one could be tempted to put no staffing at all on certain days in which ξ takes a bad value, to save costs. Under our model, this is not allowed.

3.4 The SAA Formulation

Instead of solving the two-stage problem (**P1**), we will solve an SAA version. We generate N *scenarios* (realizations of ξ) by Monte Carlo. Let ξ_n denote the realization of ξ under scenario n. Each scenario provides a different probability distribution of the Λ_k's, conditional on ξ_n, for the second stage. Then we define a discrete probability distribution over these N scenarios by giving probability $p_n > 0$ to scenario n, where $\sum_{n=1}^N p_n = 1$. The usual and simplest approach with independent random scenarios is to put $p_n = 1/N$ for all n, which gives the standard

empirical distribution, and this is what we will do in our numerical examples. For scenario n, we denote $r_n^+ = r^+(\xi_n)$, $r_n^- = r^-(\xi_n)$, and $z_n = (z_{1,n}, \ldots, z_{I,n})^{\mathrm{T}}$. For each ξ_n, we estimate the probability $g_k(z;\xi_n) = \mathbb{P}[S_k(z(\xi_n)) \geq l_k \mid \xi_n]$ in the constraints of (**P1**) by simulating the call center M times independently, over the given period, conditional on ξ_n. These simulations are also independent across the scenarios. We compute the empirical SL $\hat{S}_k^m(z_n;\xi_n)$ for each k and each replication m generated conditional on ξ_n, and we estimate $g_k(z;\xi_n)$ by the proportion of the M replications for which the SL constraint was met:

$$\hat{g}_{k,M}(z_n;\xi_n) = \frac{1}{M}\sum_{m=1}^{M} \mathbb{I}[\hat{S}_k^m(z_n;\xi_n) \geq l_k] \quad \text{for } k = 0, \ldots, K, \tag{2}$$

where $\mathbb{I}[\cdot]$ is the indicator function. With these ingredients, the SAA can be written as

$$(\mathbf{P2}) \quad \begin{cases} \min\limits_{x,\, r_n^+,\, r_n^-} & c^{\mathrm{T}}x + \sum\limits_{n=1}^{N} p_n \left[(c^+)^{\mathrm{T}} r_n^+ - (c^-)^{\mathrm{T}} r_n^-\right], \\ \text{subject to} & \begin{cases} x + r_n^+ - r_n^- = z_n, \quad \text{for } n = 1, \ldots, N, \\ \hat{g}_{k,M}(z_n;\xi_n) \geq 1 - \delta_k, \quad \text{for } k = 0, \ldots, K,\ n = 1, \ldots, N, \\ x \in X,\ r_n^+, r_n^- \geq 0 \text{ and integer, for } n = 1, \ldots, N. \end{cases} \end{cases}$$

Ta et al. (2020) investigate the convergence properties of this SAA problem. When N and M increase to infinity, under the assumptions that the first and second stage feasible sets are finite and that the sample average approximations of the constraints almost surely converge the true function at any feasible point, they show that the optimal value and the solutions of the SAA problem converge almost surely to the corresponding ones for the true problem (**P2**). Such conditions hold in typical call center examples. They also study the convergence of large-deviations probabilities.

Two important difficulties arise when solving the SAA (**P2**): (i) the expressions $\hat{g}_{k,M}(z_n;\xi_n)$ in the constraints are nonlinear and (ii) (**P2**) is expensive to solve when N is large. To handle issue (i), Cezik and L'Ecuyer (2008) use a cutting-plane method in which the nonlinear constraints are replaced by several linear cuts. This approach can work reasonably well in the simple situation where there is no recourse. But in the two-stage setting, the SAA (**P2**) becomes much more expensive to solve when there is a large number of scenarios. This motivates our introduction of a decomposition method for (**P2**) in the next section.

4 General Methodology with a Decomposition Approach

We now introduce our proposed decomposition approach to solve the two-stage staffing optimization problem. To deal with the chance constraints in the SAA, we use a cutting-plane approach, but we also incorporate a way to strengthen the linear cuts, and we use a simulation-based decomposition algorithm that allows us to efficiently find good staffing solutions.

4.1 Cut Generation

We recall the cutting-plane method used by Atlason et al. (2008) and Cezik and L'Ecuyer (2008) to approximate the chance constraints by linear ones. The method relies on the hypothesis that the SL function is concave in z, at least around the optimal solution. In this paper, instead of being on the expected SL as in those previous papers, our constraints are on a tail probability of the SL, $\mathbb{P}[S(z) \geq l]$, and it is this probability function rather than the expected SL that is assumed to be concave near the optimum. Chan et al. (2016) observe that in the single-skill case, these probability functions typically have an S shape: they are convex for small z and concave for large enough z, just like the expected SL. In multi-skill settings, however, the multivariate concavity of $g_k(z;\xi_n)$ with respect to z can be very difficult to verify. And even when it holds, the SAA counterpart $\hat{g}_{k,M}(z;\xi_n)$ may very well fail to be concave near the optimum. For this reason, this cutting-plane approach is really a heuristic. Moreover, we have also observed empirically that the "S shape" tends to be more compressed (the S is less elongated) for the probability function than for the expected SL, and the region of (approximate) concavity tends to be smaller, which makes the problem harder to solve by the cutting-plane approach. For this reason, we have to be more careful when selecting the initial constraints with a fluid approximation as discussed below.

In our approach, we consider each scenario separately, and for each staffing solution that violates a chance constraint for that scenario, we generate linear cuts based on an (tentative) estimation of the sub-gradient at that staffing point. After adding enough cuts, we obtain a feasible staffing solution for the chance constraints, for the given scenario. The result of this procedure is a set of linear cuts that serve as an approximation of the chance constraints and which are used to solve the two-stage problem.

The cutting-plane method is an iterative algorithm that starts at an infeasible solution z and adds new linear cuts based on the sub-gradient of $\hat{g}_{k,M}(z)$ until a feasible solution is obtained. As in Cezik and L'Ecuyer (2008), and Chan et al. (2016), to avoid starting the algorithm at a null solution ($z = 0$) or with an initial staffing that is much too low for concavity to hold, we rely on a heuristic that uses a fluid model approximation and adds linear constraints to impose that the system has enough capacity (in the fluid model) to serve at least a fraction α_k of the expected number of arrivals for each call type k. We emphasize that this is usually not an accurate approximation of the true constraints, it is only a crude approximation used to put some initial "minimal requirements" (or lower bound) on the staffing. The constraints can be written as

$$\sum_{i \in \mathcal{G}_k} \mu_{k,i} w_{k,i,n} \geq \alpha_k \, \mathbb{E}[\Lambda_{k,n} \mid \xi_n], \qquad k = 1, \ldots, K$$

$$\sum_{k \in \mathcal{S}_i} w_{k,i,n} \leq z_i, \qquad i = 1, \ldots, I$$

$$w_{k,i,n} \geq 0, \qquad k = 1, \ldots, K, \; i = 1, \ldots, I,$$

in which the time unit is the length of the period, $\mathbb{E}[\Lambda_{k,n} \mid \xi_n]$ is the expected number of arrivals for call type k in scenario n, $1/\mu_{k,i}$ is the expected service time for call type k by agent group i, $w_{k,i,n} \geq 0$ represents the (fractional) number of agents of group i assigned to calls of type k in scenario n, and we want to select the parameters α_k so that the initial solution is in a region in which the $\hat{g}_{k,M}$ are likely to be concave. If $\alpha_k < 1$, over an infinite time horizon it would mean that at least a fraction $1 - \alpha_k$ of the calls abandon, on average. Over a finite time horizon, this fraction could be a little smaller because a few of these calls can be served after the end of the period. So if we expect few abandons, it makes sense to take α_k close to 1, then iteratively use simulation to estimate the probability values in the constraints, and increase α_k if they are too small. But with the probability constraints, one must pay more attention to the choice of α_k, as explained earlier. To do this, we used the following heuristic in our experiments. We select a threshold $\rho > 0$ (e.g., $\rho = 0.5$), and we add agents (by increasing α_k) to the groups that serve call type k if the estimated tail probability in (2) is smaller than ρ. We stop this procedure when all the probability values are larger than ρ. We expect (and assume) that after this, the staffing belongs to the concave region and the sub-gradient cuts are valid. In our experiments, we encountered cases where α_k went as large as 4; see Section 5.3. Such large values of α_k are often required when call k has a low volume or a very demanding SL requirement. In some emergency call centers, for example, agents may be busy less than 25–30% of their time, to make sure that almost all calls are answered immediately. Thus, reasonable choices for the α_k's depend very much on the application.

Then we generate sub-gradient-based linear cuts independently for each scenario, as follows. For scenario n with realization ξ_n, let z^* denote the optimal solution under the current set of linear constraints, let $g_k^n(z) = \hat{g}_{k,M}(z, \xi_n)$ and let $q_{nk}(z^*)$ denote the estimated sub-gradient of g_k^n at z^*, which is a I-dimensional vector whose element i is defined as

$$q_{nk}^i(z^*) = [g_k^n(z^* + de_i) - g_k^n(z^*)]/d,$$

where e_i is the ith unit vector (with 1 at position i and 0 elsewhere), $d \geq 1$ is an integer, and the simulations required to compute g_k^n at the two different values to obtain the finite difference are always made with well-synchronized common random numbers, as explained in L'Ecuyer and Buist (2006). If the empirical function $g_k^n(z)$ was guaranteed to be always concave in each coordinate, we could always take $d = 1$, but this function is somewhat noisy, so it may fail to be concave even where its expectation is concave, especially when M is small, and it is sometimes safer to take $d = 2$ or 3 for this reason. This issue is discussed in Ceşik and L'Ecuyer (2008).

If $q_{nk}(z^*)$ is a sub-gradient of g_k^n at z^*, then we have $g_k^n(z^*) + q_{nk}(z^*)(z - z^*) \geq g_k^n(z)$. Since we want $g_k^n(z) \geq 1 - \delta_k$, the following inequality must hold:

$$q_{nk}(z^*)z \geq 1 - \delta_k - g_k^n(z^*) + q_{nk}(z^*)z^*. \tag{3}$$

We add this inequality as a constraint (a linear cut) to the linear program for scenario n, which

reads as:

$$\min_{(z,w)\in\mathbb{N}^I\times\mathbb{R}_+^{K\times I}} \{c^{\mathrm{T}}z \mid \ A^n z \le b^n,\ \mathcal{H}^n z + \mathcal{K}^n w \le h^n\}, \tag{4}$$

where $A^n z \le b^n$ refers to the set of sub-gradient cuts and $\mathcal{H}^n z + \mathcal{K}^n w \le h^n$ are constraints given by the fluid model. The cutting-plane procedure permits one to approximate (**P2**) by a mixed-integer linear programming (MIP) model. Proposition 1 below states that under appropriate concavity assumptions, by adding enough cuts to approximate the chance constraints, we can obtain an optimal solution to (**P2**) by solving the corresponding MIP.

Let $\widehat{Q}(x) = \frac{1}{N}\sum_{n=1}^{N} \widehat{Q}_M(x;\xi_n)$ denote the value of the second stage of (**P2**) for a given x, where

$$\begin{aligned} \widehat{Q}_M(x;\xi_n) = \ \min \quad & (c^+)^{\mathrm{T}}r^+ - (c^-)^{\mathrm{T}}r^- \\ \text{subject to} \quad & \hat{g}_{k,M}(x + r^+ - r^-,\ \xi_n) \ge 1 - \delta_k \qquad k = 0,\dots,K \\ & r^+, r^- \in \mathbb{N}^I. \end{aligned}$$

For each scenario ξ_n, we denote by $\overline{Q}_M(x;\xi_n)$ the value of the second stage after replacing the constraints $\hat{g}_M(z;\xi_n) \ge 0$ by the linear cuts, i.e.,

$$\text{(\textbf{P3})} \qquad \begin{cases} \overline{Q}_M(x;\xi_n) = \min\limits_{r^+,r^-} & (c^+)^{\mathrm{T}}r^+ - (c^-)^{\mathrm{T}}r^- \\ \qquad\qquad \text{subject to} & A^n(x + r^+ - r^-) \le b^n \\ & \mathcal{H}^n(x + r^+ - r^-) + \mathcal{K}^n w \le h^n \qquad r^+, r^- \in \mathbb{N}^I \end{cases}$$

where $A^n(x + r^+ - r^-) \le b^n$ are the linear cuts added for scenario n. By incorporating these replacements in (**P2**), we obtain the following large MIP, whose solution approximates the solution of (**P2**):

$$\text{(\textbf{P4})} \qquad \min_{x\in\mathbb{N}^I} \left\{ \bar{f}(x) = c^{\mathrm{T}}x + \frac{1}{N}\sum_{n=1}^{N} \overline{Q}_M(x;\xi_n) \right\}.$$

The next proposition says that if the linear cuts are always upper bounds on the chance constraints and we add enough of them, we obtain an optimal solution for (**P2**) by solving (**P4**).

Proposition 1 *Suppose that each time a linear cut of the form (3) is added to (**P4**), $q_{nk}(z^*)$ is a sub-gradient of the function g_k^n within the current feasible set, so that the inequality (3) is valid. If $(x^*, \bar{f}^*)$ is an optimal solution and the optimal value of (**P4**) and if (r_n^{*+}, r_n^{*-}) is an optimal solution to (**P3**) such that $\hat{g}_{k,M}(x^* + r_n^{*+} - r_n^{*-}\ ;\ \xi_n) \ge 1 - \delta_k$ for all n and k, then $(x^*, \bar{f}^*)$ is also an optimal solution and the optimal value to (**P2**).*

Proof. Under the given assumption, given a first-stage solution x, we always have

$$\left\{ (r^+, r^-) \,\middle|\, \begin{array}{l} \hat{g}_M(x + r^+ - r^-; \xi_n) \ge 0 \\ r^+, r^- \in \mathbb{N}^I \end{array} \right\} \subseteq \left\{ (r^+, r^-) \,\middle|\, \begin{array}{l} A^n(x + r^+ - r^-) \le b^n \\ \mathcal{H}^n(x + r^+ - r^-) + \mathcal{K}^n w \le h^n \\ r^+, r^- \in \mathbb{N}^I,\ w \ge 0 \end{array} \right\}, \tag{5}$$

where $A^n(z) \leq b^n$ is the collection of all linear cuts of the form (3). We denote by $\{x_1^*, r_{1n}^{*+}, r_{1n}^{*-},\ n = 1, \ldots, N\}$ an optimal solution to (**P2**) and by $\{x_2^*, r_{2n}^{*+}, r_{2n}^{*-},\ n = 1, \ldots, N\}$ an optimal solution to (**P4**). According to (5) we have

$$c^{\mathrm{T}} x_1^* + \frac{1}{N} \sum_{n=1}^{N} (c^+)^{\mathrm{T}} r_{1n}^{*+} - (c^-)^{\mathrm{T}} r_{1n}^{*-} \geq c^{\mathrm{T}} x_2^* + \frac{1}{N} \sum_{n=1}^{N} (c^+)^{\mathrm{T}} r_{2n}^{*+} - (c^-)^{\mathrm{T}} r_{2n}^{*-}. \tag{6}$$

Moreover, if $\hat{g}_{k,M}(x_2^* + r_{2n}^{*+} - r_{2n}^{*-}; \xi_n) \geq 1 - \delta_k$ for all n, k, then $\{x_2^*, r_{2n}^{*+}, r_{2n}^{*+},\ n = 1, \ldots, N\}$ is also a feasible solution to (**P2**), so

$$c^{\mathrm{T}} x_1^* + \frac{1}{N} \sum_{n=1}^{N} (c^+)^{\mathrm{T}} r_{1n}^{*+} - (c^-)^{\mathrm{T}} r_{1n}^{*-} \leq c^{\mathrm{T}} x_2^* + \frac{1}{N} \sum_{n=1}^{N} (c^+)^{\mathrm{T}} r_{2n}^{*+} - (c^-)^{\mathrm{T}} r_{2n}^{*-}. \tag{7}$$

From (6) and (7) we can deduce that $\{x_2^*, r_{2n}^{*+}, r_{2n}^{*+},\ n = 1, \ldots, N\}$ is also an optimal solution to (**P2**). This completes the proof. ■

So in principle, we can obtain an optimal solution to the SAA problem (**P2**) by adding enough linear cuts to the second-stage problems and then solve the MIP (**P4**) via a standard solver such as CPLEX. However, in a large scale setting and as the number of scenarios increases, (**P4**) would be too large and too hard to solve directly. We can then rely on the L-shaped algorithms presented next.

4.2 L-shaped Algorithm

For any first-stage solution x of (**P4**), evaluating this solution requires solving N second-stage sub-problems of the form (**P3**). This suggests a Benders decomposition approach for the two-stage problem, also known as L-shaped in stochastic programming. Introduced by Benders (1962), the method has received a lot of attention and improvements, and is now a standard approach in many applications, as reflected in the survey of Rahmaniani et al. (2017). However, the usual L-shaped algorithm exploits strong duality at the second stage, a property that does not hold for our problem as it involves integer variables at both first and second stages. Solving the SAA in our setting also requires expensive simulation runs to evaluate the constraints at the second stage, and therefore solving it exactly, even with a decomposition method, is very challenging when N and M are large. The absence of general efficient methods for this type of problem reflects this difficulty (see Birge and Louveaux, 2011, Chapter 7). Several techniques have been proposed over the years, but these techniques are either expensive, or developed under specific restrictions on the two-stage problem, e.g., that the recourse matrix has only integer coefficients, which is not the case in our context. In what follows, we present a simple integer L-shaped algorithm that can be combined with mixed-integer rounding inequalities (in Section 4.3) to efficiently find good integer solutions for the two-stage problem.

The general idea of the L-shaped method is to build a master problem in x and add a first-stage variable capturing the value of the recourse objective, considerably reducing the problem

size. We then reconstruct the second-stage information by deriving linear constraints (cuts) on the master problem variables. Here, we do this by considering a continuous relaxation of the second-stage problem and using the duality properties of this relaxation that allow us compute subgradients of the recourse objective.

For any first-stage solution x, to get a feasible solution for the second stage, we just need to add a large enough vector r^+ of agents and set $r^- = 0$. This means that the problem (**P1**) has a relatively complete recourse, i.e., that the second-stage problems always have feasible solutions, for any given first-stage solution x (see Birge and Louveaux, 2011, Page 113). In addition, under the concavity assumption, the linear cuts generated from the cutting-plane method (Section 4.1) are upper bounds on the chance constraints. In other terms, the linearized second-stage problem will be a relaxation of the true second-stage problem, and consequently, any feasible solution of the true second-stage problem will be feasible for the relaxed second-stage problem. Therefore, (**P4**) is also relatively complete. Thus, when applying an L-shaped method to solve (**P4**), we only need to add optimality cuts, i.e., linear cuts to build a piecewise linear function that approximates the recourse function, to the master problem.

We can write the master problem of (**P4**) as

$$(\textbf{MP1}) \qquad \begin{cases} \min\limits_{x,\theta} & c^{\mathrm{T}}x + \theta \\ \text{subject to} & \Pi x - \mathbf{1}\theta \leq \pi_0 \\ & x \in X \end{cases} \tag{8}$$

where the variable $\theta \in \mathbb{R}$ serves as an underestimation of the second-stage objective function, while the constraints (8) are optimality cuts obtained on the continuous relaxation of the second-stage problems. Suppose the constraints of the second-stage problem for scenario n can be written as $T^n x + W^n y = r^n$, where y is the vector of second-stage variables. In our context, y contains r^+, r^-, and w (coming from the fluid model). The continuous relaxation of the second-stage problem associated to scenario n is

$$\min_y \left\{ q^{\mathrm{T}} y \;\middle|\; T^n x^* + W^n y = r^n,\ y \geq 0 \right\}.$$

Let σ_n be a solution of the dual of the continous relaxation, i.e.

$$\sigma_n \in \arg\max_{\sigma} \left\{ (r^n - T^n x^*)^{\mathrm{T}} \sigma \;\middle|\; (W^n)^{\mathrm{T}} \sigma \leq q \right\}.$$

It is easy to show that the optimality cut

$$\theta \geq \frac{1}{N} \sum_{n=1}^{N} \sigma_n^{\mathrm{T}} (r^n - T^n x),$$

is a valid cut for the stochastic integer program (**P4**) (Birge and Louveaux, 2011, Section 7.1).

It is also possible to add several cuts per master iteration as we can partition the set of all

scenarios into L disjoint subsets $N_1, \ldots, N_L$ and reformulate (**MP1**) as

$$(\textbf{MP2}) \quad \begin{cases} \min\limits_{x,\theta_1,\ldots,\theta_L} & c^{\mathrm{T}}x + \sum\limits_{l=1}^{L} \theta_l \\ \text{subject to} & \Pi^l x - \mathbf{1}\theta_l \leq \pi_0^l, \qquad l = 1, \ldots, L \\ & x \in X, \end{cases} \tag{9}$$

where the constraints (9) are optimality cuts given by L subsets of scenarios. For each subset N_l, the following optimality cut can be added to the master problem:

$$-\frac{1}{N_l}\left(\sum_{n\in N_l} \sigma_n^{\mathrm{T}} T^n\right) x - \theta_l \leq -\frac{1}{N_l}\sum_{n\in N_l} \sigma_n^{\mathrm{T}} r^n. \tag{10}$$

We summarize this L-shaped approach in Algorithm 1. If $L = 1$, we have a single-cut L-shaped algorithm in which only one cut is generated per iteration. The multi-cut L-shaped algorithm is obtained by setting $L = N$, as we generate cuts for each scenario (Birge and Louveaux, 2011, Page 198). The L-shaped algorithm produces a sequence of first-stage candidates $\{x^0, x^1, \ldots\}$, and based on the properties of the optimality cuts, the algorithm stops when it finds a candidate solution that was already seen previously (van Slyke and Wets, 1969).

Algorithm 1: L-shaped algorithm

repeat

Select L clusters of scenarios that form a partition of all scenarios

Solve (MP2) to obtain a solution $(x^*, \theta_1^*, \ldots, \theta_L^*)$

Compute

$$\overline{Q}(x^*) = \sum_{n=1}^{N} \min_y \left\{ q^{\mathrm{T}} y \;\middle|\; T^n x^* + W^n y = r^n,\ y \geq 0 \right\}$$

if $\sum_{l=1}^{L} \theta_l^* < \overline{Q}(x^*)$ **then**

Add L optimality cuts to (MP2)

until $\sum_{l=1}^{L} \theta_l^* \geq \overline{Q}(x^*)$;

Return x^* as a first-stage solution

4.3 Strengthening the Cutting Planes

In this section we present a way to strengthen the sub-gradient cuts defined in (3) by using mixed-integer rounding (MIR) inequalities. This approach plays a central role in the development of strong cutting planes for mixed-integer programming. MIR inequalities can be derived from a single mixed-integer constraint, and have been shown to be able to generate all facets inducing valid inequalities for any mixed 0-1 integer program (Nemhauser and Wolsey, 1990). Bodur and Luedtke (2017) exploit such MIR inequalities for the first-stage decisions, the second-stage problems being continuous. In our case, we only focus on the second-stage problems, the master problem computational cost being negligible compared to the second-stage programs

that require simulation.

Consider a sub-gradient cut of the form $\sum_{i=1}^{I} a_i z_i \geq b$. Since the sub-gradients are always generated to be non-negative, we have $a_i \geq 0$ for all $i = 1, \ldots, I$. Let $\mathcal{P} = \{z \in \mathbb{N}^I \mid \sum_{i=1}^{I} a_i z_i \geq b\}$ be the set of feasible solutions under the sub-gradient cuts.

Proposition 2 *The following inequalities hold for all $z \in \mathcal{P}$:*

$$\sum_{\substack{t=1,\ldots,I \\ t \neq i}} a_t z_t + d_i a_i z_i \geq \left\lceil \frac{b}{a_i} \right\rceil d_i a_i, \qquad \forall\ i = 1, \ldots, I, \tag{11}$$

where $d_i = b/a_i - \lceil b/a_i \rceil + 1$.

Proof. Given $i \in \{1, \ldots, I\}$ such that $a_i > 0$, we can write the inequality $\sum_{i=1}^{I} a_i z_i \geq b$ as

$$\sum_{\substack{t=1,\ldots,I \\ t \neq i}} \frac{a_t z_t}{a_i} + z_i \geq \frac{b}{a_i},$$

which can be written as

$$\sum_{\substack{t=1,\ldots,I \\ t \neq i}} \frac{a_t z_t}{a_i} \geq \frac{b}{a_i} + 1 - \left\lceil \frac{b}{a_i} \right\rceil + \left\lceil \frac{b}{a_i} \right\rceil - z_i - 1. \tag{12}$$

Since $z_i \in \mathbb{N}$, we consider the two cases $z_i \geq \lceil b/a_i \rceil$ and $z_i \leq \lceil b/a_i \rceil - 1$. If $z_i \geq \lceil b/a_i \rceil$ then

$$\sum_{\substack{t=1,\ldots,I \\ t \neq i}} \frac{a_t z_t}{a_i} \geq \left(1 + \frac{b}{a_i} - \left\lceil \frac{b}{a_i} \right\rceil\right) \left(\left\lceil \frac{b}{a_i} \right\rceil - z_i\right) = d_i \left(\left\lceil \frac{b}{a_i} \right\rceil - z_i\right), \tag{13}$$

as the left side of the inequality is non-negative and the right side is non-positive. Moreover, if $z_i \leq \lceil b/a_i \rceil - 1$, given that $b/a_i - \lceil b/a_i \rceil + 1 \leq 1$, from (12) we obtain

$$\sum_{\substack{t=1,\ldots,I \\ t \neq i}} \frac{a_t z_t}{a_i} \geq d_i + d_i \left(\left\lceil \frac{b}{a_i} \right\rceil - z_i - 1\right) = d_i \left(\left\lceil \frac{b}{a_i} \right\rceil - z_i\right). \tag{14}$$

We obtain (11) by combining (13) and (14). ■

We now consider a set of feasible staffing solutions at scenario n after adding sub-gradient cuts and fluid constraints $\mathcal{P}^n = \left\{A^n z \leq b^n,\ \mathcal{H}^n z + \mathcal{K}^n w \leq h^n\right\}$. Let J be the number of rows of matrix A^n, a^n_{ij} its element on row i and column j, and b^n_j the j^{th} element of vector b^n. The constraints given by the sub-gradient cuts can be strengthened using Proposition 2 as follows. We will use these stronger inequalities in our method.

Corollary 1 *The following inequalities hold for all* $z \in \mathcal{P}^n$

$$\sum_{\substack{t=1,\dots,I \\ t \neq i}} a^n_{jt} z_t + d^n_i a^n_{ji} z_i \leq \left\lceil b^n_j / a^n_{ji} \right\rceil d^n_i a^n_{ji}, \qquad i \in \{1,\dots,I\}, \quad j \in \{1,\dots,J\}, \quad a^n_{ji} \neq 0, \tag{15}$$

where $d^n_{ji} = b^n_j / a^n_{ji} - \lceil b^n_j / a^n_{ji} \rceil + 1$.

4.4 The Simulation-based Algorithm

Algorithm 2 summarizes our complete simulation-based method. The algorithm has two main parts. In the first part, we solve the staffing optimization problem for each scenario separately to approximate the corresponding chance constraints by linear cuts. In the second part, we iteratively solve the two-stage stochastic linear programs in which the chance constraints are replaced by linear cuts. If the second-stage solution is found to be unfeasible for the chance constraints, we use simulation to generate more linear cuts (3) to better approximate the chance constraints. This iterative procedure stops when we find first-stage and second-stage solutions that satisfy all the chance constraints. Proposition 3 states that under reasonable conditions, this will happen after a finite number of steps.

Given that the arrival rates are assumed to be bounded, we can always choose a staffing large enough such that all the probability constraints are satisfied. So, without loss of generality we can assume that the set of feasible staffing solutions at the first stage is finite. Steps 1 and 2 of Algorithm 2 are basically a procedure to separately solve the staffing optimization problem for each scenario, i.e., we iteratively generate cuts and solve the corresponding linear programs until getting a staffing solution satisfying all the chance constraints. An important step of the algorithm is that when there is a call type for which the corresponding probability value in (2) is too small, then we need to adjust the staffing, as the current staffing does not belong to the concave region of the probability function and would result in bad cuts. Moreover, since the linear cuts added after Steps 1 and 2 of Algorithm 2 might be not sufficient to accurately approximate the chance constraints, in Step 3 we need to solve the approximate problem (**P4**) to get first- and second-stage solutions and add more cuts if these solutions do not satisfy the chance constraints. (**P4**) can be solved directly as a, potentially large, MIP problem, as in Chan et al. (2016), as presented in Algorithm 3. Alternatively, we can use the L-shaped decomposition method (Algorithm 1). As the algorithm relaxes the integrality condition on the second-stage variables, we have to solve all second-stage subproblems again after the L-shaped execution in order to obtain integer solutions, as presented in Algorithm 4. We also use the cut strengthening technique discussed in Section 4.3. We now prove that Algorithm 2 is well defined.

Proposition 3 *Assuming that the arrival rates are always bounded from above and the support set* X *of first-stage solutions is finite, Algorithm 2 stops after a finite number of iterations.*

Proof. Since X is finite, this implies that the algorithm must stop after a finite number of

Algorithm 2: Simulation-based two-stage staffing with recourse

 # 1. Initialization

– Select a threshold $\rho > 0$ to determine a "concave region" for the functions $\hat{g}_M$, e.g., $\rho = 0.5$

– Add preliminary constraints using the fluid model approximation

– Select a step size $d \in \mathbb{N}^*$ for the sub-gradient estimations and $s \in \mathbb{N}^*$

 # 2. Construction of piecewise linear approximations of second-stage chance constraints

For each scenario $n = 1, \ldots, N$

repeat

Solve the corresponding second-stage problem (4)

$$\min_{z,w} \left\{ c^{\mathrm{T}} z \mid A^n z \leq b^n, \mathcal{H}^n z + \mathcal{K}^n w \leq h^n \right\}$$

to obtain a solution z^*

 # 2.1 For each k with too small probability value, add s agents to a group that can serve call type k

repeat

Run the simulation with staffing z^* to obtain $\hat{g}_M(z^*; \xi_n)$

$\bar{k} = \text{argmin}_k \ \hat{g}_{k,M}(z^*; \xi_n)$

if $\hat{g}_{\bar{k},M}(z^*; \xi_n) < \rho$ **then**

Select i randomly and uniformly in $\mathcal{G}_{\bar{k}}$ and set $z_i^* = z_i^* + s$

until $\hat{g}_{\bar{k},M}(z^*; \xi_n) \geq \rho$ *for all* k;

 # 2.2 Add sub-gradient cuts

for $k = 0, \ldots, K$ **do**

if $\hat{g}_{k,M}(z^*; \xi_n) < 1 - \delta_k$ **then**

Add sub-gradient cut (3) to the set $\{A^n z \leq b^n\}$

until $\hat{g}_{k,M}(z^*; \xi_n) \geq 1 - \delta_k$ *for all* k;

– Add valid inequalities for each sub-gradient cuts initialized (as per Corollary 1)

 # 3. Iteratively solving the linear problem and adding more linear cuts

repeat

 # 3.1. Solve the two-stage problem with recourse

– Solve sub-problem (**P4**) and add linear cuts to obtain a first-stage solution x^* as well as recourse solutions (r_n^{*+}, r_n^{*-}), $n = 1, \ldots, N$.

 # 3.2. Add more linear cuts if there are unsatisfied chance constraints

for $n = 1, \ldots, N;\ k = 0, \ldots, K$ **do**

$z_n^* = x^* + r_n^{*+} - r_n^{*-}$

if $\hat{g}_{k,M}(z_n^*; \xi_n) < 1 - \delta_k$, add linear cuts to the set $\{A^n z \leq b^n\}$

until $\hat{g}_{k,M}(x^* + r_n^{*+} - r_n^{*-}; \xi_n) \geq 1 - \delta_k$ *for all* n *and* k *# Stop when all constraints are satisfied*;

steps. In Steps 2 and 3 of Algorithm 2, for each scenario, each time when a staffing solution is infeasible, this solution is removed by sub-gradient cuts. Since the arrival rates are bounded from above, the number of infeasible solutions (r^+, r^-) for the second-stage problem is finite, so the number of added cuts for each scenario must be finite. Therefore, Algorithm 2 converges in a finite number of iterations. ■

Algorithm 3: Simulation-based deterministic equivalent (DE) algorithm

– Solve the MIP sub-problem (**P4**) and obtain a solution x^*

 # Add more linear cuts if there are unsatisfied chance constraints

for $n = 1, \ldots, N;\ k = 0, \ldots, K$ **do**

  $z_n^* = x^* + r_n^{*+} - r_n^{*-}$

  if $\hat{g}_{k,M}(z_n^*; \xi_n) < 1 - \delta_k$ **then**

    Add sub-gradient cut (3) to the set $\{A^n z \leq b^n\}$

Algorithm 4: Simulation-based L-shaped (LS) algorithm with strengthened cuts

– Solve the second-stage relaxation of (**P4**) using the L-shaped (Algorithm 1) and obtain a first-stage solution x^*

 # Compute integer second-stage solutions

– Compute $(r_n^{*+}, r_n^{*-}) = \text{argmin}_{r^+, r^- \in \mathbb{N}^I} \overline{Q}_M(x^*; \xi_n),\ n = 1, \ldots, N$

 # Add more linear cuts if there are unsatisfied chance constraints

for $n = 1, \ldots, N;\ k = 0, \ldots, K$ **do**

  $z_n^* = x^* + r_n^{*+} - r_n^{*-}$

  if $\hat{g}_{k,M}(z_n^*; \xi_n) < 1 - \delta_k$ **then**

    Add sub-gradient cut (3) and corresponding MIR inequalities (11) to the set $\{A^n z \leq b^n\}$

5 Numerical Illustrations

5.1 Algorithms and Experimental Setting

We evaluate the performance of the proposed simulation-based decomposition algorithm using three call center models of different sizes: a small one, a medium one, and a large one. We compare our approach with a direct solution of (**P4**) via a MIP solver such as CPLEX, as in Chan et al. (2016). Problem (**P4**) can be formulated as the following deterministic equivalent:

$$(\textbf{MIP}) \quad \begin{cases} \min\limits_{x, r_n^+, r_n^-} & c^{\mathrm{T}} x + \dfrac{1}{N} \displaystyle\sum_{n=1}^{N} (c^+)^{\mathrm{T}} r_n^+ - (c^-)^{\mathrm{T}} r_n^- \\ \text{subject to} & A^n (x + r_n^+ - r_n^-) \leq b^n,\ \forall n = 1, \ldots, N \\ & \mathcal{H}^n (x + r_n^+ - r_n^-) + \mathcal{K}^n w \leq h^n,\ \forall n = 1, \ldots, N \\ & x \in X,\ r_n^+, r_n^- \in \mathbb{N}^I,\ w \geq 0. \end{cases}$$

When reporting our results, we denote Algorithm 2 by LS and the approach in which (**P4**) is solved directly by CPLEX by DE (deterministic equivalent). In our experiments with the three examples, we use the multi-cut LS (Algorithm 1). We will show in Section 5.7 that this multi-cut version outperforms the single-cut one, especially for medium and large call centers.

When running the algorithms, we use independent random numbers across the scenarios for both the first-stage and second-stage simulations. When estimating subgradients, on the other hand, we use common random numbers across the two terms of the finite difference, as in Ceçik and

L'Ecuyer (2008). We also use common random numbers between LS and DE, when comparing them. We use step sizes $d = 1$ and $s = 1$, and take $M = 1000$ for all examples. To assess the quality of the solutions returned by the algorithms, we perform an out-of-sample evaluation of each returned first-stage solution, on an independent set of scenarios. For this, we take 1000 scenarios for the small example, and 100 scenarios for the medium and large examples. For each scenario, we fix the first-stage solution and we estimate the expected cost of the recourse by doing second-stage simulations and solving the second-stage problem in the same way as when we solve the entire two-stage SAA. We take the average over all scenarios and add the cost of the initial staffing, to obtain an estimate of the total expected cost for the given first-stage solution. We compute and report the "out-of-sample" costs given by the first-stage solutions returned by the LS and DE approaches for these new sets of scenarios.

In our experiments, the cost c_i of an agent of group i is taken as an affine function of its number of skills:

$$c_i = 1 + 0.05(|\mathcal{S}_i| - 1)$$

where $|\mathcal{S}_i|$ is the cardinality of $\mathcal{S}_i$, for all i, and $c = (c_1, \ldots, c_I)^\mathrm{T}$. For the costs of adding or removing agents, we consider three cases, labeled R1, R2, and R3, as defined in Table 1.

Test case	c^+	c^-
R1	$2c$	$0.5c$
R2	$1.5c$	$0.75c$
R3	$1.1c$	$0.9c$

Table 1: Costs of adding and removing agents

The arrival rate λ_k for call type k (for the entire period) is the realization of a random variable Λ_k of the form $\Lambda_k = \xi_k \beta_k$ (a product of two random variables), where the realization of ξ_k is unveiled at the beginning of the second stage, while β_k remains unknown, but has a smaller variance than Λ_k. This product form agrees with standard random arrival-rate models discussed in Avramidis et al. (2004), Ibrahim et al. (2012), Oreshkin et al. (2016), for example. Realistic models for actual call centers are likely to be more complicated, but for our purpose of showing the ideas, we prefer to keep the setting simple. For our illustrations, we take simple choices of distributions: we suppose that ξ_k has a truncated normal distribution with parameters that generally depend on k and that β_k follows a symmetric triangular distribution of mean and mode 1, minimum 0.9, and maximum 1.1 (see Avramidis et al., 2004). The normal distribution is truncated to satisfy the assumptions of Proposition 3, although the truncated part has a very small probability and the impact of the truncation is negligible (we checked this with additional experiments). To select the parameters α_k in the initialization stage, we used the method described in Section 4.1, with $\rho = 0.5$.

The experiments were conducted on a machine running Debian 8 with Intel(R) Xeon(R) E5620 CPUs running at 2.40GHz. This computer has two CPUs with a total of 8 cores (or 16 threads) and 98GB of memory. The simulations were made using the *ContactCenters* simulation software

(Buist and L'Ecuyer, 2005), developed in Java with the SSJ simulation library. The algorithms were coded in MATLAB and linked to IBM ILOG CPLEX 12.6 optimization routines under default settings. To speed up the computations, the steps of performing simulations and adding sub-gradient cuts for each scenario were run in parallel using all CPU cores.

5.2 Example 1: A Small Call Center

We first consider a small call center with $K = 2$ call types and $I = 2$ agent groups, with $\mathcal{S}_1 = \{1\}$ and $\mathcal{S}_2 = \{1, 2\}$. Agents in group 2 prioritize calls of type 2 over those of type 1, and arriving calls of type 1 are first routed to idle agents in group 1, if any. This small example will permit us to use more scenarios than the larger ones. We assume that for the two call types: (i) each caller abandons immediately with probability 2% if it has to wait, (ii) patience times are exponential with means 10 and 6 minutes, (iii) the "mean" arrival rate ξ_k follows a normal distribution with means 100 and 70 calls per hour, with standard deviations that are 15% of the means, and truncated to 2.5 standard deviations, and (iv) all service times are exponential with means 10 and 7.5 minutes. The length of the period is one hour. The parameters in the SL constraints are $\tau_1 = \tau_2 = \tau_0 = 120$ (seconds), $l_1 = l_2 = 80\%$, and $l_0 = 85\%$. For each case (R1, R2, and R3), we generate 5 independent sets of 100, 200, 300, 400, 500 scenarios. The parameters α_k for the initial constraints with the fluid model are taken as $\alpha_1 = \alpha_2 = 1$.

Steps 1 and 2 of Algorithm 2 are shared by the LS and DE methods, so only the computation time for Step 3 differs between the two approaches. We report the latter to compare the two methods, and we also report the total CPU time for the three steps, so one can judge the relative contribution of Step 3. We will find that Step 3 takes only a small fraction of the total CPU time for the small examples, and takes most of the time for the large examples. We note that most of the computations for Step 3 used Java code and CPLEX, whereas the first two steps were implemented in MATLAB and could probably be made more efficient. For DE, the MIP problem **(MIP)** is typically very large because of the large number of scenarios, and CPLEX cannot find an optimal solution even with a time budget of several hours. We set the time limit to 200 seconds and the optimality gap to 0.05% for CPLEX when solving (**MIP**). In practice, the time limit was always reached. In contrast, LS solved each linear problem in less than a second, and Algorithm 1 ran in a few seconds only, so we did not set a time limit for CPLEX.

The results for the three cost structures R1, R2, and R3 are reported in Table 2. The smallest costs and CPU times are in bold. For this small example, the "in-sample" objective values returned by both approaches are similar. Out of 30 cases, LS wins 9 times, DE 9 times and the other cases give an equality. However, LS runs significantly faster than DE (for Step 3) for all instances. In the out-of-sample evaluation, the two methods return solutions having the same cost in 14 cases, LS wins in 9 cases, and DE in 7 cases. Overall, LS performs better for this example, because it returns its solution faster, and the quality of the solution is comparable. In all cases, the out-of-sample costs given by the two methods are quite close in value.

Targets	N	Method	R1				R2				R3			
			Cost	Step 3 CPU time	Total CPU time	Out-of-sample cost	Cost	Step 3 CPU time	Total CPU time	Out-of-sample cost	Cost	Step 3 CPU time	Total CPU time	Out-of-sample cost
(0.80, 0.85)	100	LS	**32.98**	133	1114	33.14	32.53	128	1109	32.11	**31.56**	76	1057	31.61
		DE	33.01	292	1273	**33.03**	32.53	214	1195	32.11	31.57	281	1262	31.61
	200	LS	32.92	207	2163	33.13	**32.00**	217	2173	**32.22**	31.59	171	2127	31.61
		DE	**32.87**	421	2377	**33.02**	32.12	432	2388	32.31	**31.57**	355	2311	**31.54**
	300	LS	32.65	360	3325	33.14	32.18	327	3292	32.22	31.13	378	3343	31.61
		DE	**32.62**	722	3687	**33.03**	**32.11**	578	3543	**32.21**	**32.12**	633	3598	31.61
	400	LS	33.25	503	3961	33.13	32.13	672	4130	32.21	**31.75**	770	4228	**31.54**
		DE	33.25	1142	4600	33.13	32.13	987	4445	32.21	31.76	990	4448	31.61
	500	LS	32.92	742	4975	33.13	32.12	893	5216	32.21	31.50	974	5207	31.61
		DE	32.92	998	5231	33.13	32.12	1298	5531	32.21	31.50	1137	5370	31.61
(0.90, 0.95)	100	LS	35.42	210	1329	35.55	34.96	231	1350	34.62	**34.00**	297	1416	**34.09**
		DE	35.42	312	1430	35.55	**34.92**	345	1464	**34.56**	34.19	526	1645	34.59
	200	LS	**35.32**	205	2235	**35.54**	34.42	316	2346	34.62	34.06	663	2693	34.09
		DE	35.56	523	2553	35.78	34.42	711	2741	34.62	34.06	865	2895	34.09
	300	LS	**35.00**	567	3768	**35.54**	**34.57**	479	3680	34.62	33.55	601	3802	**34.09**
		DE	35.02	865	4066	35.55	34.89	957	4158	**34.56**	**33.23**	987	4188	34.69
	400	LS	35.62	345	4357	**35.54**	34.53	645	4657	34.62	34.18	603	4615	**34.02**
		DE	**35.45**	998	5010	35.55	34.53	1123	5135	34.62	**34.02**	1023	5035	34.56
	500	LS	35.25	652	5873	35.54	**34.53**	812	6033	**34.62**	33.90	986	6207	34.09
		DE	35.25	1203	6424	35.54	34.83	1376	6597	34.66	33.90	1321	6542	34.09

Table 2: Value of the best solution found for (**P4**) (Cost), CPU time for Step 3, total CPU time (both in seconds), and cost of the retained first-stage solution of (**P4**) estimated out-of-sample, for the small call center

5.3 Example 2: A Medium-Size Call Center

We now consider a medium-size call center with $K = 6$ call types and $I = 8$ agent groups. We assume that (i) the callers do not abandon immediately in case they have to wait, (ii) patience times are exponential with means between 36 and 52 minutes, (iii) for the different k, we suppose that ξ_k follows a normal distribution with mean from 0.45 to 9.15 calls per minute and standard deviation which is 10% of the mean, truncated to 2.5 standard deviations on each side of the mean, and (iv) all service times have a lognormal distribution with mean between 5.1 and 11.3 minutes. We take 10 hours for the period length. This corresponds to an oversimplified situation in which the arrival rate is constant over a day. We also ran experiments with a period length of one hour. We take $\tau_k = 120$ (seconds) and $l_k = 80\%$ for $k = 0, \ldots, K$. We try two sets of targets for the chance constraints: (i) $1 - \delta_0 = 85\%$ and $1 - \delta_k = 80\%$, $k = 1, \ldots, K$, and (ii) $1 - \delta_0 = 95\%$ and $1 - \delta_k = 90\%$ for $k = 1, \ldots, K$. The parameters α_k in the fluid model are taken as $1, 4, 1, 1.2, 1, 3$. The values of $\alpha_k = 4$ and 3 may appear surprisingly large (we impose a capacity that is 3 or 4 times the load for certain call types), but this is often required for some low-volume call types, and may depend on call priorities and routing rules. For this reason, the initialization step in Section 4.1 in which we increase α_k by moving a threshold ρ, is very important in multiskill settings. It is also a good idea to increase α_k faster for call types having a lower contribution to the overall load.

Since the simulation here is more expensive than for the small call center of the previous example, we only consider instances with less than 100 scenarios. For each cost structure, we independently generate instances of 20, 50, 70 scenarios and we use the sample size $M = 1000$ to estimate the chance constraints. We solve each instance and report the corresponding first-stage solutions. For the out-of-sample validation, we use 100 independent scenarios. We set a time budget of 10 minutes for CPLEX.

Table 3 reports the results. As in the previous example, the smallest costs and shortest CPU times are indicated in bold. The solutions returned by the two methods have similar costs, both in-sample and out-of-sample, although LS is slightly better more often than DE. However, LS runs significantly faster. Better solutions are obtained when increasing the number of scenarios from 20 to 70 for the cost structures R2 and R3, but not for R1, for which the difference between c^+ and c^- is larger.

Table 4, gives the first-stage solutions, the first-stage costs as well as the averages of the numbers of added or removed agents for the three cases with $N = 70$ scenarios. As for the small call center, we see that the first-stage costs under R1 and R2 are higher than under R3, and the average value of r^+ under R1 and R2 is much smaller than under R3. In the latter case, the recourse costs are smaller, especially the addition of agents, leading to a smaller initial staffing.

We also ran this medium-sized model for a period of one hour, with targets (0.80, 0.85), and the results are in Table 5. They are very similar to the 10-hour case. Of course, the CPU times are much smaller for the one-hour case (we have much less to simulate). We see that LS wins

			Cost		Step 3 CPU time (hours)		Total CPU time (hours)		Out-of-sample cost	
$(1-\delta_k, 1-\delta_0)$	Cases	N	LS	DE	LS	DE	LS	DE	LS	DE
(0.80,0.85)	R1	20	186.90	186.90	**0.26**	0.54	**0.36**	0.64	188.25	**188.11**
		50	**188.10**	188.15	**1.95**	3.43	**2.18**	3.66	**188.02**	188.09
		70	**184.35**	184.63	**1.55**	3.94	**1.91**	4.30	**188.6**	188.91
	R2	20	**179.39**	179.47	**0.74**	3.61	**0.84**	3.71	**186.99**	187.26
		50	179.94	**179.90**	**1.87**	3.10	**2.10**	3.33	185.60	**185.22**
		70	**183.86**	183.87	**3.32**	5.45	**3.68**	5.81	**183.17**	183.32
	R3	20	**180.10**	180.13	**1.44**	2.57	**1.54**	2.67	**187.41**	188.10
		50	**177.43**	177.48	**1.61**	5.25	**1.84**	5.48	184.15	184.15
		70	**175.31**	175.41	**3.57**	10.08	**3.93**	10.44	**183.32**	183.64
(0.90,0.95)	R1	20	191.43	**191.34**	**1.31**	2.94	**1.42**	3.05	194.68	**194.65**
		50	**193.64**	193.78	**1.17**	2.69	**1.42**	2.94	**194.32**	194.57
		70	**191.16**	191.17	2.39	**2.33**	2.78	**2.72**	**195.32**	195.46
	R2	20	185.77	**185.75**	**0.37**	0.87	**0.48**	0.98	**193.20**	193.34
		50	**186.40**	186.46	**2.07**	3.84	**2.32**	4.09	**191.21**	191.32
		70	**190.33**	190.34	**1.44**	4.63	**1.83**	5.02	**189.71**	189.89
	R3	20	185.03	**185.00**	**2.06**	4.00	**2.17**	4.11	195.62	**194.89**
		50	**183.21**	183.30	**2.54**	4.43	**2.79**	4.68	190.72	190.72
		70	**180.47**	180.61	**3.38**	10.65	**3.77**	11.04	189.28	189.28

Table 3: Value of best solution found for (**P4**) (Cost), CPU time for Step 3, total CPU time (in hours), and cost of retained first-stage solution estimated out-of-sample, for the medium-size call center

both ways: shorter CPU times and also lower out-of-sample costs in almost all cases.

5.4 Example 3: A Larger Call Center

We now consider a larger call center with $K = 20$ call types and $I = 15$ agent groups. We assume that (i) the callers abandon with probability 0.1 in case they have to wait, (ii) all patience times are exponential with means 6 minutes, (iii) for each call type k, the arrival rate ξ_k follows a normal distribution with mean from 130 to 260 calls per hour and standard deviations which is 10% of the mean, truncated to 2.5 standard deviations, and (iv) all service times are exponential with means 7.5 minutes. We take $\tau_k = \tau_0 = 20$ (seconds), $l_k = 50\%$ for $k = 1, \ldots, K$, and $l_0 = 80\%$. The length of the period is one hour. For the chance constraints, we try (i) $1-\delta_0 = 85\%$ and $1-\delta_k = 80\%$, $k = 1, \ldots, K$, and (ii) $1-\delta_0 = 95\%$ and $1-\delta_k = 90\%$ for $k = 1, \ldots, K$. For each cost structure R1, R2 and R3, we test LS and DE with 20, 50, and 70 scenarios. For DE, we give CPLEX a time budget of 10 minutes and a MIP gap of 0.05%. We take $\alpha_k = 1$ for all k.

Table 6 reports the results. Both in-sample and out-of-sample, the costs of the retained solutions are slightly better for LS than for DE. However, LS requires much less CPU time for all instances. The out-of-sample costs are also improved when we increase the number of scenarios.

The computing time for one iteration of Step 3 of Algorithm 2 for the LS and the DE can be approximated as $(\nu_{\text{LS}} + \nu)$ and $(\nu_{\text{DE}} + \nu)$, respectively, where ν_{LS} stands for the total computing time to solve (**P4**) by the LS method (Algorithm 2), ν_{DE} is the total computing time required

$(1-\delta_k, 1-\delta_0)$	Case	Algorithm	x^{T}	c^{T} x	Avg. r^+	Avg. r^-	Total cost	Out-of-sample cost
(0.80, 0.85)	R1	LS	(33, 26, 88, 6, 0, 0, 4, 11)	181.30	4.11	10.64	184.35	188.60
		DE	(34, 26, 88, 6, 0, 0, 5, 10)	182.40	3.78	11.27	184.63	188.91
	R2	LS	(34, 26, 92, 6, 0, 0, 6, 10)	187.70	4.30	14.40	183.86	183.17
		DE	(33, 26, 92, 5, 0, 0, 6, 11)	186.60	3.99	11.34	183.87	183.32
	R3	LS	(33, 23, 84, 7, 0, 0, 3, 4)	165.90	15.36	9.24	175.31	183.32
		DE	(33, 23, 84, 8, 0, 0, 2, 5)	167.10	14.06	8.67	175.41	183.64
(0.90, 0.95)	R1	LS	(37, 25, 91, 4, 3, 0, 6, 7)	186.70	4.71	10.69	191.16	195.32
		DE	(36, 26, 92, 4, 2, 0, 6, 7)	186.55	4.76	10.63	191.17	195.46
	R2	LS	(32, 27, 93, 7, 2, 0, 4, 12)	190.90	5.21	11.04	190.33	189.71
		DE	(32, 27, 94, 6, 2, 0, 4, 13)	192.00	4.63	11.20	190.34	189.89
	R3	LS	(32, 24, 86, 8, 0, 0, 3, 5)	170.15	17.33	10.66	180.47	189.28
		DE	(32, 24, 86, 8, 0, 0, 3, 5)	170.15	17.03	10.11	180.61	189.28

Table 4: First-stage solutions, first-stage costs and average number of additions and removals of agents for $N = 70$ for the medium call center

by CPLEX to solve **(MIP)**, and ν is the CPU time required to perform the simulation and add more sub-gradient cuts for unsatisfied chance constraints (Step 2.2 in Algorithm 2). For the medium and large examples, ν_{LS} is very small (a few seconds, see Table 10 below) compared to ν_{DE} (set to 10 minutes). The number of iterations at Step 3 is always smaller for LS than for DE. This is why LS is faster than DE in all instances.

Table 7 reports the first-stage solutions, first-stage costs, the average numbers of added or removed agents for R1, R2, R3, the optimal value of the respective stochastic programs, and the out-of-sample costs. As for the smaller call centers, we observe that in all three cases, LS generally gives a slightly lower first-stage cost than DE. The costs for R1 and R2 are larger than for R3, and we also obtain a larger first-stage cost when the SL targets are higher. Moreover, the average numbers of added or removed agents is smaller for R1 and R2 than for R3.

5.5 Consistent Values of Returned Solutions

Given that our approach is partly heuristic, there is always a risk that once in a while, it returns a poor solution due to a bad cut. To have an idea of how often this may happen, we made further experiments in which we ran Algorithm 2 several times independently, and compared the out-of-sample costs of the returned solutions. To illustrate the type of results we found, Table 8 reports the out-of-sample costs over 10 independent runs, for the medium and large call center

		Cost		Step 3 CPU time (seconds)		Total CPU time (seconds)		Out-of-sample cost	
case	N	LS	DE	LS	DE	LS	DE	LS	DE
R1	20	169.66	169.66	**285**	1030	**542**	1287	**175.39**	175.56
	50	**167.33**	167.38	**454**	1675	**1029**	2250	**176.13**	176.94
	70	**169.70**	169.73	**757**	1396	**1673**	2312	**175.20**	175.67
R2	20	**165.41**	165.48	**138**	1337	**395**	1594	**169.83**	170.31
	50	168.80	**168.78**	**249**	1617	**824**	2192	**168.80**	169.18
	70	166.82	166.82	**735**	1042	**1651**	1958	**169.76**	170.10
R3	20	164.38	164.38	**295**	829	**552**	1086	169.63	169.63
	50	165.76	**165.73**	**475**	2269	**1050**	2844	**169.51**	169.59
	70	166.51	**166.50**	**824**	2152	**1740**	3068	**169.42**	169.59

Table 5: Results for a one-hour period, for the medium-size example

			Cost		Step 3 CPU time (hours)		Total CPU time (hours)		Out-of-sample cost	
$(1-\delta_k, 1-\delta_0)$	Case	N	LS	DE	LS	DE	LS	DE	LS	DE
(0.80,0.85)	R1	20	**156.62**	156.75	**1.08**	5.20	**1.24**	5.36	159.77	**158.64**
		50	**157.13**	157.20	**2.44**	7.98	**2.73**	8.27	**158.59**	159.39
		70	**157.66**	158.65	**2.29**	9.90	**2.67**	10.28	**158.52**	158.75
	R2	20	**157.47**	158.31	**1.07**	5.24	**1.23**	5.40	**157.85**	158.50
		50	**156.54**	156.67	**1.98**	7.27	**2.27**	7.56	**156.98**	157.07
		70	**154.86**	156.11	**2.67**	6.34	**3.05**	6.72	**156.78**	156.96
	R3	20	**161.61**	164.37	**1.12**	4.89	**1.28**	5.05	**158.43**	157.58
		50	**154.65**	154.86	**2.37**	10.48	**2.66**	10.77	**158.03**	158.09
		70	**155.72**	159.18	**2.76**	9.01	**3.14**	9.39	**158.02**	159.43
(0.90,0.95)	R1	20	170.23	**170.18**	**0.04**	1.17	**0.27**	1.40	174.85	**174.83**
		50	**171.55**	171.68	**0.41**	1.93	**0.84**	2.36	**174.44**	174.66
		70	**172.44**	172.61	**0.82**	1.82	**1.40**	2.40	**174.57**	174.78
	R2	20	172.02	**171.95**	**0.29**	1.28	**0.52**	1.51	**173.82**	173.92
		50	**169.97**	170.01	**0.16**	0.96	**0.59**	1.39	**172.46**	172.57
		70	**169.63**	169.77	**0.58**	1.19	**1.16**	1.77	**172.68**	172.70
	R3	20	**168.37**	168.44	**0.40**	3.95	**0.63**	4.18	175.56	**175.07**
		50	**167.17**	167.64	**0.96**	5.19	**1.39**	5.62	**173.60**	174.79
		70	**167.38**	167.80	**1.19**	5.04	**1.77**	5.62	**171.02**	171.77

Table 6: Value of the best solution found for (**P4**) (Cost), CPU time for Step 3, total CPU time (in hours), and cost of retained solution estimated out-of-sample, for the large call center

$(1-\delta_k, 1-\delta_0)$	Models	Algo.	x^{T}	c^{T} x	Avg. r^+	Avg. r^-	Total cost	Out-of-sample cost
(0.80, 0.85)	R1	LS	(20, 0, 3, 15, 0, 0, 21, 12, 15, 7, 12, 5, 5, 6, 8)	156.05	1.44	2.92	157.66	158.52
		DE	(21, 0, 2, 16, 0, 4, 22, 13, 14, 10, 11, 2, 4, 5, 6)	157.50	1.35	3.50	158.65	158.75
	R2	LS	(20, 0, 5, 15, 0, 0, 22, 12, 13, 10, 11, 6, 5, 5, 5)	156.10	1.33	3.87	154.86	156.78
		DE	(19, 0, 5, 16, 0, 0, 23, 12, 11, 7, 12, 4, 8, 6, 7)	157.05	1.70	4.39	156.11	156.96
	R3	LS	(19, 0, 6, 17, 0, 0, 21, 13, 15, 7, 11, 5, 3, 2, 7)	151.85	4.94	3.50	155.72	158.02
		DE	(19, 0, 4, 16, 0, 0, 22, 16, 13, 7, 12, 0, 4, 6, 7)	151.95	4.58	2.94	159.18	159.43
(0.90, 0.95)	R1	LS	(21, 0, 7, 18, 0, 3, 23, 11, 15, 6, 11, 6, 7, 3, 10)	170.35	1.93	4.17	172.44	174.57
		DE	(20, 0, 8, 18, 0, 3, 23, 11, 16, 6, 12, 6, 6, 3, 10)	170.40	1.64	4.59	172.61	174.78
	R2	LS	(22, 0, 6, 17, 0, 1, 22, 14, 15, 7, 12, 3, 7, 4, 10)	168.00	2.79	4.83	169.63	172.68
		DE	(22, 0, 6, 17, 0, 1, 22, 13, 15, 7, 13, 3, 7, 4, 10)	168.00	2.76	4.60	169.77	172.70
	R3	LS	(19, 0, 6, 18, 0, 0, 22, 14, 15, 4, 13, 4, 4, 4, 10)	159.60	10.99	7.14	167.38	171.02
		DE	(20, 0, 6, 18, 0, 0, 22, 14, 15, 4, 13, 4, 5, 4, 10)	162.00	10.16	7.87	167.80	171.77

Table 7: First-stage solutions, first-stage costs and average number of added or removed agents for $N = 70$, for the large call center

examples, with the LS and DE methods, for the case R1 with $N = 70$ and targets $(0.80, 0.85)$. We see that there is very little variation between the values of the returned solutions.

5.6 A Comparison with Mean Value and Robust Approaches

Some may argue that the two-stage stochastic model considered in this paper is too much work, in particular with large-scale call centers, as the model involves a set of solutions instead of one solution as in one-stage models. To illustrate the worth of this additional complexity, we compare our approach with two alternatives, namely, the *mean value* problem and a *robust* model. The mean value (MV) problem or expected value problem is obtained by replacing all the random variables, in our case the the random factor ξ, by their expected values (Birge and Louveaux, 2011, Section 4.2). Denoting $\mathbb{E}[\xi]$ by $\bar{\xi}$, this leads to the problem

$$(\mathbf{MV}) \quad \begin{cases} \min\limits_{x} & c^{\mathrm{T}}x \\ \text{subject to} & \mathbb{P}[S_k(x) \geq l_k \mid \bar{\xi}] \geq 1-\delta_k, \quad k = 0, \ldots, K, \\ & x \geq 0 \text{ and integer.} \end{cases} \tag{16}$$

Let x^{MV} denote the solution of (16). The Value of the Stochastic Solution (VSS) is defined as VSS = EEV − RP where RP is the optimal value of **(P1)** and EEV $= c^T x^{\mathrm{MV}} + \mathbb{E}_\xi[Q(x^{\mathrm{MV}}, \xi)]$.

	Medium		Large	
Runs	LS	DE	LS	DE
1	188.60	188.91	158.52	158.75
2	188.87	188.67	158.52	158.75
3	188.60	188.91	158.52	158.75
4	188.60	188.91	158.62	158.85
5	188.60	189.12	158.52	158.85
6	188.60	189.12	158.52	158.61
7	188.45	188.91	158.52	158.75
8	188.87	188.67	158.52	158.75
9	188.60	188.91	158.62	158.75
10	188.87	188.91	158.52	158.75
Average	188.67	188.90	158.54	158.76

Table 8: Out-of-sample costs of returned solutions for 10 independent runs, for the medium and large call center examples, for case R1 with $N = 70$ and targets $(0.80, 0.85)$.

If x^{MV} is feasible for (**P1**), which is the case if $\bar{\xi}$ is a possible realization of ξ as (**P1**) is relatively complete, the VSS is positive.

Alternatively, we define a robust program by setting $r^+(\xi) = r^-(\xi) = 0$ for all $\xi \in \Xi$, leading to the one-stage problem

$$(\mathbf{RO}) \qquad \begin{cases} \min\limits_{x} & c^{\mathrm{T}}x \\ \text{subject to} & \mathbb{P}[S_k(x) \geq l_k \ \mid \xi] \geq 1 - \delta_k, \quad \forall \xi \in \Xi,\ k = 0, \ldots, K, \\ & x \geq 0 \text{ and integer.} \end{cases}$$

We denote by x^{RO} the solution of (**RO**) and $\mathrm{ERO} = c^T x^{\mathrm{RO}} + \mathbb{E}_\xi[Q(x^{\mathrm{RO}}, \xi)]$. Similarly to the VSS, we define the quantity $\mathrm{VRO} = \mathrm{ERO} - \mathrm{RP}$. Since (**P1**) is relatively complete, if $\bar{\xi} \in \Xi$, VRO is positive. In practice, we cannot impose that the SL constraint is satisfied for all ξ, and we approximate the robust problem a sample of N scenarios:

$$(\mathbf{RON}) \qquad \begin{cases} \min\limits_{x} & c^{\mathrm{T}}x \\ \text{subject to} & \mathbb{P}[S_k(x) \geq l_k \ \mid \xi_n] \geq 1 - \delta_k, \quad \forall n = 1, \ldots, N,\ k = 0, \ldots, K, \\ & x \geq 0 \text{ and integer.} \end{cases}$$

(**MV**) requires the evaluation of one chance constraint only, and therefore is the fastest to solve. (**RON**) requires all the chance constraints to be satisfied for all scenarios, and therefore is more expensive. As before, we approximate the chance constraints using simulation over 1000 days, and we use the same scenarios to construct the robust problem as in (**P4**). We use the same random numbers between all the programs, and the standard cutting-plane method to solve (**MV**) and (**RON**). We denote the solutions of the approximate mean value and robust problems by $\hat{x}^{\mathrm{MV}}$ and $\hat{x}^{\mathrm{RO}}$ respectively, and use these solutions to estimate the VSS et VRO, using an out-of-sample of 70 scenarios, kept the same for the three problems. The value RP is

Example	Targets	Model	Cost gap		Total CPU time (hours)		
			VSS	VRO	MV	RO	(**P4**)
Medium	(0.80, 0.85)	R1	4.69 (2.55 %)	7.80 (4.13 %)	0.13	0.64	1.91
		R2	12.86 (7.00 %)	9.05 (4.94 %)			3.68
		R3	17.07 (9.74 %)	4.05 (2.25 %)			3.93
	(0.90, 0.95)	R1	6.89 (3.60 %)	6.37 (3.26 %)	0.16	1.95	2.78
		R2	15.37 (8.08 %)	5.83 (3.07 %)			1.83
		R3	19.88 (11.02 %)	8.29 (4.95 %)			3.77
Large	(0.80, 0.85)	R1	8.79 (5.58 %)	6.23 (3.93 %)	0.01	0.77	2.67
		R2	10.28 (6.64 %)	5.89 (3.76 %)			3.05
		R3	13.01 (8.35 %)	8.22 (5.20 %)			3.14
	(0.90, 0.95)	R1	10.25 (5.94 %)	2.70 (1.55 %)	0.07	1.22	1.40
		R2	11.98 (7.06 %)	5.80 (3.36 %)			1.16
		R3	15.16 (9.06 %)	8.20 (4.75 %)			1.77

Table 9: Gaps and CPU times for the Mean Value and Robust approaches, for the medium and large examples

estimated using our LS approach, noting that the costs given by the DE are also quite similar.

We compute the gaps for the three instances of 70 scenarios with two sets of targets (0.80, 0.85) and (0.90, 0.95), as in the previous sections. Table 9 reports the CPU times for solving the mean value, robust and two-stage problems, and the gaps in nominal value as well as in relative value. The CPU times reported for solving the two-stage instances include the times for the initialization step, so they correspond to the "total CPU time" in Tables 3 and 6. The gaps are significant, while smaller for the robust model, suggesting that for the considered model, it is more interesting to staff against the worst case and pay a penalty to remove agents than to consider the average scenario. On the other hand, the CPU times for solving the robust model are slightly less than those required by the LS approach, except for large instances with targets (0.80, 0.85) for which the robust model is solved three times faster approximately. As expected, the computing time reduction is impressive for the mean value problem, as only one scenario is considered, but the VSS shows that it leads to a much more costly solution.

5.7 A Comparison of the Single-cut and Multi-cut LS Approaches

We provide a brief comparison of the performance of the multi-cut and single-cut L-shaped approaches on our three call center examples. While the multi-cut approach usually decreases the number of master iterations, i.e., the number of master problems to solve, the generation of many cuts can however significantly slow down the master problem solving, and some authors, e.g., Bodur and Luedtke (2017), have obtained better results using a single cut approach. In our case, the second-stage problems are however expensive to solve due to the presence of chance-constraints, approximated by simulation, so we expect that the multi-cut L-shaped algorithm (with $L = N$) will perform better. We try the three cost structures R1, R2, and R3 for the recourse. We also set a limit of 300 iterations per call of Algorithm 1. Table 10 reports the average number of iterations and the average total CPU times in Algorithm 1 per call to this

algorithm, with each of the two LS approaches. Again, the smallest numbers are in bold. The symbol "–" indicates that the corresponding approach failed to converge within 300 iterations. We find that for all call center sizes, the multi-cut approach requires fewer iterations. The CPU time is slightly larger with the multi-cut for the small call center (and also increases with the number N of scenarios, because there are then more constraints in the master problem), but becomes much smaller than for the single-cut when the size of the model increases. For the largest model, the single-cut approach fails to converge within 300 iterations in all instances, while the multi-cut converges in about 20 to 60 iterations, and the average CPU times are reasonable (20 to 200 seconds). The results show the superiority of the multi-cut approach for our simulation-based decomposition algorithm, in particular for large instances.

Case			Small call center			Medium call center			Large call center		
	# scenarios		300	600	800	20	50	70	20	50	70
R1	single-cut	# iterations	7.4	6.6	6.6	71.2	111.5	99.6	–	–	–
		CPU time (s)	**3.6**	**6.5**	**8.9**	59.5	98.9	190.0	–	–	–
	multi-cut	# iterations	4.7	4.4	3.9	26.2	26.2	24.0	54.2	26.6	23.6
		CPU time (s)	4.3	17.9	20.2	**17.7**	**21.1**	**29.5**	**156.6**	**31.6**	**29.7**
R2	single-cut	# iterations	9.1	8.5	9.3	79.2	82.5	98.4	-	-	-
		CPU time (s)	**7.5**	**15.7**	**20.4**	65.2	95.6	181.3	-	-	-
	multi-cut	# iterations	8.1	7.9	7.5	25.3	24.5	23.7	62.2	55.6	58.1
		CPU time (s)	11.7	42.3	82.4	**16.3**	**21.0**	**27.9**	**70.1**	**65.3**	**72.3**
R3	single-cut	# iterations	8.2	9.4	9.1	58.8	72.3	70.2	-	-	-
		CPU time (s)	**9.6**	**16.3**	**26.6**	51.2	79.3	145.2	-	-	-
	multi-cut	# iterations	7.5	8.1	8.9	19.3	20.1	18.9	43.4	38.2	35.7
		CPU time (s)	11.9	48.2	98.7	**13.3**	**17.2**	**21.2**	**53.3**	**45.3**	**56.2**

Table 10: Comparison of the single-cut and multi-cut approaches

6 Conclusion

We have proposed and tested a simulation-based SAA method combined with a decomposition algorithm for staffing optimization under arrival rate uncertainty. The problem is formulated as a two-stage stochastic program with integer recourse and chance constraints on the service levels at the second stage. While such a representation improves the model realism, it requires expensive simulations for each arrival rate scenario. Our proposed approach, based on a Benders decomposition with strengthened cuts, nevertheless provides good solutions with a reasonable computing effort. Our numerical results, based on call center models of three different sizes, show that the decomposition method outperforms a direct approach that does not use decomposition to solve the approximating MIP problem, especially for the large call center example.

These results open several interesting directions for future research, e.g., the extension of the method from staffing to scheduling problems. The proposed methodology might also be useful for other similar workforce management problems, such as staffing and scheduling in hospitals, clinics, and retail stores, for example, and especially for applications in which the constraints are constructed based on complex queuing models, for which the performance needs to be approximated by simulation.

Acknowledgment

This work has been supported by a Canada Research Chair, an Inria International Chair, and a Hydro-Québec research grant to P. L'Ecuyer, by NSERC Discovery Grants to F. Bastin and P. L'Ecuyer, and by scholarships from the CIRRELT, DIRO and Université de Montréal to T.A. Ta. We benefited from valuable discussions with Tien Mai from Singapore Management University.

References

J. Atlason, M. A. Epelman, and S. G. Henderson. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research*, 127:333–358, 2004.

J. Atlason, M. A. Epelman, and S. G. Henderson. Optimizing call center staffing using simulation and analytic center cutting plane methods. *Management Science*, 54(2):295–309, 2008.

A. N. Avramidis, A. Deslauriers, and P. L'Ecuyer. Modeling daily arrivals to a telephone call center. *Management Science*, 50(7):896–908, 2004.

A. N. Avramidis, W. Chan, and P. L'Ecuyer. Staffing multi-skill call centers via search methods and a performance approximation. *IIE Transactions*, 41(6):483–497, 2009.

A. N. Avramidis, W. Chan, M. Gendreau, P. L'Ecuyer, and O. Pisacane. Optimizing daily agent scheduling in a multiskill call centers. *European Journal of Operational Research*, 200 (3):822–832, 2010.

A. Bassamboo, J. M. Harrison, and A. Zeevi. Design and control of a large call center: Asymptotic analysis of an LP-based method. *Operations Research*, 54(3):419–435, 2006. ISSN 0030-364X.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, New York, NY, USA, second edition, 2011.

M. Bodur and J. R. Luedtke. Mixed-integer rounding enhanced Benders decomposition for multiclass service-system staffing and scheduling with arrival rate uncertainty. *Management Science*, 63(7):2073–2091, 2017.

E. Buist and P. L'Ecuyer. A Java library for simulating contact centers. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the 2005 Winter Simulation Conference*, pages 556–565. IEEE Press, 2005.

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

W. Chan, G. Koole, and P. L'Ecuyer. Dynamic call center routing policies using call waiting and agent idle times. *Manufacturing & Service Operations Management*, 16(4):544–560, 2014.

W. Chan, T. A. Ta, P. L'Ecuyer, and F. Bastin. Two-stage chance-constrained staffing with agent recourse for multi-skill call centers. In *Proceedings of the 2016 Winter Simulation Conference*, pages 3189–3200, Piscataway, NJ, USA, 2016. IEEE Press.

N. Channouf, P. L'Ecuyer, A. Ingolfsson, and A. N. Avramidis. The application of forecasting techniques to modeling emergency medical system calls in Calgary, Alberta. *Health Care Management Science*, 10(1):25–45, 2007.

A. Charnes and W. W. Cooper. Chance-constrained programming. *Management Science*, 5: 73–79, 1959.

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

N. Gans, H. Shen, Y.-P. Zhou, N. Korolev, A. McCord, and H. Ristock. Parametric forecasting and stochastic programming models for call-center workforce scheduling. *Manufacturing and Service Operations Management*, 17(4):571–588, 2015.

L. V. Green, P. J. Kolesar, and J. Soares. Improving the SIPP approach for staffing service systems that have cyclic demands. *Operations Research*, 49(4):549–564, 2001.

L. V. Green, P. J. Kolesar, and J. Soares. An improved heuristic for staffing telephone call centers with limited operating hours. *Production and Operations Management*, 12:46–61, 2003.

I. Gurvich, J. Luedtke, and T. Tezcan. Staffing call centers with uncertain demand forecasts: A chance-constrained optimization approach. *Management Science*, 56(7):1093–1115, 2010.

J. M. Harrison and A. Zeevi. A method for staffing large call centers based on stochastic fluid models. *Manufacturing and Service Operations Management*, 7(1):20–36, 2005.

S. Helber and K. Henken. Profit-oriented shift scheduling of inbound contact centers with skills-based routing, impatient customers, and retrials. *OR Spectrum*, 32:109–134, 2010. ISSN 0171-6468.

R. Ibrahim, P. L'Ecuyer, N. Régnard, and H. Shen. On the modeling and forecasting of call center arrivals. In C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, editors, *Proceedings of the 2012 Winter Simulation Conference*, pages 256–267. IEEE Press, 2012.

R. Ibrahim, P. L'Ecuyer, H. Shen, and M. Thiongane. Inter-dependent, heterogeneous, and time-varying service-time distributions in call centers. *European Journal of Operational Research*, 250:480–492, 2016a.

R. Ibrahim, H. Ye, P. L'Ecuyer, and H. Shen. Modeling and forecasting call center arrivals: A literature study and a case study. *International Journal of Forecasting*, 32(3):865–874, 2016b.

O. Jouini, G. Koole, and A. Roubos. Performance indicators for call centers with impatient customers. *IIE Transactions*, 45(3):341–354, 2013.

G. Koole. *Call Center Optimization*. MG books, Amsterdam, 2013.

P. L'Ecuyer and E. Buist. Variance reduction in the simulation of call centers. In *Proceedings of the 2006 Winter Simulation Conference*, pages 604–613. IEEE Press, 2006.

S. Liao, C. van Delft, G. Koole, and O. Jouini. Staffing a call center with uncertain non-stationary arrival rate and flexibility. *OR Spectrum*, 34:691–721, 2012.

S. Liao, C. van Delft, and J. P. Vial. Distributionally robust workforce scheduling in call centers with uncertain arrival rates. *Optimization Methods and Software*, 28(3):501–522, 2013.

G. L. Nemhauser and L. A. Wolsey. A recursive procedure to generate all cuts for 0–1 mixed integer programs. *Mathematical Programming*, 46(1-3):379–390, 1990.

B. Oreshkin, N. Régnard, and P. L'Ecuyer. Rate-based daily arrival process models with application to call centers. *Operations Research*, 64(2):510–527, 2016.

A. Pot, S. Bhulai, and G. Koole. A simple staffing method for multi-skill call centers. *Manufacturing and Service Operations Management*, 10:421–428, 2008.

R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.

T. R. Robbins and T. P. Harrison. A stochastic programming model for scheduling call centers with global service level agreements. *European Journal of Operational Research*, 207(3):1608–1619, 2010.

S. G. Steckley, S. G. Henderson, and V. Mehrotra. Forecast errors in service systems. *Probability in the Engineering and Informational Sciences*, 23(2):305–332, 2009.

T. A. Ta, T. Mai, F. Bastin, and P. L'Ecuyer. On a multi-stage discrete stochastic optimization problem with stochastic constraints and nested sampling. *Mathematical Programming*, 2020. URL https://doi.org/10.1007/s10107-020-01518-w.

R. M. van Slyke and R. Wets. L-Shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

R. B. Wallace and W. Whitt. A staffing algorithm for call centers with skill-based routing. *Manufacturing and Service Operations Management*, 7(4):276–294, 2005.

W. Whitt. Dynamic staffing in a telephone call center aiming to immediately answer all calls. *Operations Research Letters*, 24(5):205–212, 1999.