MANUFACTURING & SERVICE OPERATIONS MANAGEMENT
Vol. 00, No. 0, Xxxxx 0000, pp. 000–000
ISSN 1523-4614 | EISSN 1526-5498 | 00 | 0000 | 0001

INFORMS
DOI 10.1287/xxxx.0000.0000

Dynamic Call Center Routing Policies Using Call Waiting and Agent Idle Times

Wyean Chan
DIRO, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal (Québec), H3C 3J7, CANADA,
chanwyea@iro.umontreal.ca

Ger Koole
Department of Mathematics, VU University Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands,
ger.koole@vu.nl

Pierre L'Ecuyer
DIRO, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal (Québec), H3C 3J7, CANADA,
lecuyer@iro.umontreal.ca

We study call routing policies for call centers with multiple call types and multiple agent groups. We introduce new weight-based routing policies where each pair (call type, agent group) is given a matching priority defined as an affine combination of the longest waiting time for that call type and the longest idle time or the number of idle agents in that agent group. The coefficients in this combination are parameters to be optimized. This type of policy is more flexible than traditional ones found in practice, and it performs better in many situations. We consider objective functions that account for the service levels, the abandonment ratios and the fairness of occupancy across agent groups. We select the parameters of all considered policies via simulation-based optimization heuristics. This only requires the availability of a simulation model of the call center, which can be much more detailed and realistic than the models used elsewhere in the literature to study the optimality of certain types of routing rules. We offer a first numerical study of realistic routing rules that takes into account the complexity of real-life call centers.

Key words: multi-skill call centers; contact centers; call routing policies; simulation; stochastic optimization
History:

1. Introduction

Call centers have a significant economic importance in today's world, as explained in Gans et al. (2003), Akşin et al. (2007), and Koole (2013), for example. Managers face complex optimization problems where the goal is to meet various constraints on quality of service (QoS) at the least possible cost, or optimize a given *performance measure* (PM) for a given budget. Most of these costs are actually the salaries of the people who answer the calls.

We consider an inbound multi-skill call center, where arriving calls initiated by the customers are categorized by the type of service that they require, the *call type*. *Agents*, or *customer sale representatives*, are partitioned into *agent groups*, where all agents in each group are trained to answer the same subset of call types, called their *skill set*. It is not economical and sometimes even impossible to train all agents to have all skills; in practice most agents have only a few skills.

Managing a multi-skill call center means, among other things, to decide how many agents of each skill set should be assigned to work in each time period (the *staffing* problem), establish admissible work schedules for agents with the right skill sets to cover those staffing requirements (the *scheduling* problem), and select rules (*routing* policies) to match the calls to agents with the right skills. A routing choice arises each time a new call arrives and there are idle agents able to serve it: "Which agent should serve this call?" A decision also has to be taken when an agent becomes free and there are calls waiting to be served: "Which waiting call should this agent serve next?" Ideally, the skill sets, staffing, scheduling, and routing should all be optimized simultaneously, but

for large real-life call centers, this leads to excessively difficult optimization problems. In practice, much effort is often put into the staffing and scheduling of agents, while the skill sets and routing policy are often selected ad hoc rather than being systematically optimized. However, the routing policy usually plays an important role in the performance of a multi-skill call center. In this paper, our focus is on routing policies and their optimization for fixed staffing levels and skill sets.

The most rudimentary routing rule is to assign the longest waiting call to the longest idle agent that has the right skill to serve it. This is a combination of the well-known *first-come first-served* (FCFS) and *longest idle server first* (LISF) rules. With the emergence of multi-skill call centers, more complex routing policies have been designed. A popular strategy is to assign different routing priorities between agents and calls. An agent will serve a call of higher priority even if some calls of lower priorities have waited longer. Sometimes, the skill sets of certain agents are restricted manually (in an ad hoc fashion) in real-time for certain durations. We will show in our examples that better performance can often be achieved by allowing more flexibility, using the policies introduced in this paper.

Optimization of the routing policy is a control problem that could be solved in principle with dynamic programming (DP). This approach has been studied for call centers with few call types, and under simplifying assumptions of the model such as Poisson arrivals and exponential service times (Koole and Pot 2005, Koole et al. 2009). These call centers are generally modeled as continuous-time Markov chains. But for real-life multi-skill call centers, the number of states in the DP model is usually much too large (it grows exponentially with the number of call types and groups) for an optimal solution to be practically computable. Furthermore, certain types of PMs are hard to approximate using DP. For example, if the PM depends on all waiting times, then the state in the DP formulation must keep track of all those waiting times. Finally, it would be difficult to implement the optimal control policies in the routers of real-life call centers, because they are usually much too complex.

For large call centers, routing algorithms that are asymptotically optimal in a heavy-traffic regime, when agent occupancies converge to 100%, have been proposed and studied by various authors. The asymptotic optimality is proven under simplifying assumptions, for example, that the service time distribution depends only on the agent group and not on the call type, or that the call center has a single call type, and for a given (fixed) staffing. Consequently, the asymptotically optimal policy often has a simple form that ignores certain features of the real system. For example, see van Mieghem (1995, 2003), Harrison and Zeevi (2004), Mandelbaum and Stolyar (2004), Atar (2005), Atar et al. (2010), Bell and Williams (2001, 2005), Milner and Olsen (2008), Gurvich and Whitt (2009, 2010), Armony and Ward (2010, 2013), Tezcan and Dai (2010). Other studies propose frameworks that simultaneously optimize the routing and the scheduling or staffing of a call center. Examples are Wallace and Whitt (2005), Sisselman and Whitt (2007) and Bassamboo et al. (2006). Most of these studies base their results on heavy-traffic limits. Two exceptions are Wallace and Whitt (2005) and Sisselman and Whitt (2007), whose results are based mainly on simulation. Most policies are also work-conserving, but there are some exceptions, such as Atar (2005), Bell and Williams (2001, 2005), and Bassamboo et al. (2006).

This large body of research work provides satisfactory routing rules for many practical settings, but not all. Typical situations where they can fail are when certain call types have low volumes, or when only a fraction of calls are counted or penalized (for example, if the performance measure is a function of the number of calls answered within a given time limit). Such situations are common. Our study focuses on call centers of small and medium size, whose behavior may differ from those obtained in heavy-traffic regimes, and for which *non-work-conserving* policies (where an agent can stay idle even when there is work available) can perform better, because they may permit one to save some agents for calls that are more important or that those agents can better handle. Example 1 below shows the usefulness of non-work-conserving policies in certain situations. Xu et al. (1992) give similar examples.

Example 1. Consider an N-model call center (see Figure 1 in Section 5) with 2 call types, where group 1 can only serve calls of type 1 and group 2 can serve all calls. There are 10 agents in group 1 and 3 agents in group 2. Arrivals are from two independent Poisson processes, with rates of 3.0 and 0.75 calls per minute for call types 1 and 2, respectively. This represents a common situation. Service times are exponential (and independent) with means 3 and 2 minutes for call types 1 and 2, respectively. Suppose the service level target is to answer 70% of calls within 30 seconds of waiting time, for each call type. When calls are assigned to the longest waiting agent that can handle them (LISF), and agents are assigned to the longest waiting call they can handle (FCFS), the percentage answered within 30 seconds is (approximately) 78% and 40%; call type 2 is 30% below its target. As another possibility, if group 1 gives priority to call type 1, and group 2 gives priority to call type 2, the service levels reach 77% and 55%, respectively. A better result is obtained by making the last policy non-work-conserving, as follows: if there is only 1 free agent in group 2, this agent cannot answer any call of type 1 as long as this condition holds. This leads to service levels of 69% and 70%, showing the effectiveness of idleness in skill-based routing. Koole (2013) has an accompanying website where these results can be reproduced. □

Inspired in part by this example, we propose new dynamic routing policies called *weight-based routing* (WR) where the matching between a call type and an agent group is done according to the highest index rule, as follows. An index (or weight) function is defined, for each pair of call type and agent group that can handle it, as an affine function of the longest waiting time for that call type and the longest idle time (or the number of idle agents, in one variant) in that group, when there is at least one waiting call of that type and one idle agent in that group, otherwise the index is $-\infty$. The coefficients and constant terms in these affine functions are parameters that are optimized beforehand, e.g., by simulation-based heuristics. Whenever at least one pair (call type, agent group) has a positive index, the pair with the highest index is selected, and the agent that is currently idle for the longest time in that group immediately answers the oldest waiting call of that type. The number of parameters in these policies remains reasonable, so they are much easier to implement in a real router than the complicated optimal control policies that could possibly be obtained from dynamic programming.

These WR policies are related to generalized forms of the $c\mu$ rules (G$c\mu$), in which an agent always selects the call for which the rate of expected cost reduction is the largest (van Mieghem 1995). We also define a generalized form of work-conserving $c\mu$ rule named LG$c\mu$, in which an agent that becomes idle immediately takes the call with largest index among those she can handle (if any), where the index is defined as the service rate of the agent for that call multiplied by an affine function of the waiting time. Likewise, when a new call arrives, it is handled by the agent with largest index that can handle that call type (if any), where the index is defined as the service rate of the agent for that call multiplied by an affine function of the time since this agent is idle. The coefficients and constants in the affine functions are optimized beforehand, just like for WR.

An important difference between WR and LG$c\mu$ is that WR allows negative constant terms in the affine functions; this permits one to impose delays on waiting calls (by keeping agents idle) when the index is negative. Thus, WR has the flexibility of being non-work-conserving. Another difference is that WR allows different parameters for each pair (call type, agent group), which also increases the flexibility. On the other hand, the optimization generally requires more computing time when there are more parameters to optimize. For this reason, we explore variants of WR with fewer parameters. We also find that the optimization time is much smaller when we start already from a reasonable suboptimal solution or from a good solution to a slightly different model. This can be exploited to speed up the computations in practice, for both WR and LG$c\mu$, for example when optimizing the routing rules of successive periods in a day, when the neighbor periods have similar arrival patterns and staffing, or to update the routing parameters when arrival-rate forecasts are updated.

In contrast to the studies described earlier, which are mostly based on heavy-traffic approximations, we only assume in our study the availability of a detailed simulation model of the call center and we use simulation-based optimization heuristics that allow more diversified objective functions than the approximations. Simulation is the only way to study richer objectives and controls that depend on the waiting and agent availability times in realistic systems. We consider several types of routing policies, for which we optimize the parameters for a given staffing and selected objective functions, and we compare their performance. The call center model itself can incorporate arbitrary probability distributions, so there are much less limitations on the model than with analytical approaches. What we present here is the first study of realistic routing rules that takes into account the complexity of real-life call centers. The optimization problem we consider is formulated in terms of penalty functions only, rather than imposing hard constraints on long-term expectations. Penalties can be imposed as functions of standard measures such as abandonment ratios, average waiting times, and service level (the fraction of calls answered within a given waiting time limit). They can also account for other important considerations such as agent occupancy and fairness between different agent groups, for example. We find that our WR policies are competitive with all other routing policies considered in this study, in all examples that we have tried, even in situations where G$c\mu$ rules are proved to be asymptotically optimal. And they sometimes provide much better performance, particularly when the penalties are on service levels.

The remainder is organized as follows. In Section 2, we define our model and the performance measures that we look at. In Section 3, we define the routing rules that we consider, we introduce our new WR policy as well as an LG$c\mu$ policy, and we discuss the choice of weights in those policies. In Section 4, we summarize the heuristic algorithm that we have used to optimize the routing policy parameters. In Section 5, we report our numerical experiments, first for systems with two or three call types such as the X and W models, then for a larger model. We use the W-model to compare the robustness of the policies and to illustrate how the parameter optimization for WR, LG$c\mu$, and other policies can be made much faster by starting from a good solution to a similar model (where the model parameters did not change too much). We also give an example where a G$c\mu$ policy is known to be asymptotically optimal, and show how closely WR can compete with this policy when its parameters are optimized by our heuristic. This is representative of our experiments for this type of situation. A conclusion follows in Section 6.

2. The model

We consider a call center where arriving calls are categorized in K types, named 1 to K. These calls are served by agents divided into G groups. Group $g \in \{1, \ldots, G\}$ is staffed with y_g agents, each having the skill set $\mathcal{S}_g \subseteq \{1, \ldots, K\}$, of cardinality $h_g = |\mathcal{S}_g|$, which gives the subset of call types that this agent can serve. An agent with very few skills (like one or two) is called a *specialist* and an agent with many skills, a *generalist*. We assume that all agents in the same group are homogeneous. Let $\mathcal{I}_k = \{g : k \in \mathcal{S}_g\}$, the set of agent groups that can serve call type k. The number of agents and the sizes of the agent groups are fixed a priori. We do not assume any particular arrival process, service time distribution or patience time distribution, we only need to be able to simulate them. If a call cannot be served immediately on arrival, it is put at the back of a waiting queue associated with its call type. We assume that each customer requires only one type of service and exits the system either at the completion of service or when its waiting time exceeds its (random) *patience time*. The work is non-preemptive: once an agent starts serving a call, there can be no interruption until service completion. The assignment between a call and an agent is decided by the router, according to a routing policy such as those described in Section 3. When we optimize, all model parameters are fixed, except for the routing policy.

2.1. Performance Measures (PM)

The objective functions we considered are defined in terms of the following PMs: the service level, the abandonment ratio and the agent occupancy. These PMs can be measured per time period, per call type, per agent group, or in aggregated form (for example, globally for all call types). They are often used in practice. Our method also allows the use of other PMs, such as the average waiting time, the average excess waiting time above a given threshold (Koole 2013), and the matching rates between selected call type/agent group pairs (Sisselman and Whitt 2007). Henceforth, π denotes a routing policy and $\mathbb{E}$ the mathematical expectation operator.

The first PM, the *service level* (SL), is defined as the fraction of calls answered within a time threshold τ called the *acceptable waiting time* (AWT), in the long run. That is, the SL over a given time period, under policy π, is

$$S(\pi,\tau) = \frac{\mathbb{E}[X(\pi,\tau)]}{\mathbb{E}[N - N_{\mathrm{A}}(\pi,\tau)]}, \tag{1}$$

where $X(\pi,\tau)$ is the number of served calls that have waited no more than τ, N is the total number of call arrivals during the period, and $N_{\mathrm{A}}(\pi,\tau)$ is the number of calls that abandoned after waiting less than τ. According to the renewal reward theorem, this ratio of expectations is the correct representation of the fraction of calls answered within the time limit in the long run, that is, over an infinite number of independent replicates of the (random) call center behavior over the given time period. It differs from the expectation of the ratio $X(\pi,\tau)/(N - N_{\mathrm{A}}(\pi,\tau))$, which would give more weight to calls that are on days with smaller arrival volumes. We used the SL definition (1) in our numerical experiments, but there are other possible definitions (see Jouini et al. (2013)). When these measures are for a given call type k, we add a subscript k in the notation; for example τ_k and $S_k(\pi,\tau_k)$ denote the AWT and the SL of call type k.

The second PM we consider is the *abandonment ratio*, defined as a ratio of expectations:

$$A(\pi) = \frac{\mathbb{E}[Z(\pi)]}{\mathbb{E}[N]}, \tag{2}$$

where $Z(\pi)$ is the number of calls that abandoned during the period considered. Again, we add a subscript k when the measure is for call type k.

The third measure is the *occupancy ratio* of agent groups. It is used to measure fairness between agent groups. We define the occupancy ratio of group g as

$$O_g(\pi) = \frac{1}{y_g T}\mathbb{E}\left[\int_0^T G_g(\pi,t)dt\right], \tag{3}$$

where T is the time horizon and $G_g(\pi,t)$ is the number of busy agents in group g at time t.

Although the measures considered here are long-term averages, it is also important in some contexts to take into account the variability of the PM across time periods, for example across days or across hours (e.g., due to the uncertainty in arrival rates). For instance, even though the SL for $\tau = 20$ seconds is above 80% in the long run, it may happen that the (random) SL over a given day, defined by removing the expectations in (1), is below 80% for more than half of the days, or could be below 40% on some days. One may then want to consider the probability that this random SL is above a given value, or some more general characterization of its distribution that accounts for the variability, in the PMs we optimize (Liao et al. 2010, Gurvich et al. 2010). Our simulation-based methodology would apply to those types of PMs as well. But here, we focus on the long-term measures defined earlier. The routing parameters could also be re-optimized adaptively during the day to account for new information.

2.2. Objective Functions

In the optimization problems considered here, instead of imposing hard constraints on the PMs as often done (Atlason et al. 2004, Ceşik and L'Ecuyer 2008, Avramidis et al. 2010), we use penalty costs. The objective is to find routing-rule parameters that minimize the sum of these penalty costs. This makes a comparison between policies easier and avoids unfeasible problems. The penalty costs are (truncated) polynomial functions of the PMs. We express them as functions of the routing policy π. They are defined as follows.

1. The penalty cost for violating the SL targets is

$$F_{\mathrm{S}}(\pi) = \sum_{k=1}^{K} c_{\mathrm{S},k} \max(t_k - S_k(\pi, \tau_k), 0)^{e_{\mathrm{S},k}}, \tag{4}$$

where for each call type k, τ_k is the AWT, S_k is the SL, t_k is the corresponding target, $e_{\mathrm{S},k} \geq 0$ is the polynomial degree, and $c_{\mathrm{S},k}$ is the penalty weight. In practice, missing the SL target by a small percentage is often deemed acceptable, while missing it by a large percentage is highly unacceptable. This can motivate the choice of exponents $e_{\mathrm{S},k}$ larger than 1. Fairness between call types can also be an important criteria; for example, it is usually preferable to have two call types at p% below their SL targets than to have one call type right on target and the other call type $2p$% below its target. By taking large values of $e_{\mathrm{S},k}$, one can penalize this form of unfairness between SL violations across call types. The weights $c_{\mathrm{S},k}$ may be selected to take into account the volumes of the different call types (for example, $c_{\mathrm{S},k}$ could be proportional to the fraction of calls that are of type k) and they may also reflect other considerations (for example, certain call types deemed more important than others can have a larger $c_{\mathrm{S},k}$, or a larger t_k, or both). We could also include a term for the aggregate SL target in the sum.

2. The penalty for abandonments is

$$F_{\mathrm{A}}(\pi) = \sum_{k=1}^{K} c_{\mathrm{A},k} \max(A_k(\pi) - u_{\mathrm{A},k}, 0)^{e_{\mathrm{A},k}}, \tag{5}$$

where A_k is the abandonment ratio, $u_{\mathrm{A},k}$ is the abandonment threshold, $e_{\mathrm{A},k} \geq 0$, and $c_{\mathrm{A},k} \geq 0$, for each call type k. More important call types may be given larger $c_{\mathrm{A},k}$ or smaller $u_{\mathrm{A},k}$.

3. The penalty for unfairness between agent group occupancies is

$$F_{\mathrm{O}}(\pi) = \sum_{g=1}^{G} c_{\mathrm{O},g} \left| O_g(\pi) - \bar{O} \right|^{e_{\mathrm{O},g}}, \tag{6}$$

where for each group g, O_g is the occupancy ratio, $\bar{O} = (1/G)\sum_{g=1}^{G} O_g(\pi)$ is the average of group occupancies, $e_{\mathrm{O},g} \geq 0$, and $c_{\mathrm{O},g} \geq 0$. An alternative definition of $\bar{O}$ would weigh the groups by their sizes y_g. The $c_{\mathrm{O},g}$ can also account (or not) for the group sizes. Instead of penalizing unfairness as we do here, there could be a target occupancy ratio for each group, which would replace $\bar{O}$ in the formula. This type of unfairness penalty function could also be considered for the PMs of call types.

Our overall objective function is the sum of the penalty functions in (4) to (6):

$$F_{\mathrm{SAO}}(\pi) = F_{\mathrm{S}}(\pi) + F_{\mathrm{A}}(\pi) + F_{\mathrm{O}}(\pi), \tag{7}$$

The weights $c_{\mathrm{S},k}$, $c_{\mathrm{A},k}$ and $c_{\mathrm{O},g}$ can be selected to give more or less importance to any of the three terms in (7). A term can be removed by taking the weights equal to 0. To simplify the notation, we will omit the subscript $*$ in F_{SAO} if $c_{*,j} = 0$ for all j, where j is either the call type or agent group. For example, we denote the objective function (7) by $F_{\mathrm{SA}}(\pi)$ when $c_{\mathrm{O},g} = 0$ for all g.

We emphasize that the simulation-based stochastic optimization method used in this paper is independent of the choice of objective function. That is, (7) can be replaced by a more general penalty function. This function is taken as a *black box* function by the optimization method: we only assume the possibility of performing (noisy) function evaluations (via simulation) for any considered policy π. A major advantage of this black-box scheme is that the method usually requires minimal modifications when changing the objective function.

3. Routing Rules

We describe the routing policies considered in this paper. Most of them are based on routing rules found in industry. We also consider generalized forms of $c\mu$ rules and introduce our WR policies. All policies considered here satisfy two basic fairness rules: (1) for any given call type, the calls are always served in FCFS order, and (2) for agents of the same group, the longest idle agent is always the next one to work (LISF). These two rules are in force in all call centers that we have encountered. However, it is acceptable for a call to be served before a call of another type that has waited longer, and for an agent who has been idle for a shorter time than another one to be assigned his next call earlier if they are in different groups. The short acronyms given in the following headers will be used later, when we present numerical results.

3.1. Global FCFS (G)

The simplest routing policy is when all call types have equal priorities. This policy can be implemented via a single FCFS queue, where an idle agent would scan the queue from the head and pick the first call that it can serve. When a call arrives, it will choose the idle agent that finished his last service first (LISF) among those that can serve it, independently of the group.

3.2. Priority Routing (P)

This policy, also called *overflow routing*, is available in routing equipments from vendors such as Cisco and Avaya. The call-to-agent and agent-to-call assignments are decided by priority lists, as follows. When an agent becomes idle and searches for the next waiting call to serve, a *group-to-type* list for its agent group determines the order in which the queues of the different call types are examined. Similarly, when a new call arrives and searches for an idle agent to answer it, a *type-to-group* priority list for that call type determines the order in which the agent groups are searched.

The group-to-type priority list for group g is $\mathcal{L}_g = (\mathcal{L}_g^{(1)}, \ldots, \mathcal{L}_g^{(m_g)})$, where $m_g \le h_g = |\mathcal{S}_g|$ is the number of priority levels, and $\mathcal{L}_g^{(1)}, \ldots, \mathcal{L}_g^{(m_g)}$ form a partition of $\mathcal{S}_g$. Each $\mathcal{L}_g^{(i)}$ contains call types having the same priority, served in FCFS order by agents of the group g. When an agent of group g becomes idle, he first scans the waiting queues of all call types in $\mathcal{L}_g^{(1)}$, and serves the call whose waiting time is the largest. If all these queues are empty, he continues with the call types in $\mathcal{L}_g^{(2)}$, and so on. Similarly, the type-to-group priority list for call type k is $\mathcal{G}_k = (\mathcal{G}_k^{(1)}, \ldots, \mathcal{G}_k^{(\ell_k)})$, where ℓ_k is the number of priority levels, and $\mathcal{G}_k^{(1)}, \ldots, \mathcal{G}_k^{(\ell_k)}$ form a partition of $\mathcal{I}_k = \{g : k \in \mathcal{S}_g\}$. An arriving call looks for an idle agent whose group is in $\mathcal{G}_k^{(1)}$, and picks one using LISF order. If none is idle, it tries the groups in $\mathcal{G}_k^{(2)}$, and so on. Wallace and Whitt (2005) and Ceşik and L'Ecuyer (2008) have used similar priority lists, but with the additional restriction that equal priorities are disallowed for call selection. For the special case where all priorities are equal (all the lists have a single subset), we recover the policy G.

3.3. Priorities with Delays (PD)

One refinement of policies with priority lists consists in adding delay conditions based on the waiting time. For each pair of group g and call type $k \in \mathcal{S}_g$, we select a *time delay* $d_{k,g} \ge 0$. An agent of group g can answer a call of type k only when this call has waited at least $d_{k,g}$. These

delays increase the flexibility and can improve the performance over policy P, by increasing the match rate between the agents and the call types from their primary skill. When the SL is involved in the objective function, the time delay would normally be set below the AWT τ_k, so the call can still have a good service if answered after the delay expires. But the idle agent may also answer another call with smaller delay (that better matches his skills) in the meantime. There are also situations where it can be optimal (depending on the objective function) to set a time delay larger than the AWT.

3.4. Priorities with Idle Agent Thresholds (PT)

Real-life call center managers often restrict temporarily the skill set of one or more agents. They take away a skill with a good SL, hoping that the agent will handle more calls of a type that has a bad SL. This idea is captured in a more dynamic way by the following modification of the policy with priority lists, which uses *priorities with idle agent thresholds*. Following Gans and Zhou (2003), we use thresholds that are not necessarily integers. For each pair of group g and call type $k \in \mathcal{S}_g$, we select a real-valued threshold $m_{k,g} \geq 0$ on the number of idle agents. If an idle agent of group g is supposed to answer a call of type k according to the priority lists, let i_g be the number of idle agents of type g, and let $\delta_{k,g} = m_{k,g} - \lfloor m_{k,g} \rfloor$ be the fractional part of $m_{k,g}$. If $i_g > \lceil m_{k,g} \rceil$, the agent takes the call. If $i_g = \lceil m_{k,g} \rceil$, the agent takes the call with probability $1 - \delta_{k,g}$. Otherwise, the agent does not take this call and we have to go further down the priority list. Because of the non-preemption assumption, this condition has no effect on the agents that are already serving calls: A working agent of group g will not stop serving a call of type k even if the number of idle agents in group g no longer exceeds the threshold.

3.5. Priorities with Delays and Idle Agent Thresholds (PDT)

By combining the PD and PT policies, we obtain a further generalization where we have priority lists, time delays, and idle agent thresholds. In principle, because of the increased flexibility, this can always provide equal or better performance than all the policies discussed previously. But the larger number of parameters also makes their optimization more difficult, so a heuristic search may (and often does) return a worst routing policy. Optimizing together the priority lists (a combinatorial problem) with the delay and idle agent threshold parameters, can be very hard.

3.6. A linear adapted generalized $c\mu$ rule (LG$c\mu$)

Suppose we want to minimize

$$\sum_{k=1}^{K} \sum_{n=1}^{N_k} C_k(w_{k,n}), \tag{8}$$

where N_k is the number of calls of type k, $w_{k,n}$ is the waiting time of the n-th call of type k, and the C_k are convex non-decreasing cost functions with derivatives C'_k. When an agent of group g becomes idle, take the waiting call of type k^* that maximizes the expected cost derivative:

$$k^* = \arg\max_{k \in \mathcal{S}_g} C'_k(w_k)\mu_{k,g}, \tag{9}$$

where w_k is the wait time so far for the oldest waiting call of type k and $\mu_{k,g}$ is the service rate (inverse of mean service time) of a call of type k by an agent of group g. This policy is called a *generalized $c\mu$ (G$c\mu$) rule*, and has been proved *asymptotically optimal* for the cost function (8), under certain *heavy-traffic* regimes, e.g., by van Mieghem (1995) for the case of a single server that can serve all call types (his proof does not require strict convexity), and by Mandelbaum and Stolyar (2004) in a more general setting with an arbitrary (fixed) number of servers and strictly convex increasing functions C_k. Atar et al. (2010) considered a model with abandonments, for a single skill group, and showed asymptotic optimality of a G$c\mu$ rule where $C'_k(w_k)$ is divided by

the abandonment rate. In van Mieghem (2003), a modified G$c\mu$ policy was proved asymptotically optimal in heavy-traffic for the maximum SL over the call types k, again for a single server. The attractiveness of this policy, besides its simplicity, is that whenever G$c\mu$ is optimal, it is also robust against changes in arrival rates, because it does not depend on the arrival rates. In all these cases, the asymptotic optimality holds only for specific types of objective functions and under simplifying conditions. These policies do not always perform well in realistic settings. One could nevertheless use (heuristically) adapted versions of the G$c\mu$ rule under realistic settings, even though they are not necessarily optimal.

Since our objectives (4) to (7) are not structured and decomposable as (8), we cannot use the G$c\mu$ rule (9) in that form. Here we consider a variant for which we take C'_k in (9) as a linear function of the form $C'_k(w) = a_k + b_k w$, hence the name LG$c\mu$ (Linear G$c\mu$), and we optimize the parameters $a_k \geq 0$ and $b_k \geq 0$ for the given objective function. Our call selection function (9) becomes:

$$k^* = \arg\max_{k \in \mathcal{S}_g} (a_k + b_k w_k)\mu_{k,g}.$$

Note that if $C_k(w) = a_k w$, so (8) is a weighted sum of waiting times, then $C'_k(w) = a_k$, which is covered by our linear setting.

The G$c\mu$ policy does not consider the choice of agent when a new call arrives. In LG$c\mu$, we use another linear function to select an idle agent when a call of type k arrives:

$$g^* = \arg\max_{g \in \mathcal{I}_k} (e_g + f_g v_g)\mu_{k,g},$$

where v_g is the largest idle time in group g. In case of equality, we pick the group at random, with probabilities proportional to the number of idle agents. Considering the agent idle times in LG$c\mu$ can help balance agent group occupancies.

If we choose $b_k = 0$ and $a_k = 1$ for all k, and $f_g = 0$ and $e_g = 1$ for all g, the LG$c\mu$ rule becomes the *Fastest-Server-First* (FSF) rule. Tezcan and Dai (2010) have shown that this rule is asymptotically optimal for the special case of an N-model (two call types and two agent groups, one can serve only one call type and the other can serve both), under a many-server asymptotic regime, under the assumptions of linear holding and abandonment costs, and service rates that are only agent-dependent (independent of the call type). In our experiments with this model in the Online Supplement, WR performs as well as LG$c\mu$.

All these G$c\mu$-type policies are *work-conserving*, which is too restrictive in some situations. They have the advantage of being simple and easy to implement, and tend to be robust to arrival rate uncertainties. In contrast to the original G$c\mu$ rule where the parameters are directly derived from C_k, we have to optimize $2(K+G)$ parameters in LG$c\mu$.

3.7. Weight-based Routing (WR)

We now introduce a routing policy based on weights, defined as affine functions of call waiting times and agent idle times. Each pair of group g and call type $k \in \mathcal{S}_g$ is given a *weight* $c_{k,g} \in \mathbb{R}$, which can be interpreted as an index of priority, defined as:

$$c_{k,g} = q_{k,g} + a_{k,g} w_k + b_{k,g} v_g, \tag{10}$$

where w_k is the longest waiting time of a call of type k currently in queue, v_g is the current longest idle time of an agent of group g, and the $q_{k,g} \in \mathbb{R}$, $a_{k,g} \geq 0$, and $b_{k,g} \geq 0$ are (fixed) real-valued parameters. Only the $q_{k,g}$'s are allowed to be negative. When there are either no idle agents in group g or no call of type k waiting in queue, we put $c_{k,g} = -\infty$. The $c_{k,g}$'s change continuously in time. Whenever there is at least one $c_{k,g} \geq 0$, the router selects a pair $(k^*, g^*) = \arg\max_{k,g} c_{k,g}$, and it assigns the waiting call of type k^* with the longest waiting time to the longest idle agent of group

g^*. When all $c_{k,g} < 0$, no call is assigned to an agent. A negative $q_{k,g}$ can be used to approximate a delay before an assignment (k, g) is made; that is, $a_{k,g}w_k + b_{k,g}v_g$ must exceed $-q_{k,g}$. This can be helpful in all situations where a delay is useful, in particular when the SL dominates the objective function.

When all the $q_{k,g}, a_{k,g}$ and $b_{k,g}$ are non-negative, there is no delay and the router can recompute and check the weights $c_{k,g}$ only on the event of a call arrival or a service completion. When $c_{k,g}$ is allowed to be negative, it must be recomputed more frequently. In our implementation, we update the $c_{k,g}$'s at every second whenever there are waiting calls being delayed and idle agents that can handle them.

There is a total of $3\sum_{g=1}^{G} h_g$ parameters to specify for this WR routing policy. We also consider the following variants of WR, with different definitions of $c_{k,g}$:

WR-sep: $c_{k,g} = q_{k,g} + a_k w_k + b_g v_g$

WR-sep2: $c_{k,g} = q_k + r_g + a_k w_k + b_g v_g$

WR-idnum: $c_{k,g} = q_{k,g} + a_{k,g} w_k + b_{k,g} i_g$ (number of idle agents)

WR-idt: $c_{k,g} = q_{k,g} + a_{k,g} w_k + b_{k,g} v_g$, combined with thresholds $m_{k,g}$ as in PT

WR-neg: WR in which $a_{k,g}$ and $b_{k,g}$ can be *negative*.

In WR-sep, the number of parameters is reduced by assuming that $a_{k,g}$ only depends on k and $b_{k,g}$ only depends on g. In WR-sep2, it is reduced further by assuming that $q_{k,g} = q_k + r_g$ where q_k only depends on k and r_g only depends on g. These are restricted forms of WR, so in principle they should never do better. WR-sep2 can be too restrictive (and perform poorly) when the service rates depend on both the call type and the agent: When an agent becomes idle (g is fixed), the weight $c_{k,g} = q_k + a_k w_k + r_g$ and the call selection depend only on $q_k + a_k w_k$ and not on g. The same applies to agent selection. However, they sometimes do better in practice when the parameters are optimized only approximately via heuristics, because there are fewer parameters to optimize, namely $K + G + \sum_{g=1}^{G} h_g$ and $2(K + G)$, respectively, compared to $3\sum_{g=1}^{G} h_g$ for WR. Note that one could take the solution of WR-sep or WR-sep2 as a starting point in the optimization of WR (we did not do it in our experiments).

In WR-idnum and WR-idt, i_g represents the current number of idle agents in group g, and $m_{k,g} \geq 0$ is the same real-valued threshold defined in the PT policy. In WR-idnum, we replace v_g by i_g, which is easier to update. In WR-idt, we keep v_g and add a threshold $m_{k,g}$ on i_g. If the threshold is not satisfied, then $c_{k,g} = -\infty$ automatically. These variants tend to perform well when there are call types with small volume. In our experiments, WR-idnum was often the best performer overall. We could also think of replacing v_g by the SL of call type k so far, or by the occupancy ratio of agents in group g so far during the day, etc. We did not try it.

Finally, in WR-neg we allow all coefficients to be negative. It is instructive to see what happens when we do that. For instance, if the objective is defined by the SL only, then there is no incentive to serve calls that have already waited more than the AWT. To approximate that (crudely), WR-neg will often select negative coefficients $a_{k,g}$, so that the more the call has waited, the lower is its priority. This does not lead to an optimal policy for F_S, because the FIFO rule only allows to serve the oldest call in each queue. Moreover, one may argue that this is a form of cheating, so we examine this variant only for comparison purpose and to exhibit its behavior.

WR would generalize LG$c\mu$ if we had separate intercepts $q^{\mathrm{A}}_{k,g}$ and $q^{\mathrm{C}}_{k,g}$, where A represents the agent selection and C the call selection. When there is no delay (the intercepts are never negative), separate intercepts would pose no problem; we would take the weight $c^{\mathrm{A}}_{k,g}$ with intercept $q^{\mathrm{A}}_{k,g}$ when a call arrives, and $c^{\mathrm{C}}_{k,g}$ with intercept $q^{\mathrm{C}}_{k,g}$ when an agent becomes available. This is a simple generalization of LG$c\mu$ with parameters that depend on both k and g. But when there are delays, both w_k and v_g can be positive, and it is unclear which intercept to choose. For this reason, we did not implement WR with two different intercepts. The delay rule is consistent in PD because

there is one delay time for each pair (k, g), and in WR because the same weight formula is used for each pair (k, g).

To better understand the relationship between the WR policy and the more traditional priority lists, and convince ourselves that WR is more general and flexible than the G, P and PD policies, we examine in the Online Appendix how these policies can be approximated arbitrary closely by WR policies. Because of this flexibility, we can expect WR to perform at least as well as G, P and PD, and our experiments confirm this. WR based on (10) cannot approximate the policies with idleness thresholds, PT and PDT, but WR-idt can do it.

The affine form of the weights in (10) is only one possibility. We could consider more general polynomials or other types of functions. But for more complex definitions of the weights, the implementation may become more complex and the optimization process can be much more difficult.

4. Routing Optimization

We now discuss how we have optimized the parameters of the routing policies in our experiments. This optimization is required not only for WR, but for other policies as well. This optimization process is difficult for the following reasons: the complexity of the routing mechanisms, the stochastic nature of the model (only noisy observations of the PMs can be obtained for any choice of parameters, via simulation), the mixture of combinatorial and real-value parameters, the dimension of the parameter space, and the possibly large number of local minima of the objective function. No efficient and foolproof optimization algorithm is available to optimize the routing policies with the *black box*-type objective functions considered here. We have to rely on heuristics.

We tested different algorithms based on known metaheuristic ideas. In this section, we outline a simple and easy-to-implement heuristic that we have adapted to optimize all the routing policies in this paper. It is based on the cross-entropy method for optimization and can be seen as a form of *modified genetic algorithm* (MGA).

4.1. A Modified Genetic Algorithm (MGA)

Classic genetic algorithms (Goldberg 1989) typically use crossover and mutation operators to generate randomly new populations of solutions. Their performance in applications depends very much on how these operators are defined, and this is highly problem-dependent. A class of so-called *model-based methods* that includes the *estimation of distribution algorithms* (EDA) (Mühlenbein and Paaß 1996, Larrañaga et al. 1999) and of the *cross-entropy* (CE) method for optimization (Rubinstein and Kroese 2004, de Boer et al. 2005, Botev et al. 2013), which became popular recently, can be viewed as a form of genetic algorithm that operates on a parameterized probability distribution $\Phi(\boldsymbol{\theta})$ over the set of solutions, by changing the parameter vector $\boldsymbol{\theta}$ rather than by making direct changes to the individual solutions, at each iteration.

Let $f(\mathbf{x})$ be the penalty function to minimize and suppose it has the form (7). Our MGA starts with an initial parameter vector $\boldsymbol{\theta} = \boldsymbol{\theta}^{(0)}$ for the distribution, generates a *population* of P admissible solutions from this distribution, estimates the costs of these P solutions by simulation, and keeps the $\hat{P}$ solutions having the smallest estimated costs, where $\hat{P} \le P$. This is the *elite population*. Then a new parameter vector $\tilde{\boldsymbol{\theta}}$ is estimated by maximum likelihood from the elite population, and $\boldsymbol{\theta}$ is reset to a convex combination of its previous value and the new $\tilde{\boldsymbol{\theta}}$, with the smoothing parameter γ. More precisely, we update $\boldsymbol{\theta} = \gamma\tilde{\boldsymbol{\theta}} + (1-\gamma)\boldsymbol{\theta}$, and we use $\gamma = 0.5$ in our experiments. Then, a new population of P solutions is generated from the corresponding distribution $\Phi(\boldsymbol{\theta})$, and so on. The idea is that by re-estimating the parameter vector from the elite sample at each iteration, the density (or mass) of the distribution should concentrate progressively around an optimal solution. We stop if (1) the trace (or the maximum element) of the covariance matrix of the current probability distribution is smaller than ϵ, or (2) we have reached a maximum number of iterations maxIt, or (3) the $(\hat{P}/P)$ quantile cost, which is the highest cost in the elite population,

did not reduce for maxItQ consecutive iterations. The algorithm returns the best solution found, $\mathbf{x}^*$, and an estimation $\hat{f}(\mathbf{x}^*)$ of $f(\mathbf{x}^*)$ obtained by independent (out-of-sample) simulations.

4.2. Routing policy optimization with the MGA

For the real-valued parameters in the policies, we use for Φ the multivariate normal distribution with independent coordinates. For the parameters that should not be negative, we truncate the normal distribution to $[0, \infty)$. The vector $\boldsymbol{\theta}$ has two coordinates for each parameter: the mean and the variance. The maximum likelihood estimator is easy to compute. For each parameter of the initial vector $\boldsymbol{\theta}^{(0)}$, we generally take the mean at (or near) 0 and a large variance (e.g., for WR and LG$c\mu$), and a smaller variance when more appropriate (e.g., for the delays and idle agent thresholds, we used initial means and standard deviations of 10 seconds and 1 agent). A good choice of initial variance can be selected via pilot runs. Its scaling must depend on the choice of time units; e.g., if we use hours rather than seconds for $q_{k,g}$, the initial standard deviation should be roughly 3600 times smaller. Further details on how we have applied MGA for all policy parameters, including ordering the priority lists, are given in the Online Supplement.

4.3. Re-optimization when the model is slightly modified

The MGA may require a large amount of time to find a good solution when there are many parameters to optimize. If this optimization has to be done from scratch for each half-hour of the day, and redone whenever the arrival-rate forecasts are updated or the staffing is changed, for example, the required computing time may become excessive. Fortunately, one can do much better simply by starting MGA from the previous solution and doing a more local search.

When a day is divided into 30-minute or 15-minute periods, the arrival rates and other model parameters typically change only gradually from one period to the next. For any good solution for the previous period, there is often a reasonably good solution for the current period that is not far from it. Thus, a natural approach when optimizing the routing parameters for a given period is to start from the solution for the previous period, and optimize around it. Similarly, if we re-optimize for a given period after small changes in the model (e.g., due to updates of call volume forecast, changes in staffing due to absenteeism or modified agent schedules, etc.), we can start from the previous solution for that period. This can be repeated more than once over the day.

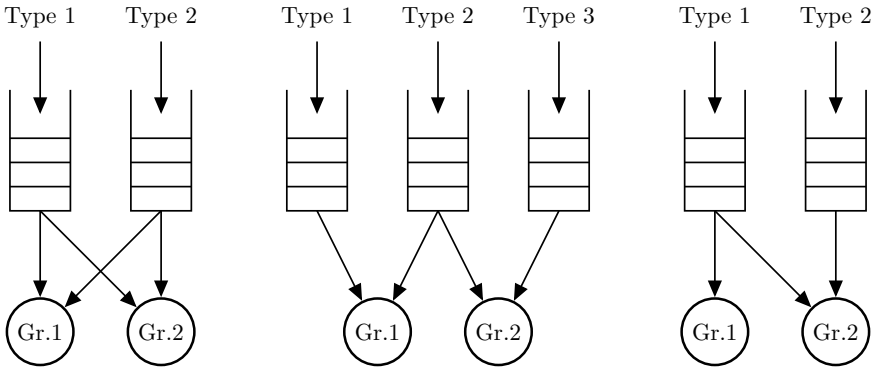
In our implementation, we set the initial mean values of the parameter distributions in the MGA to the values in the best solution obtained for the previous model, and we set the standard deviations to around 10% of the means (this percentage can be decreased if the model change is smaller, and increased if the change is more important). We had good success with this re-optimization strategy. The resulting solutions had costs similar to those obtained via a full optimization from scratch, but at only a small fraction of the cost. See Section 5.6 for a numerical illustration. This recycling of previous solutions is an important ingredient to make parameter optimization via the MGA practically viable.

5. Numerical Experiments

We report on numerical experiments with the routing policies, first for simple canonical models with 2 or 3 call types and 2 agent groups, then for a larger model with 6 call types and 8 agent groups. The simple models are the well-known X, W and N models (Gans et al. 2003), illustrated in Figure 1. Real-life call centers often have call types with uneven (both small and large) arrival volumes. This is embodied in our examples. Non-work-conserving policies are often useful in this type of situation (Bell and Williams 2001, Perry and Whitt 2009), e.g., to reserve agents with special skills for certain low-volume call types.

We have experimented with various arrival process models and distributions across the examples. For each call type k, the arrival process is either Poisson with fixed rate λ_k, or Poisson-gamma,

Figure 1 **The X, W and N models. Each model has 2 or 3 call types and 2 agent groups. The arrows between the call types and the agent groups define the skill sets.**



meaning that the arrival process of each day is Poisson with a rate that is gamma distributed. The service time distribution for call type k and agent group g is either exponential with mean $\mu_{k,g}^{-1}$ or lognormal with mean $\phi_{k,g}$ and standard deviation $\omega_{k,g}$. The patience times are exponential with mean ν_k^{-1} for call type k. All these random variables are independent.

All the rates and means in the models are given in minutes, unless specified otherwise. To avoid very small numbers, S_k, A_k, and O_g are given as percentages. The PMs of a solution are reported as $\mathbf{S} = (S_1, \ldots, S_K), \mathbf{A} = (A_1, \ldots, A_K)$, and $\mathbf{O} = (O_1, \ldots, O_G)$. In the simulator, all times (waiting times, agent idle times, delay times, etc.) are counted in seconds, for all policies.

There is no well-established systematic way to compare different performance vectors over multiple skills. Usually, one prefers several small gaps to the target instead of one skill performing very badly and the others on target. This motivates our choice of the following quadratic objectives:

$$F_{\mathrm{S}}(\pi) = \sum_{k=1}^{K} \max(t_k - S_k(\pi, \tau_k), 0)^2, \tag{11}$$

$$F_{\mathrm{SA}}^{\lambda}(\pi) = \sum_{k=1}^{K} \lambda_k \left(\max(t_k - S_k(\pi, \tau_k), 0)^2 + A_k(\pi)^2\right), \tag{12}$$

$$F_{\mathrm{SO}}(\pi) = \sum_{k=1}^{K} \max(t_k - S_k(\pi, \tau_k), 0)^2 + 5 \sum_{g=1}^{G} \left|O_g(\pi) - \bar{O}\right|^2. \tag{13}$$

In $F_{\mathrm{SA}}^{\lambda}(\pi)$, the call types are penalized in proportion to their volume, whereas in the other objectives, they all have the same weight. Likewise, we could penalize the occupancy imbalances by weights $c_{\mathrm{O},g} = y_g$. Setting those weights would be a managerial decision. One potential drawback of $F_{\mathrm{SA}}^{\lambda}(\pi)$ is that call types with small volumes may have virtually no influence on the total cost, and be neglected. A compromise could be to make the penalties partially proportional to the call volumes or number of agents. We also performed experiments where the objective functions penalized only the abandonments (F_{A}) or the expected waiting times. In those experiments, we found much less difference between the WR and LG$c\mu$ policies; that is, delays rarely helped. Section 5.4 gives one such example, where the penalty is on expected waiting times only and where WR and LG$c\mu$ perform equally well. Of course, this may not always be true.

Let f represent the objective function, selected from (11) – (13), to minimize. Since we cannot evaluate f exactly, we replace it by an estimate (a sample function) $\hat{f}$, which we optimize via the MGA algorithm described earlier. This $\hat{f}$ is defined as the sample objective function obtained by simulating n independent replications (or "days") of T hours of operation, with the same random numbers across all policies and all parameter values, so $\hat{f}$ becomes a function of the policy

parameters only (after the random numbers are fixed, it becomes a deterministic function). In other words, we generate the same history of calls, with the same arrival times, service times, and patience times, across all policies and parameter values. The total number of simulated hours per simulation, nT, is kept small to control the CPU time, so $\hat{f}$ can be quite noisy. We repeat the entire optimization process r times, independently; that is, we optimize separately r independent sample functions $\hat{f}$. We take $r = 5$ for all examples, except for the N-model where we set $r = 3$. We simulate the r final solutions out-of-sample, with $100n$ (100 times more) new independent replications and *common random numbers* across policies, and compare the policies. These out-of-sample evaluations provide 95% confidence interval of widths smaller than 1% for most PMs. Thus, we have an accurate evaluation of policies once their parameters are selected. For PDT, it is very difficult to optimize all the parameters simultaneously, so we execute a multi-stage (itMGA stages, to be exact) of the MGA. At each stage, the MGA optimizes a subset of parameters (more details in the Online Supplement).

The examples have been constructed so that it would be difficult to obtain a perfect score of 0 for the objective function. For the results reported here, the optimization is always performed from scratch, except in Section 5.6. The CPU times for optimization vary from 2 hours for the X-model, 4 hours for the W-model and around 15 hours for the larger example. We also repeated the experiments with a smaller number of MGA iterations, e.g., half the numbers reported here, and the comparison between the policies remains valid. We now give an idea of the optimization speed of MGA for WR. For the canonical examples, the first iteration of MGA usually finds costs that are about twice as large as the final cost. At half the number of iterations, WR is already better or equal to other policies and the out-of-sample cost is often less than 1% above the reported values. For the larger example, MGA can reduce the penalties of WR by an additionnal 10% to 20% if we double the CPU time. On the other hand, the non-WR policies do not show any significant improvements with longer optimization time. This gap often depends on the number of parameters to optimize and the size of the solution space. The MGA framework could be easily adapted to parallel computing. Also our optimization method could be improved further, for example by adding a local search around the retained solution. All executions were performed on a 2.4Ghz Intel Xeon CPU and simulations were performed using the Java library ContactCenters (Buist and L'Ecuyer 2005).

We have tested many examples and objective functions (including penalties on the average waiting times), but our reported results focus more on situations where the best policies are non-work-conserving, because these are the situations where WR is often much better than other policies. There are also many situations where WR is competitive, but not really better than other policies such as P, PD, PT, PDT, and LG$c\mu$. This is typically the case in heavy-traffic situations and when penalties are only on abandonments or waiting times.

5.1. Experiments with the X-model

We consider an X-model, with two call types and two agent groups, where each group can serve both call types, so $\mathcal{S}_1 = \mathcal{S}_2 = \{1, 2\}$, as shown in Figure 1. Typically, each group would have a primary skill and a secondary skill, and would be faster with its primary skill. We take Poisson arrivals with rates $\lambda_1 = 18$ and $\lambda_2 = 1.8$, exponential service times with rates $\mu_{1,1} = 0.198$, $\mu_{1,2} = 0.18$, $\mu_{2,1} = 0.162$, and $\mu_{2,2} = 0.18$, and exponential patience times with rates $\nu_1 = 0.12$ and $\nu_2 = 0.24$. The staffing vector has $y_1 = 90$ and $y_2 = 14$ (so the center has 104 agents), and the average agent occupancy is around 95%. The service level targets are $t_k = 80\%$ with $\tau_k = 20$ seconds for both call types. The simulation parameters are $T = 100$ hours and $n = 6$.

Table 1 gives the average out-of-sample estimated cost, followed by the lowest (best) cost (columns F_S^*, $F_{SA}^{\lambda*}$ and F_{SO}^*), over the $r = 5$ optimization runs, for each routing policy and objective function. It also shows the performances S_k and A_k, simulated out-of-sample, of the best run of

Table 1 **Average cost over 5 runs, followed by the cost and PMs of the best runs estimated out-of-sample for each policy, for the X-model example.**

Policy	F_{S}	F_{S}^*	(S_1, S_2, A_1, A_2)	F_{SA}^λ	$F_{\mathrm{SA}}^{\lambda *}$	(S_1, S_2, A_1, A_2)	F_{SO}	F_{SO}^*
G	143.2	143.2	(71.2, 71.9, 2.8, 5.5)	28.73	28.73	(71.2, 71.9, 2.8, 5.5)	143.3	143.3
P	21.3	20.9	(77.4, 76.2, 2.3, 5.6)	5.04	5.00	(77.4, 76.2, 2.3, 5.6)	36.9	36.9
PD	8.3	7.8	(77.3, 79.2, 2.5, 5.0)	4.42	**4.24**	(77.8, 76.9, 2.3, 5.7)	40.8	38.0
PT	13.2	12.6	(76.5, 79.3, 2.4, 5.0)	5.09	4.73	(77.7, 75.3, 2.2, 5.8)	20.0	19.4
PDT	**7.6**	**5.6**	(77.6, 78.4, 2.4, 5.4)	**4.35**	4.28	(77.8, 76.6, 2.3, 5.7)	**18.7**	15.8
LG$c\mu$	44.5	41.9	(75.2, 75.7, 2.2, 10.5)	8.23	6.68	(78.7, 74.0, 2.0, 11.3)	63.7	42.8
WR	8.8	8.2	(77.3, 79.1, 2.4, 4.9)	4.46	4.38	(78.0, 75.8, 2.2, 6.1)	19.1	18.6
WR-sep	8.2	8.0	(77.2, 79.4, 2.4, 4.9)	4.51	4.43	(77.8, 77.0, 2.2, 6.3)	19.4	18.3
WR-sep2	44.0	34.9	(75.7, 76.0, 2.5, 10.3)	8.59	8.40	(78.3, 72.6, 2.3, 12.0)	58.6	52.9
WR-idnum	8.8	8.3	(77.3, 79.0, 2.3, 4.9)	4.68	4.43	(77.5, 77.9, 2.3, 5.2)	21.4	**12.4**
WR-idt	13.5	12.7	(76.5, 79.4, 2.4, 5.2)	4.71	4.51	(78.2, 75.7, 2.2, 7.0)	18.9	18.6
WR-neg	10.3	7.9	(77.3, 79.1, 2.4, 4.9)	**4.23**	**3.82**	(78.6, 78.0, 2.2, 7.5)	**17.7**	16.5

Table 2 **Best solution out of 5 runs for the policy LG$c\mu$, for the X-model example.**

Objective	a_1	b_1	a_2	b_2	e_1	f_1	e_2	f_2
F_{S}	538	101	2418	0	1116	152	991	72.1
F_{SA}^λ	147	201	2824	0.626	1590	8.19	1512	1.79
F_{SO}	1220	142	3844	0.190	855	132	1244	107

Table 3 **Best solution out of 5 runs for selected policies of WR and objective functions, for the X-model.**

Policy	Objective	$q_{1,1}$	$q_{1,2}$	$q_{2,1}$	$q_{2,2}$	$a_{1,1}$	$a_{1,2}$	$a_{2,1}$	$a_{2,2}$	$b_{1,1}$	$b_{1,2}$	$b_{2,1}$	$b_{2,2}$
WR	F_{S}	1129	−116	384	1213	19.3	6.48	0.952	0.831	1.25	0.787	3.68	0
WR	F_{SA}^λ	2405	−549	−350	943	172	42.0	20.3	302	31.2	7.30	0	81.3
WR	F_{SO}	3250	−975	188	1600	243	1.98	18.5	72.1	27.9	111	7.61	78.5
WR-sep	F_{S}	2195	−232	355	1456	12.6	12.6	5.56	5.56	6.20	1.35	6.20	1.35
WR-sep	F_{SA}^λ	1674	−1075	−84.1	2937	76.5	76.5	5.48	5.48	48.5	6.32	48.5	6.32
WR-idnum	F_{S}	1503	−323	345	443	3.64	13.8	10.4	6.01	882	95.7	87.0	951
WR-neg	F_{SA}^λ	2498	−842	313	3554	190	46.6	−1.40	−24.8	89.5	7.93	44.2	230
WR-neg	F_{SO}	2191	−697	−533	1079	144	−32.6	36.9	186	129	149	0.462	61.8

each policy for the objectives F_{S} and F_{SA}^λ. The best solutions for LG$c\mu$ and selected WR policies are shown in Tables 2 and 3. Typically, the cost of the returned policy has small variance, but the solutions themselves can vary a lot; i.e., very different policies can have similar near-optimal costs. As an illustration, the best solution of WR for F_{SA}^λ, shown in Table 3, row 2, has an out-of-sample cost of 4.38. The second best solution of WR has a cost of 4.39 and its parameters are quite different: $(q_{1,1}, q_{1,2}, q_{2,1}, q_{2,2}) = (2145, -564, -160, 2242)$, $(a_{1,1}, a_{1,2}, a_{2,1}, a_{2,2}) = (123, 46.3, 11.5, 127)$ and $(b_{1,1}, b_{1,2}, b_{2,1}, b_{2,2}) = (104, 10.5, 28.4, 69.0)$. Many reasons can explain this richness of solutions, for example: if a group has to give strict priority to a call type, then the corresponding weights can take any large positive values, or in the case a group should never answer a call type, then the weights can take any large negative values.

Overall, the work-conserving policies do not perform well for this example. Policy G is the worst performer, followed by LG$c\mu$ and P, along with WR-sep2, which can be non-work-conserving. The cost differences between policies are smaller when the penalties are proportional to the arrival rates with F_{SA}^λ. We can observe the effect of different penalty weights: lower-volume call type 2 has more abandonments and lower SL under F_{SA}^λ than F_{S}. Among the WR policies, the simplified WR-sep2 does poorly and the others do well in all cases except for WR-idt with the F_{S} objective, where the performance is not as good. The PT policy is comparable to WR-idt. PD and PT perform well for some objectives, while the policies PDT, WR and WR-sep perform well for all objectives. WR-neg

performs the best for F_{SO} and $F_{\mathrm{SA}}^{\lambda}$, however the solutions often have negative $a_{k,g}$ coefficients. In particular, $a_{2,1} < 0$ for $F_{\mathrm{SA}}^{\lambda}$ and $a_{1,2} < 0$ for F_{SO}. We present a more detailed discussion of the routing solutions in the Online Supplement.

5.2. Experiments with the W-model and Poisson-gamma arrivals

We take a more elaborate example with Poisson-gamma arrivals and lognormal service times. The W-model has 3 call types and 2 agent groups with skill sets $\mathcal{S}_1 = \{1, 2\}$ and $\mathcal{S}_2 = \{2, 3\}$, as shown in Figure 1. The call center is open 10 hours a day and starts with empty queues. The arrival process for each call type is Poisson-gamma, with a stationary Poisson arrival process for each day, whose random rate (for the entire day) has a gamma distribution. These gamma random variables for the three call types have means $(3000, 1000, 200)$ and standard deviations $(244.9, 223.6, 40)$. The arrival processes are independent across call types.

The service times are lognormals with means $(\phi_{1,1}, \phi_{2,1}, \phi_{2,2}, \phi_{3,2}) = (8, 10, 9, 15)$ and standard deviations $(\omega_{1,1}, \omega_{2,1}, \omega_{2,2}, \omega_{3,2}) = (8, 10, 11, 12)$. The patience times are exponential with means $(\nu_1^{-1}, \nu_2^{-1}, \nu_3^{-1}) = (5, 9, 10)$. The staffing numbers are $y_1 = 48$ and $y_2 = 12$, for a total of 60 agents. The average agent occupancy is 92%, but it can drop to 85% when subject to occupancy fairness under F_{SO}. The SL targets are $t_1 = t_2 = 80\%$ and $t_3 = 90\%$ of calls answered within $\tau_1 = 60$, $\tau_2 = 90$ and $\tau_3 = 30$ seconds. The simulation parameters are $T = 10$ hours and $n = 300$ simulated days.

In this example, call type 3 has the smallest volume, but group 2 is faster in serving the shared call type 2. Also, type 3 requires a higher quality of service, and type 1 is more impatient. The FSF rule here is not optimal. It is better to "reserve" enough agents of group 2 to answer calls of type 3. The arrival rates are quite variable, so there are days when the call center will be overloaded and days when it will be underloaded.

Table 4 **Average cost over 5 runs, followed by the cost and PMs of the best run estimated out-of-sample for each of the retained policies, for the W-model example.**

Policy	$F_{\mathrm{SA}}^{\lambda}$	$F_{\mathrm{SA}}^{\lambda *}$	$(S_1, S_2, S_3, A_1, A_2, A_3)$	F_{SO}	F_{SO}^{*}	$(S_1, S_2, S_3, O_1, O_2)$
G	12.39	12.39	(82, 94, 58, 8.4, 3.9, 7.5)	1068	1068	(82, 93, 58, 93, 90)
P	8.97	8.97	(90, 77, 69, 6.6, 9.9, 5.1)	519	519	(78, 100, 68, 93, 89)
PD	8.95	8.91	(90, 78, 71, 6.8, 10.1, 4.8)	228	201	(76, 100, 76, 86, 85)
PT	**7.44**	7.35	(85, 79, 83, 7.8, 8.2, 3.1)	66	62	(77, 78, 85, 91, 88)
PDT	7.50	7.34	(85, 79, 83, 7.7, 8.5, 3.1)	**4**	**1**	(84, 81, 89, 85, 85)
LG$c\mu$	8.59	8.53	(89, 81, 69, 7.1, 8.2, 5.1)	443	423	(78, 100, 71, 93, 88)
WR	7.78	7.71	(86, 79, 79, 7.8, 8.3, 3.6)	60	15	(79, 86, 86, 84, 84)
WR-sep	7.76	7.68	(89, 79, 78, 7.3, 9.2, 3.8)	119	91	(73, 100, 84, 86, 84)
WR-idnum	7.47	**7.31**	(86, 80, 79, 7.6, 8.0, 3.6)	20	**1**	(79, 91, 89, 85, 86)
WR-idt	7.49	7.36	(85, 79, 83, 7.9, 8.0, 3.1)	17	4	(80, 82, 88, 86, 86)

Table 5 **Best solution out of 5 runs for selected policies of WR, for the W-model.**

Policy	Objective	$q_{1,1}$	$q_{2,1}$	$q_{2,2}$	$q_{3,2}$	$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,2}$	$b_{1,1}$	$b_{2,1}$	$b_{2,2}$	$b_{3,2}$
WR	$F_{\mathrm{SA}}^{\lambda}$	313	316	−1337	267	191.6	47.5	2.3	78.9	75.3	105.9	25.9	47.3
WR	F_{SO}	−929	−1353	−5497	−148	16.0	18.3	0	75.9	0.306	0.817	41.4	81.1
WR-idnum	$F_{\mathrm{SA}}^{\lambda}$	174	−85	−1099	395	36.8	13.0	3.6	7.2	760.3	375.8	777.5	463.9
WR-idnum	F_{SO}	−875	−648	−1866	365	14.4	10.6	2.85	11.7	12.8	28.4	862	623

Table 4 gives the average costs (columns $F_{\mathrm{SA}}^{\lambda}$ and F_{SO}) and lowest costs (columns $F_{\mathrm{SA}}^{\lambda *}$ and F_{SO}^{*}) over the $r = 5$ optimization runs, followed by the PMs of the best runs, estimated out-of-sample for each policy and selected objective functions. The best policies overall are those with thresholds: WR-idnum, WR-idt, PT and PDT. WR and WR-sep follow closely behind, then PD. The solution

costs of the WR policies have greater variance under the function F_{SO}. The non-work-conserving policies, LG$c\mu$, P, and G, are the worst performers. This behavior agrees with the observations of Perry and Whitt (2009), who pointed out the usefulness of threshold policies in unexpected overloaded conditions. Table 5 shows the best solutions for 4 WR policies.

Policies P and LG$c\mu$ fail to provide good SL to call type 3 because group 2 serves too many calls of type 2. In our best solutions with these policies for F_{SO}, group 1 serves a maximum number of calls of type 2 and group 2 always prioritizes call type 3 over type 2. With $F^{\lambda}_{\mathrm{SA}}$, the weight of type 3 in the objective function (its arrival rate) is 15 times smaller than that of type 1 and this has an impact on the solution. Even if S_1 is well above its target of 80%, the penalty incurred by the abandonments A_1 outweights the really bad service level S_3.

For each policy and objective function, the costs of the 5 runs estimated out-of-sample do not vary much in general. As a typical example, the costs with WR-idnum for $F^{\lambda}_{\mathrm{SA}}$ are 7.31, 7.36, 7.48, 7.57 and 7.64.

5.3. An example with 6 call types and 8 agent groups based on a real call center

We now consider an example based on the call center of a utility provider in Quebec, which employs nearly 1500 agents. It has 96 call types and over 375 agent groups. The reason for the high number of call types is that customers can choose to be served in either French (primary language) or English, and they are categorized as residential or commercial clients. However, residential clients asking service in the primary language constitute the bulk of the volume. This also entails fewer observations of the multiple rare call types, and their estimations can be very noisy. For this reason, we choose to restrict our example to the 6 highest volume call types (representing 64% of the total volume) and the 8 largest groups that can serve them. The skill sets of these groups are $\mathcal{S}_1 = \{1,3,4,5\}$, $\mathcal{S}_2 = \{1,2\}$, $\mathcal{S}_3 = \{3,5\}$, $\mathcal{S}_4 = \mathcal{S}_7 = \{3,5,6\}$, $\mathcal{S}_5 = \{1,3,5\}$, $\mathcal{S}_6 = \{1,2,3,5\}$ and $\mathcal{S}_8 = \{1,3,5,6\}$. Groups 4 and 7 have identical skill sets but they serve at different speeds.

We selected a specific period of the day. Then we took a typical staffing vector for the 8 groups from the data and we rescaled it by an appropriate factor to obtain the staffing $(y_1,\ldots,y_8) = (26,22,38,9,25,9,16,12)$, for a total of 157 agents. We assume that the arrivals are Poisson with constant rates, independent across call types, patience times are exponential, and service times are lognormal and depend on both the call type and the agent group. The parameters of these distributions were estimated from approximately one year of available data for the selected period. This gave the arrival rates $(\lambda_1,\ldots,\lambda_6) = (3.72, 0.453, 9.15, 0.607, 2.92, 0.670)$ and mean patience times $(\nu_1^{-1},\ldots,\nu_6^{-1}) = (52.0, 36.0, 40.8, 50.8, 41.3, 15.3)$. These long patience times may be explained by the monopolistic advantage of this provider. Each of the 25 pairs (k,g) of agent group g and skill $k \in \mathcal{S}_g$ has its own lognormal service time parameters. This call center is open from Monday to Friday, and most of the skill pairs have been observed on 240 to 252 days. These numbers vary because the call center keeps evolving: the routing policy changes, agents change groups, groups sometimes have no agents, etc. The estimated means vary from 5.14 to 11.3 minutes and the standard deviations are from 5.88 to 22.0. More details, including the distribution parameters for each skill pair, are given in the Online Supplement. We optimize the parameters of the routing policies over a time horizon of $T = 100$ hours, which approximates a steady-state behavior, with $n = 6$ replications. The average agent occupancy is around 85% in this example. The SL targets are $t_k = 80\%$ with $\tau_k = 120$ seconds for all call types.

A particularity of this example is the random after-service time of an agent (to complete the paperwork, close the client's files, go to the restroom, ...). When an agent hangs up a call with a customer, there is 60% chance that he will disconnect and be *unavailable* for a random exponential time with mean of 2 minutes. This disconnection time is counted in the idle time of the agent, but only *available* idle agents are counted in the threshold rules of PT, PDT, WR-idnum and WR-idt. As a consequence, the occupancy ratio is never expected to be near 100% even when the call center is strongly understaffed.

We execute the MGA with $P = 400$, $\hat{P} = 20$ and maxIt $= 30$. For PDT, we take maxIt $= 10$, $P = 200, \hat{P} = 10$, and we set the multi-step parameter itGMA $= 7$. The parameters of PD or PT are hard to optimize simultaneously, so we run the MGA in two stages. The first stage optimizes the priorities with maxIt $= 20$ and the second stage optimizes the delays or thresholds with maxIt $= 10$. This is suboptimal, but it finds better solutions than the one-stage version with the given CPU budget.

Table 6 **Average and lowest costs estimated out-of-sample for the retained policy, over 5 runs, for the example with 6 call types and 8 groups.**

Objective	G	P	PD*	PT*	PDT	LG$c\mu$	WR	WR-sep	WR-idnum	WR-idt	WR-neg
F_{S}	2679	215	185	127	126	317	**37.4**	48.8	58.4	130	37.5
F_{S}^{*}	2679	192	121	84.9	78.7	292	**25.9**	41.3	42.3	83.5	28.5
$F_{\mathrm{SA}}^{\lambda}$	105	17.6	16.1	16.6	15.0	21.2	**5.64**	6.83	5.77	12.8	7.16
$F_{\mathrm{SA}}^{\lambda *}$	105	13.3	11.2	11.9	12.8	17.4	**4.62**	5.53	4.63	11.0	5.61
F_{SO}	2793	326	282	278	219	376	**101**	137	109	189	**92.6**
F_{SO}^{*}	2793	268	215	221	194	359	82.0	109	**72.8**	172	80.4

Table 6 reports the average costs (rows $F_{\mathrm{S}}, F_{\mathrm{SO}}$ and $F_{\mathrm{SA}}^{\lambda}$) and lowest costs (rows $F_{\mathrm{S}}^{*}, F_{\mathrm{SO}}^{*}$ and $F_{\mathrm{SA}}^{\lambda *}$) of $r = 5$ optimization runs, evaluated out-of-sample. The lowest costs for each objective function are in bold. PD* and PT* denote PD and PT policies whose parameters are optimized via two-stage MGA (the solutions from the single-stage MGA are worse than P in this case).

The WR policy gives the best results overall, followed by other WR variants. Theoretically, WR-neg should be at least as good as WR, because it is more general, but it is also more difficult to optimize, so it sometimes gives a larger cost. The simpler WR-sep may have an advantage over WR during early iterations of MGA, but WR eventually prevails. WR-idt performs poorly compared with its WR cousins; optimizing the weights and thresholds simultaneously is more difficult. Although the out-of-sample costs vary over the 5 runs, the WR family clearly dominates.

Let us look at the changes in PMs when changing the penalty weights. The PMs of the best runs with WR are $\mathbf{S} = (79.0, 96.3, 75.4, 78.6, 80.1, 78.8)$ and $\mathbf{A} = (2.3, 1.6, 3.0, 2.5, 2.7, 7.4)$ for F_{S}, and $\mathbf{S} = (77.8, 73.8, 77.8, 73.9, 77.3, 86.9)$ and $\mathbf{A} = (2.3, 3.7, 2.9, 3.1, 2.9, 5.7)$ for $F_{\mathrm{SA}}^{\lambda}$. With F_{S}, it is more advantageous to improve the quality of service of the call types having smaller volumes, and the opposite holds for $F_{\mathrm{SA}}^{\lambda}$. Call type 3, which has the highest volume, has the lowest SL at 75.4% for F_{S}, but its SL is increased to 77.8% for $F_{\mathrm{SA}}^{\lambda}$. The opposite occurs for call type 2, which has the lowest volume: its SL goes from 96.3% for F_{S} down to 73.8% for $F_{\mathrm{SA}}^{\lambda}$.

5.4. N-model with penalties on expected waiting times only

In certain settings, under specific asymptotic regimes, the form of the optimal policy is known explicitly. For example, in an N-model under heavy-traffic for which the cost is defined as a linear combination of the mean waiting times of the two call types, the optimal policy is known to be either a $c\mu$-type rule or a priority rule (such as policy P), depending on the coefficients in the linear combination, see Tezcan and Dai (2010). In the Online Supplement, we examine how well WR and LG$c\mu$, combined with MGA, perform for this type of example. We find that they both perform very well: In all cases that we have tried, the MGA returns policies that are practically as good as the asymptotically optimal ones.

5.5. Robustness of the policies

We compare the robustness of the routing policies through simple tests with the W-model presented in Section 5.2. To execute these tests, we take the best solution out of 5 runs for $F_{\mathrm{SA}}^{\lambda}$ and we change deterministically the parameters of the call center. We simulated 2000 days for each scenario. In

the figures, we compare the policies by their cost ratio over the lowest cost (among the tested solutions); the best policy has a ratio of 1. The vertical axes of the graphs are in log scale.

We first test the robustness of the policies facing different arrival processes. The policies were optimized assuming independent Poisson-gamma arrival process for each call type, where each day has a stationary Poisson process but with random daily arrival rate. We are now interested in seeing how the routing policies perform on a particular day with fixed Poisson arrival rates. The rates are chosen within one standard deviation from the mean of the gamma variables, so these scenarios are quite likely to happen.

Figure 2 **W-model with objective function $F_{\mathrm{SA}}^{\lambda}$: we simulate the best solution of each policy, but change the arrival rates by the quantity (z_1, z_2, z_3) given in the figure.**

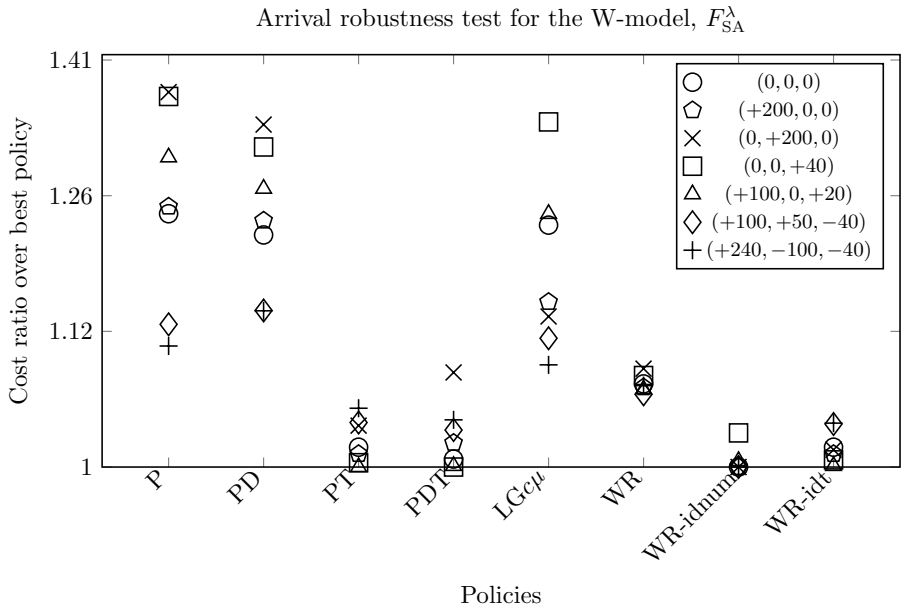
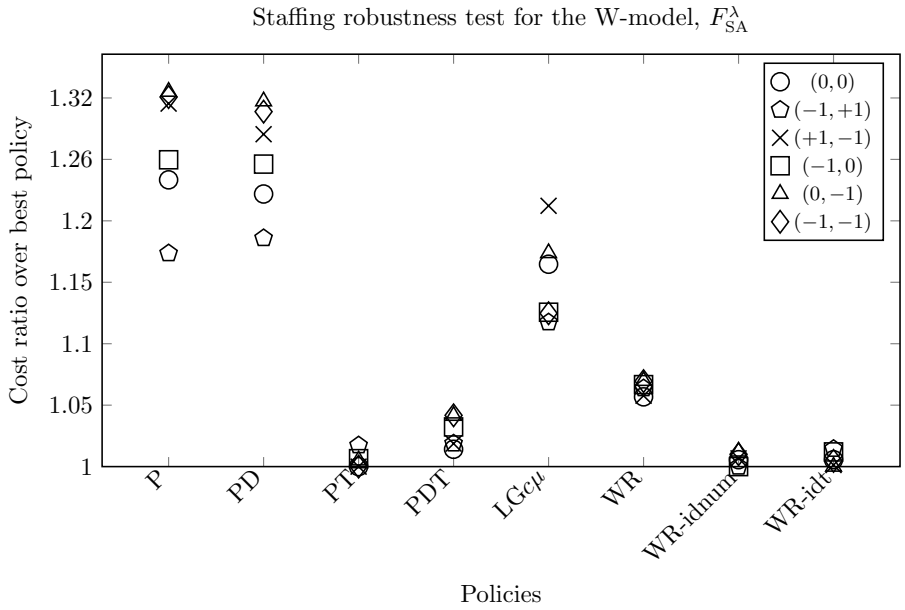


Figure 2 presents the results where the daily arrival rate of type k is changed by z_k from the mean rates of (3000, 1000, 200). WR-idnum is the most robust. Overall, WR-idt, PDT, and PT are competitive, followed by WR. Policies P, PD and LG$c\mu$ do not do well in these tests. Using hard threshold policies like PT and PDT may not work so well when the overloaded types are in opposition with the thresholds. In this example, group 2 gives priority to type 3, and policies PT and PDT set a positive threshold on $m_{2,2}$. The policies PT and PDT perform well when call type 3 becomes overloaded, but not when type 3 is underloaded and type 1 is overloaded. Group 1 needs to serve more call type 1, since they are the only agents that have the right skill, and subsequently, group 2 needs to serve more of call type 2. It is not a good idea to have a hard threshold in this case. We see WR-idnum as a better choice.

In the second test, we modify the staffing vector while keeping the original Poisson-gamma arrival process. We do not increase the total sum of agents because it is less interesting and absenteeism is a common problem in practice. The original staffing vector is $y_1 = 48$ and $y_2 = 12$, and we change the number of agents by quantities Δy_1 and Δy_2. Figure 3 shows the results for different scenarios. Reducing the staffing level increases the load of the call center, and policies using thresholds (WR-idnum, WR-idt, PT and PDT) work better. WR-idnum is the most robust, followed closely by WR-idt and PT.

Figure 3 **W-model with objective function $F_{\mathrm{SA}}^{\lambda}$: we simulate the best solution of each policy, but change the staffing vector by the quantity $(\Delta y_1, \Delta y_2)$ given in the figure.**



5.6. Experiments with MGA for re-optimization

Here we experiment with the re-optimization approach described in Section 4.3, with the same W-model example as in Sections 5.2 and 5.5, with the $F_{\mathrm{SA}}^{\lambda}$ objective function, for policies P, LG$c\mu$, WR, WR-idnum and WR-idt. We start with the same solutions that were computed in Section 5.2 and used in Section 5.5 for the Poisson-gamma arrival processes with means of (3000, 1000, 200) calls per day and the original staffing vector $(y_1, y_2) = (48, 12)$. Then we suppose that the arrival-rate vector takes a specific value that differs from the mean by (z_1, z_2, z_3), as in Section 5.5, and we re-optimize locally the parameters of the routing policies. For this re-optimization, we reduce the number of iterations of the MGA by a factor of 6 and the number of simulated days from $n = 300$ to $n = 100$, compared with the original optimization; so the computing budget is reduced roughly by a factor of 18. The re-optimization takes less than 12 minutes in all cases, whereas the full optimization from scratch was taking over 4 hours in some cases. Note that for policy P in this example, there are just a few combinations to compare, so the optimization is much faster than for LG$c\mu$ and WR, and the re-optimization just compares again all the possibilities. For each policy and each test vector (z_1, z_2, z_3), we repeated the re-optimization $r = 5$ times independently, and estimated the average cost of those 5 solutions using independent (out-of-sample) simulations over $n = 30,000$ days. This average cost is reported in column "Step-2" of Table 7 for each policy and each test case. Column "Step-1" gives the cost of the solution found by the original optimization procedure (these are the costs used in Figure 2). Table 8 shows the results of a similar experiment when the staffing vector is changed by quantities $(\Delta y_1, \Delta y_2)$, with the original Poisson-gamma arrival process. This corresponds to Figure 3.

We find from those results that the cost with policy P is virtually unchanged after the re-optimization; there is no improvement because this policy is not sufficiently flexible. For LG$c\mu$, the re-optimization brings small improvements over the original solutions given in column "Step-1." For the WR, WR-idnum, and WR-idt policies, we find both small and large improvements, depending on the case. WR and its variants remain less expensive than P and LG$c\mu$. WR-idnum is the least expensive policy in almost all cases.

To test the effectiveness of the re-optimization with a small computing budget, we also performed a full optimization of each policy for each test case (starting with the default initial parameters).

Table 7 **Comparing the average costs with re-optimized routing parameters, for the W-model, objective function $F_{\mathrm{SA}}^{\lambda}$ and Poisson constant processes with different arrival rates.**

Test case	P		LG$c\mu$		WR		WR-idnum		WR-idt	
(z_1, z_2, z_3)	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2
(0, 0, 0)	7.25	7.25	7.20	7.19	6.28	6.19	5.85	5.74	5.94	5.81
(+200, 0, 0)	14.47	14.47	13.26	13.26	12.39	12.15	11.57	11.51	11.74	11.66
(0, +200, 0)	19.33	19.41	16.04	15.35	15.35	14.08	14.17	13.02	14.32	13.74
(0, 0, +40)	11.11	11.21	10.89	10.83	8.80	8.63	8.38	7.99	8.20	7.97
(+100, 0, +20)	12.54	12.54	11.91	11.90	10.29	10.27	9.73	9.74	9.66	9.62
(+100, +50, -40)	8.48	8.48	8.39	8.35	7.99	7.78	7.48	7.39	7.77	7.55
(+240, -100, 40)	7.58	7.63	7.45	7.33	7.30	6.96	6.83	6.60	7.09	6.80

Table 8 **Comparing the costs with re-optimized routing parameters, for the W-model, objective function $F_{\mathrm{SA}}^{\lambda}$ and different staffing vectors.**

Test case	P		LG$c\mu$		WR		WR-idnum		WR-idt	
$(\Delta y_1, \Delta y_2)$	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2
(-1, +1)	8.54	8.54	8.17	8.15	7.77	7.65	7.31	7.31	7.44	7.40
(+1, -1)	10.34	10.38	9.63	9.58	8.37	8.44	7.94	7.93	7.88	7.94
(-1, 0)	12.19	12.19	10.96	10.94	10.40	10.27	9.74	9.76	9.85	9.89
(0, -1)	13.69	13.36	12.10	11.84	11.01	10.70	10.41	10.15	10.33	10.15
(-1, -1)	18.09	17.97	15.46	14.79	14.62	13.86	13.80	12.94	13.76	13.50

We found that the full optimization gave solutions whose costs were very close (sometimes a little lower or a little higher) to the costs obtained by the local re-optimization.

This experiment indicates that a two-step approach, which consists of performing a lengthy optimization from scratch sometime in advance, and then a short local re-optimization at the last minute when more accurate forecasts are available or some model parameters have changed slightly, can be effective and viable in a practical context. The lengthy optimizations can be performed on multiple parallel processors if needed, say one processor per period of the day, for example.

6. Conclusion

We proposed a routing policy based on weights, expressed as linear functions of the call waiting times and agent idle times, or number of idle agents. To optimize the policy parameters, we adapted the cross-entropy method in a setting where performance constraints are incorporated into the objective function via penalty costs. This method evaluates the objective function as a black box that returns noisy values. Important advantages of our simulation-based optimization approach over other analytical-based or numerical methods (for example, fluid networks and dynamic programming) are that it permits one to optimize practical call center problems without relying on asymptotic-type approximations, to easily change the objective function, and use basically any performance measure implementable in a simulator (such as the SL, abandonment rate, the AWT, agent occupancy, etc.). We compared the performance of our policy to routing policies commonly used in practice, on various examples. The flexibility of using weights instead of fixed priority rules or thresholds was often reflected in the solutions. There are problem instances where our WR policies gave far better results than other policies commonly used in practice. This typically occurs when there are both low-volume and high-volume call types, and when the objective function includes performance measures with thresholds, such as the SL. Moreover, in all our experiments, in situations where WR policies were not the best, they were competitive with the best. We do not claim that WR policies should be used in all cases, but they are certainly worth adding to the toolbox.

Possibilities for further research include improving the WR policy by considering weights that are nonlinear functions of the waiting and idle times, and that may depend on the state of the queues.

Another possibility, relevant to practice, is to extend the study on the robustness of routing policies. Our preliminary experiments suggest that WR-idnum is more robust than the other commonly-used routing policies. We plan to study this further, including the possibility of dynamically updating the policy parameters when the arrival rates depart significantly from the forecasts. An even more challenging program is to develop effective methods that can simultaneously optimize the routing parameters, the staffing, and eventually the work schedules of agents, in realistic settings.

Acknowledgments
This research has been supported by grants from NSERC-Canada and Bell Canada, as well as a Canada Research Chair and an Inria International Chair to the third author. We are grateful to Naoufel Thabet at Bell Canada for helpful discussions. We thank the referees and editors for their constructive comments and suggestions that helped to improve the quality of this paper.

References

Akşin, O. Z., M. Armony, V. Mehrotra. 2007. The modern call center: A multi-disciplinary perspective on operations management research. *Production and Operations Management* **16**(6) 665–688.

Armony, M., A. R. Ward. 2010. Fair dynamic routing in large-scale heterogeneous-server systems. *Operations Research* **58**(3) 624–637.

Armony, M., A. R. Ward. 2013. Blind fair routing in large-scale service systems with heterogeneous customers and servers. *Operations Research* **61**(1) 228–243.

Atar, R. 2005. Scheduling control for queueing systems with many servers: Asymptotic optimality in heavy traffic. *Annals of Applied Probability* **15**(4) 2606–2650.

Atar, R., C. Giat, N. Shimkin. 2010. The $c\mu/\theta$ rule for many-server queues with abandonment. *Operations Research* **58**(5) 1427–1439.

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

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

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

Bell, S. L., R. J. Williams. 2001. Dynamic scheduling of a system with two parallel servers in heavy traffic with resource pooling: asymptotic optimality of a threshold policy. *Annals of Applied Probability* **11** 608–649.

Bell, S. L., R. J. Williams. 2005. Dynamic scheduling of a parallel server system in heavy traffic with complete resource pooling: Asymptotic optimality of a threshold policy. *Electronic Journal of Probability* **10** 1044–1115.

Botev, Z. I., D. P. Kroese, R. Y. Rubinstein, P. L'Ecuyer. 2013. The cross-entropy method for optimization. *Handbook of Statistics, Volume 31: Machine Learning*. North Holland.

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

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

de Boer, P.-T., D. P. Kroese, S. Mannor, R. Y. Rubinstein. 2005. A tutorial on the cross-entropy method. *Annals of Operations Research* **134**(1) 19–67.

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

Gans, N., Y.-P. Zhou. 2003. A call-routing problem with service-level constraints. *Operations Research* **51**(2) 255–271.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st ed. Addison-Wesley Longman Publishing Co., Boston, MA, USA.

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

Gurvich, I., W. Whitt. 2009. Queue-and-idleness-ratio controls in many-server service systems. *Mathematics of Operations Research* **34**(2) 363–396.

Gurvich, I., W. Whitt. 2010. Service-level differentiation in many-server service systems via queue-ratio routing. *Operations Research* **58**(2) 316–328.

Harrison, J. M., A. Zeevi. 2004. Dynamic scheduling of a multi-class queue in the Halfin-Whitt heavy-traffic regime. *Operations Research* **52** 243–257.

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

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

Koole, G., B. F. Nielsen, T. B. Nielsen. 2009. Optimization of overflow policies in call centers. Submitted.

Koole, G., A. Pot. 2005. Approximate dynamic programming in multi-skill call centers. M. E. Kuhl, N. M. Steiger, F. B. Armstrong, J. A. Joines, eds., *Proceedings of the 2005 Winter Simulation Conference*. IEEE Press, 576–583.

Larrañaga, P., R. Etxeberria, J. A. Lozano, B. Sierra, I. Inza, J. M. Peña. 1999. A Review of the Cooperation between Evolutionary Computation and Probabilistic Graphical Models. A. Ochoa, M. R. Soto, R. Santana, eds., *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*. Habana, Cuba, 314–324.

Liao, S., C. Van Delft, G. Koole, O. Jouini. 2010. Shift-scheduling of call centers with uncertain arrival parameters. *MOSIM 2010*. Hammamet, Tunisia, 1–10. `http://www.enim.fr/mosim2010/program.php`.

Mandelbaum, A., A. L. Stolyar. 2004. Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalized c-mu rule. *Operations Research* **52** 836–855.

Milner, J. M., T. L. Olsen. 2008. Service-level agreements in call centers: Perils and prescriptions. *Management Science* **54**(2) 238–252.

Mühlenbein, H., G. Paaß. 1996. From recombination of genes to the estimation of distributions 1. Binary parameters. Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, Hans-Paul Schwefel, eds., *Parallel Problem Solving from Nature – PPSN IV*, *Lecture Notes in Computer Science*, vol. 1141. Springer Berlin / Heidelberg, 178–187.

Perry, O., W. Whitt. 2009. Responding to unexpected overloads in large-scale service systems. *Management Science* **55**(8) 1353–1367.

Rubinstein, R. Y., D. P. Kroese. 2004. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer-Verlag.

Sisselman, M. E., W. Whitt. 2007. Value-based routing and preference-based routing in customer contact centers. *Production and Operations Management* **16**(3) 277–291.

Tezcan, T., J. G. Dai. 2010. Dynamic control of N-systems with many servers: Asymptotic optimality of a static priority policy in heavy traffic. *Operations Research* **58**(1) 94–110.

van Mieghem, J. A. 1995. Dynamic scheduling with convex delay costs: the generalized cmu rule. *Annals of Applied Probability* **5** 809–833.

van Mieghem, J. A. 2003. Due-date scheduling: Asymptotic optimality of generalized longest queue and generalized largest delay rules. *Operations Research* **51**(1) 113–122.

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

Xu, S. H., R. Righter, J. G. Shanthikumar. 1992. Optimal dynamic assignment of customers to heterogeneous servers in parallel. *Operations Research* **40**(6) 1126–1138.