PREFERENCE-BASED ROUTING

by

Michael E. Sisselman

SeatLink, Inc.
917-837-7223
msisselman@seatlink.net

Ward Whitt

Columbia University
212-854-7255
ww2040@columbia.edu

February 24, 2005

Abstract

Telephone call centers and their generalizations - customer contact centers - usually handle several types of customer service requests (calls). Since customer service representatives (agents) have different call-handling abilities, contact centers exploit skill-based routing to assign calls to appropriate agents, aiming to respond properly as well as promptly. Nevertheless, despite a high level of sophistication, contact-center performance often falls far short of expectations. Too frequently, both customers and agents are dissatisfied with their contact-center experience. Agent dissatisfaction is demonstrated by high turnover and absenteeism. This problem might be addressed by empowering the agents, as others have observed, but that is difficult to accomplish in today's contact centers. As a new strategy to empower contact-center agents, we propose dynamically soliciting agents' working and call-handling preferences and responding to them. It is natural to be concerned that traditional performance requirements will not be met by such a strategy. However, we develop dynamic call-routing algorithms that take account of agent preferences, allowing for adjustments in real time or near real time, while ensuring that traditional performance requirements are met. Thus we present evidence that the strategy is workable.

Keywords: customer contact centers, employee turnover, skill-based routing, agent preferences.

1. Introduction

Many services - from emergency to retail - are largely teleservices, in that the people who provide the service and the people who receive the service, herein called *customers*, are remote from each other, at least when the service is initiated. With a teleservice, the delivery of service is provided or enabled by a customer *contact center*; e.g., see Cleveland and Mayben (1997) and Gans, Koole and Mandelbaum (2003).

A primary resource in a contact center is the group of people who respond to these service requests - the customer service representatives, herein called *agents*. The classical contact center is the (telephone) call center, where the interactions are telephone calls, but the telephone is no longer the only means of interaction. Alternative media such as email, fax, web pages and instant messaging are on the rise. Contact centers are supported by quite elaborate information-and-communication-technology (ICT) equipment, such as a private branch exchange (PBX), an automatic call distributor (ACD), personal computers (PC's) and assorted database systems. The ICT technology reduces the requirement of spatial proximity: The ICT technology makes it possible to have virtual contact centers of various kinds, ranging from a small connected group of large call centers on different continents to a large number of individual agents working out of their own homes.

Contact centers usually handle several different kinds of service requests, herein referred to as *calls*. For example, telephone callers may speak different languages or may call about different promotions. Sometimes a single contact center will respond simultaneously to calls for several very different service providers. That occurs when service providers outsource their contact center to an independent specialist. Grouping together several different contact centers can be advantageous, because it facilitates economies of scale.

Thus, it is rarely possible or cost-effective to have every agent be able to handle every type of call. Thus, the agents tend to have different call-handling skills, in different combinations. Fortunately, contact centers can handle diverse calls with agents having diverse skill sets, because modern ACD's have the capability of assigning calls to appropriate agents: Contact centers have gone beyond the traditional load-based routing to perform *skill-based routing.* Traditional load-based routing is designed to ensure that calls are handled *promptly*; skill-based routing is designed to ensure that calls are not only handled promptly but are also resolved *properly.*

Contact centers maintain elaborate management-support systems to address important aspects of contact-center performance. First, there are *customer-relations-management* (CRM) systems, that help ensure that customers are being served appropriately and business objectives are being met. The CRM systems keep information about customers, including a history of previous interactions. By analyzing the data (via "business analytics"), CRM systems enable retail businesses to know the potential sales value of each interaction. Retail businesses can also ensure that important customer information is immediately available for the agent via screen-pop on the agent's computer screen just prior to establishing contact. The CRM systems also provide reports on how well business objectives are being met, by individual agents as well as by the contact center as a whole.

There also are elaborate *workforce-management* (WFM) systems, that aim to ensure that the right number of agents with the right skills are in the right place at the right time. The WFM systems have sophisticated algorithms and computer software to perform forecasting, staffing and scheduling. These three functions aim to determine the required number of agents with specified skills over different time periods (e.g. weeks, days, and half-hours). The WFM systems also help manage agent performance and compensation.

1.1. Performance Problems

Unfortunately, despite the complex systems described above, and perhaps partly because of those complex systems, the performance of contact centers frequently falls far short of expectations. Contact centers often are not able to meet business objectives; e.g., see Cleveland and Hash (2004). First and foremost, customers are often dissatisfied with their contact-center experience. Even though there are elaborate WFM systems, contact centers often are not staffed efficiently. Even though there is the capability of performing skill-based routing, often the routing does not achieve the desired results. There often are too many agents with some skills, but not enough agents with other skills, so that some agents are unproductive, while others are overworked. Despite all the management-support systems, service levels and other performance targets often are not met. As a consequence, some service requests are handled adequately, but too many others are not. Moreover, costs often are higher than expected, while revenues are lower. Despite all the sophistication, contact-center performance frequently does not nearly reach its potential.

There are clear signs that contact centers are not achieving their desired goals. There is strong evidence that it is currently difficult to maintain an energized, effective workforce; see Deery and Kinnie (2004), Frankel et al. (1999) and Holtgrewe et al. (2002). Indeed, there is strong evidence of agent job dissatisfaction: There are frequent reports of *four telling conditions*: (i) high turnover or churn, (ii) chronic absenteeism, (iii) frequent deviation from work schedules, and (iv) agent fatigue toward the end of the day. Even if it were not possible to improve the customer experience, it would be desirable to address these problems.

Indeed, many contact centers report agent-retention problems, expressed as high *churn*. Contact centers frequently report from 20% to 200% annual turnover. There are significant costs associated with high turnover, which are often not fully appreciated; see Bliss (2004). When considering the costs of employee turnover, it is natural to classify the costs, dividing them into two types: (i) transition costs and (ii) productivity costs.

Transition costs account for the per-agent cost of terminating the departing agent, recruiting and training the new agent to replace the departing one, and disruption costs associated with the change, such as the cost of hiring a temporary employee, and the costs of managers coping with the change, such as the cost of performing exit interviews, the administrative cost of stopping benefit deductions and starting benefit enrollments, and so forth. It has been estimated that transition costs alone can be as much as $100\% - 200\%$ of an agent's annual compensation.

Since new agents typically must undergo a significant start-up learning period in order to perform effectively, high turnover also tends to produce significant *productivity costs*. High turnover implies that too many agents are in the start-up learning period. More importantly, however, high turnover strongly indicates that many agents are dissatisfied with their job, and that job dissatisfaction is likely to make the agent a less effective worker; see Quiñones et al. (1995), Batt (2000), Mitchell et al. (2001) and Glebbeek and Bax (2004). (Whitt (2004) develops models to help analyze the productivity benefits of increased retention.)

High turnover directly hurts productivity, because it is necessary to recruit and train additional agents, to allow for the eventual attrition caused by the turnover. The impact of high turnover is actually greater than might be anticipated, because high turnover inevitably is associated with high uncertainty: It is very difficult to predict the long-term effects of recruiting and training programs, when we are unsure about the ultimate retention. Even though we can plan for the average loss, with all its attendant costs, the actual number of available agents with one skill or another may end up being far less or far more then intended. In order to hedge against potential shortfalls, it is necessary to recruit and train even more agents.

Closely related to turnover is *absenteeism*. Not only do many contact centers report high turnover, but they also report high absenteeism; many of the agents on the payroll, scheduled to work on a given day, do not actually show up. Like high turnover, absenteeism can be compensated for by scheduling extra agents in advance, but obviously that is at significant cost. Moreover, there is again significant uncertainty associated with absenteeism as well. If absenteeism tends to average about 10% per day, it usually fluctuates between about 5% and 15%, causing there to be, in the end, significant uncertainty about the staff level that will actually be present. As a consequence, despite the presence of highly sophisticated optimization algorithms to do the staff scheduling, there may be even more uncertainty about the number of agents available on a given day than there is uncertainty about the number of calls that will be received on that day, which customarily is regarded as a random phenomenon.

Closely related to turnover and absenteeism is *deviation from the work schedule.* Unfortunately, even the agents who remain employed and show up as scheduled may not adhere to the daily work schedule. They may not be logged in and available to handle service requests when they are supposed to be. Or they may be logged in, but not actually available. Just like turnover and absenteeism, schedule deviation degrades the agent availability to respond to calls. Again, one may plan for such service degradation (and the associated cost), but there is yet another added uncertainty, which reduces the effectiveness of planning.

Finally, there is *agent fatigue*, which is a decline in productivity over the day. In many contact centers, the productivity of many agents declines during the day, often dropping off precipitously toward the end of the day.

1.2. Root Causes

When these agent-performance problems are discussed by contact-center management, we often hear direct or indirect criticism of the agents. The agents are criticized for their poor work ethic. We hear management express their frustration. They complain that their employees are not motivated and do not work efficiently. Frequently, the management response is to put even greater pressure on the agents. For example, in an attempt to achieve better adherence to work schedules, contact-center management might adopt a pervasive call monitoring system, recording every call.

However, we believe that such responses may be a mistake. To put the issue in perspective, it is helpful to recall the long history of the quality movement in manufacturing, led by such pioneers as W. Edwards Deming and Joseph M. Juran; see Deming (2000) and Juran and Godfrey (1999). Similar failures in manufacturing were blamed on inadequacies of the workers, but experience and statistical evidence showed that the key to improved performance is usually improved processes. That is highlighted by the 85/15 rule of Deming and Juran, which says that at least 85% of the problems are system problems, while less than 15% of the problems are due to the workers.

It is important to recognize that the fundamental problem in contact centers may be with contact-center processes, not the agents themselves. The easily measurable quantities - turnover, absenteeism, schedule deviation and fatigue - are all indications that agents are dissatisfied with their job. Indeed, research studies show that contact-center agents frequently do not have a sense of wellbeing in the workplace; see Singh et al. (1994), Batt (2000, 2002), Ruyter et al. (2001), Holman (2002, 2003) and Witt et al. (2004). When agents are required to respond to telephone calls by carefully following prepared scripts, the agents inevitably feel a loss of control, a lack of autonomy. When contact-center management adopts a pervasive monitoring system, recording every call, the agents inevitably experience anxiety and stress; see Witt et al. (2004). When agents are simultaneously trying to satisfy the customer's needs and trying to respond to management pressure to complete the calls more quickly, the agents inevitably experience role ambiguity; see Ruyter et al. (2001). More generally, the current contact-center working environment, be it in small cubicles or alone at home, too frequently tends to cause boredom, loneliness, passivity and lack of recognition.

1.3. Empowering the Agents

Indeed, it is widely recognized that agent job dissatisfaction is a serious contact-center problem. The importance of agent motivation and compensation is frequently emphasized; see Cleveland and Hash (2004). Others have said that contact-center managers need to empower the agents; e.g., see Spraetz (2004). However, the traditional responses may not be enough. Radical changes may be needed.

In this paper we aim to contribute to significant work redesign in contact centers. Our thesis is that, in many circumstances, contact-center performance can be improved by enhancing agent participation. That follows the general management principle that employee participation in decision making and problem solving usually improves the quality of work; again see Deming (2000) and Juran and Godfrey (1999); also see Hackman and Oldham (1976), Karasek and Theorell (1990), Bowen and Lawler (1995), Collins (1996), Rafig and Pervaiz (1998) and Corsun and Enz (1999).

Thus we seek to *empower* the agents. Certainly, much can be accomplished within existing contact centers. For example, management can give agents greater input in determining work schedules and meet regularly with agents seeking their input, as suggested by Spraetz (2004). Much is accomplished by simply recognizing that it is important to treat one's fellow employees with mutual respect.

However, we believe that more substantial changes may be needed. We suggest considering structural changes in the way contact centers operate. Clearly, staffing and call routing are major decision processes in contact centers. We aim to improve agent participation by soliciting agent preferences about working and call routing and responding to them. We thus propose going beyond skill-based routing to achieve *preference-based routing* and *preference-based staffing.* In addition to responding to calls promptly and properly, we want to increase customer and agent satisfaction from the contact experience.

The first step is to recognize that agents are individual people, with their own preferences. By responding to agent working and call-handling preferences, contact-center management will demonstrate that agent efforts are valued and appreciated. They will give the agents a sense of belonging. They will make it clear that the agents are part of the overall contact-center team, working towards a common goal. Moreover, the agents may have important insight about system needs and their own effectiveness.

By establishing a *dialog* with the agents, management will be better able to energize the agents. Management may give incentives to the agents, encouraging them to handle calls needing greater attention. In that way, staffing and routing may be achieved by a dynamic *game* or *market*, with adjustments in real time or near real time. One implementation involves loyalty points; the agents may accrue loyalty points for responding favorably to management incentives, working at desired times and handling desired calls. These points may be redeemable for rewards. By enabling communication among agents, management will provide a way for agents to cooperate, working together in *teams.* Consequently, agents may experience much less isolation and loneliness in the workplace.

Of course, there should be concern that the proposed actions might backfire: By allowing agent preferences to influence staffing and routing, there is a risk that contact-center performance may actually degrade. Allowing individual agents to act independently, no matter how well-meaning, does not ensure good system-wide performance. Thus, part of our task is to provide suitable management controls to help ensure that contact-center performance is satisfactory.

Mostly, however, we think there is more to gain than lose. In making this proposal, we do so with the conviction that tight management control of staffing and call routing is actually not required. In Wallace and Whitt (2004), we found that both staffing and routing could be accomplished remarkably well with relatively little effort provided there is a minimal amount of cross training. A little flexibility goes a long way: When each agent has only two skills, in appropriate combinations, it is often possible to staff primarily as if the contact center were one large contact center in which all agents had all skills. Moreover, in such an environment, it suffices to use a relatively primitive routing algorithm. Supporting theoretical results have been established by Armony et al. (2004) and Chevalier et al. (2004).

Thus, we believe that there is little to lose from doing what the agents want. Moreover, agents may respond even beyond management expectations. Agents may have access to information sources, such that their actions - though inconsistent with expressed management priorities - may subsequently be recognized as effective.

1.4. New Call-Routing Algorithms

Nevertheless, we accept that a major burden on us, in proposing this new strategy, is to show that it is workable. We need to show that we can respond to agent preferences, while still meeting traditional performance targets.

Our approach is perhaps best communicated via a simple example. Suppose that we have a contact center in which 100 agents can all respond to telephone calls in either French or English. Suppose that some agents strongly prefer responding to French-speaking calls, while others strongly prefer responding to English-speaking calls. Moreover, suppose there is a balanced load of the two kinds of service requests, so that we need 50 agents primarily devoted to responding to French-speaking calls, and 50 agents primarily devoted to responding to English-speaking calls, in order to meet traditional performance targets.

Consider two situations: First, suppose that all 100 agents prefer responding to French-speaking calls. Then clearly we have a problem. We cannot meet our performance targets if we let them do that. In that situation there is relatively little we can do; we will have to require that 50 agents be primarily devoted to responding to each group of calls.

On the other hand, suppose that 50 agents strongly prefer responding to French-speaking calls, while the other 50 strongly prefer responding to English-speaking calls. The traditional skill-based routing will have 50 agents primarily responding to each group, but no attention will be paid to agent preferences, so that there will be a preference mismatch, based on a random assignment. With preference-based routing, we will be able to let all the agents do what they want, and do not worse. Our goal is to show that, more generally, we can pay attention to agent preferences, while still meeting traditional performance constraints.

Thus, the remainder of this paper is devoted to developing new call-routing algorithms that take account of agent preferences. (The approach will also apply more generally to staffing, i.e., whether or not the agent is working at all, but we will focus on routing.) Our goal here is to develop both specific algorithms and a framework for developing other algorithms. We aim to be illustrative rather than definitive; we aim to show that effective preference-based routing (and staffing) can be done. We intend to study the performance of these algorithms by computer simulation, but we do not do that here. We leave simulation studies for future research.

We remark that conducting successful simulation studies is challenging, because a main goal of the innovation is to induce new *behavioral* responses: We want the agent to have a better sense of wellbeing in the workplace, and we want more satisfied customers as a result. Thus, the important intended benefits are not meeting conventional performance targets (e.g., service-level constraints, such as 80% of all calls answered within 20 seconds), even though we should ensure that they are indeed met, but instead are increased agent and customer satisfaction.

Of course, these feelings should have quantifiable effects: As a result of increased agent job satisfaction, we would like to see less agent turnover and absenteeism; as a consequence of having more satisfied customers, we would like to see more return business from customers and increased customer demand over time. Simulation is a standard tool for analyzing contact centers, as can be seen from Anton et al. (1999), Brigandi et al. (1994), Leamon (2004) and Wallace and Whitt (2004), but we are unaware of any previous simulation studies that have focused on these behavioral issues in contact centers. Previous simulation studies have focused on conventional congestion measures, such as customer abandonment and delays.

We envision two possible implementations for the routing: The first, less invasive, implementation is to *work with the existing call-routing algorithm* in the ACD. Given an existing algorithm, we need a way to apply agent preferences to modify (set) the *parameters* of the existing algorithm, in order to affect the routing. The second, more invasive, implementation involves *inventing an entirely new call-routing algorithm*, to replace whatever call-routing algorithm is in the current ACD. That is tantamount to inventing a new ACD. However, from an algorithmic perspective, the second implementation is easier, because we do not need to coordinate with any existing algorithm. In this paper we will provide methods for carrying out *both implementations*, but for the following discussion assume that we are considering the first implementation.

Even if we agree that preference-based routing is a good idea, there are serious challenges in its implementation: In particular, there are *information technology* (IT) challenges: We need to have a control point, such as a *preference server*, where we can orchestrate preference-based routing. Second, we need to establish communication links connecting the agents, the ACD, existing management-support systems (CRM and WFM) and this preference server. The agents should be able to open *preference windows* on their computer screens, so that they can receive information from the preference server. In brief, we need appropriate IT so that the agents can indeed express their preferences in an effective dynamic manner. In this way, agents may also be able to communicate with each other, so that they can exploit *teamwork*. Finally, we need a way to apply the received agent preferences in order to influence or modify the existing (skill-based) routing algorithm.

Here we do not discuss the IT challenges; in this paper we focus on the routing algorithm. If we are not going to invent an entirely new routing algorithm, then the possible ways to implement preference-based routing depend on the current (skill-based) routing algorithm in place. We believe that it is possible to incorporate preference-based routing within the framework of many different existing routing algorithms (without changing too much). To illustrate what we believe is possible, and to be concrete, we define a candidate skill-based-routing algorithm employing a priority matrix. The matrix element $P_{i,j}$ gives the call-handling priority for agent i to handle call type j, assuming that the calls have been classified into types. We then show how to use agent preferences to select appropriate entries in this priority matrix. In other words, we use agent preferences to set the parameters of that existing SBR algorithm. Staffing is also affected, because agents may indicate that they prefer not to work at all for some period of time.

For the second implementation, we develop an entirely new routing algorithm, to replace whatever was being used by the ACD. We invent a new routing method, called *value-based routing* (VBR), which depends on a *value* $v_{i,j}$ attached to agent i handling a call of type j. Value-based routing is convenient for us, because the values provide a means for the agents to express their preferences. We let agents specify their preferences via values. Then we provide a method to combine agent values with management values to obtain composite values. And then we use the VBR algorithm with these composite values to route the calls.

We also use the values to generate the priority matrix in our initial implementation. For the first implementation of preference-based routing, we let agents specify their preferences via values. Then, as before, we provide a method to combine agent values with management values to obtain composite values. Now, instead of performing direct VBR, we use those values to construct an appropriate priority matrix to use with the "existing" routing algorithm in the ACD. We call the VBR algorithm implemented through the priority matrix an *indirect VBR algorithm.*

Value-based routing itself is important, because it emphasizes focusing on the *value* of the service provided - the true quality of service (QoS) - rather than conventional *queueing-theory performance measures* related to congestion, such as abandonment rate, average speed to answer and service level. Moving from traditional queueing-theory performance measures to value-based performance measures in contact centers itself can be a major advance. However, here we use the VBR algorithms as a framework to develop PBR algorithms. The PBR algorithms enable the agents to participate in call-routing decisions. We thus provide a means to genuinely empower the agents.

We envision the VBR and PBR algorithms as being *dynamic feedback algorithms*, with adjustments being made by all parties in near real time. First, management adjusts its expressed value assignments in response to the observed performance of the routing, including both the network conditions and the way business objectives are being met, as well as previous agent preferences. Moreover, management may communicate incentives to the agents. In response to management input, and possibly input about network conditions and the way business objectives are being met, the agents adjust their preferences and the cycle continues. During their deliberations, the agents may possibly interact with each other, perhaps working together in teams. We envision all parties learning and adjusting over time. For example, agents may acquire *earned empowerment*: Management may adjust the weight given to each agent's preferences based on the agent's past record of performance [3].

1.5. The Rest of this Paper

Here is how the rest of this paper is organized: First, in Section 2 we specify the candidate SBR algorithm based on an agent priority matrix P, which is intended to illustrate what is done today in the better SBR algorithms. Next, in Section 3 we develop the new *direct value-based-routing* (VBR) algorithm. In Section 4 we introduced the alternative *indirect VBR algorithm* in which the values are converted into priorities for handling different types of calls, thereby affecting the existing SBR algorithm described in 2. The indirect VBR algorithm is also appealing because it reduces the complexity.

Having developed the VBR algorithms in Sections 3 and 4, we next consider ways to incorporate agent preferences. In Section 5 we develop *preference-based routing* (PBR) algorithms in the framework of the two previous VBR algorithms. (Preference-based staffing arises as a special case, because agents can indicate their preference for working at all.) We do so by expressing agent preferences in the form of values. We take several measures to ensure that management can maintain control in various ways, if it is deemed necessary. In particular, we discuss ways to *combine* management values with agent values to obtain overall *composite values*, to use in the PBR algorithms. We develop *weighted values* that allow management to gradually adjust the amount of emphasis given to agent preferences. That allows management to rapidly or slowly increase or decrease the weight given to agent input, as circumstances dictate. We include a simple specific PBR algorithm in Section 5.3. We describe the overall iterative feedback process in more detail in Section 6 and We briefly state conclusions in Section 7.

2. A Priority-Based SBR Algorithm

In this section we introduce a priority-based routing algorithm, which we regard as illustrative of current SBR algorithms. Indeed, the SBR algorithm here is similar to the routing algorithm used by Wallace and Whitt (2004), which in turn is a variant of SBR algorithms in practice. We will introduce a variety of additional parameters, which may go beyond existing SBR algorithms, but which we believe can help management produce effective VBR and PBR algorithms.

2.1. The Framework

We consider a contact center with n agents, who respond to m types of inbound calls. We primarily base routing decisions on an *agent priority matrix* P. The matrix element $P_{i,j}$ gives the priority level for agent i to handle call type j. The i^{th} *row* of the matrix P determines the routing for agent i, while the j^{th} *column* of the matrix P determines the routing for any call of type j.

We assume that $P_{i,j}$ is a positive real number for each call type j that agent i has the required skill to handle, with higher numerical values indicating higher priority. (Usually the possible priority values will be only a few integers.) Let $P_{i,j} = 0$ for all call types j that agent i can respond to only when the system is overloaded. Let $P_{i,j} = -1$ for all call types j, if any, that agent i cannot respond to under any condition.

Here we take the individual values $P_{i,j}$ as given. Moreover, at this point we are concerned with routing, not staffing. We assume that the important staffing step has been completed, so that there is an appropriate number of agents with the right skills. More generally, we want to make the best possible use of the available agents, whatever demand and staffing we are faced with. However, routing is intimately linked to staffing; it is invaluable to have the flexibility to rapidly change the staffing, if needed, as discussed in Whitt (1999).

For call routing, we must make two important decisions: First, we must decide what to do *when a new call arrives*; second, we must decide what to do *when an agent becomes free* after completing a call. Since we will introduce time thresholds, we also need to make decisions when these time thresholds are passed. We develop a class of algorithms depending on parameters to be specified.

We initially specify what we mean by agent i being *available*: We say that agent i is *available* at time t if (i) agent i is logged in to the ACD at time t, (ii) agent i is not already handling a call at time t, and (iii) agent i has been busy less than a proportion β_i of the time during the last interval of time of length λ_i minutes. Let $I_i(t)$ be the total length of time that agent i has been idle in the time interval $[0, t]$. Then the proportion of time that agent i has been idle in the interval $[t - \lambda_i, t]$ (where $\lambda_i < t$) is

$$A_i(t, \lambda_i) \equiv \frac{I_i(t) - I_i(t - \lambda_i)}{\lambda_i}, \quad t \geq \lambda_i \ . \tag{2.1}$$

We thus say that agent i is available at time t if $A_i(t, \lambda_i) \leq \beta_i$. (Clearly, we can remove the last condition by simply setting $\beta_i = 1.0$.)

We also specify a *waiting-time threshold* $\theta_{i,j}$ for a call of type j, after which agent i, if available, will respond to that call, even if call type j is not especially appropriate for agent i. The agent might then only take appropriate information to facilitate calling the customer back at a later time. (Again, we can eliminate this opportunity by simply setting $\theta_{i,j} = \infty$.)

Thus, our SBR algorithm has *four kinds of parameters*: the $n \times m = nm$ priority values $P_{i,j}$, the n busy-time-proportion bounds β_i, the n lengths λ_i of the time intervals for the busy-time bounds and the $n \times m = nm$ waiting-time thresholds $\theta_{i,j}$.

2.2. The Priority-Based SBR Algorithm

Given the agent-priority matrix P and the associated control parameters, we can now specify our priority-based SBR algorithm. There are four situations in which action should be taken:

1. When a new type-j call arrives, route (assign) the call to the available agent i with the highest positive priority level $P_{i,j}$. Break ties with the *longest-idle-agent-routing* (LIAR) policy: route the call to that agent, among all those available agents with that best priority number, that has been idle the longest. If there is no available agent i for which $P_{i,j} > 0$, then put the arriving call at the end of a queue of waiting calls of type j. Let there be a separate queue for each of the m call types, with each queue served in the FIFO order.

2. When agent i becomes free, after completing a call, first check to see if agent i satisfies the availability requirement: If agent i has been busy more than a proportion β_i of the time in the last time interval of length λ_i, let agent i go idle. Otherwise, look for waiting calls to assign to agent i. Let the candidate calls to assign to agent i be from the front of the m call-type queues (the calls of each type that have been waiting the longest). Assign the waiting call of type j (in the front of its queue) with the highest priority. Break ties by assigning the call that has been waiting the longest. If there is no waiting call for a call type j with $P_{i,j} > 0$, then look to see if there are any waiting calls, of any type j for which $P_{i,j} = 0$, that have been waiting for more than the time threshold $\theta_{i,j}$. Among all these calls, answer the one for which $w_j - \theta_{i,j}$ is largest, where w_j is the time spent waiting by the type-j call that has been waiting the longest.

3. When agent i becomes available, after sitting idle, assign a call to agent i, just as when the agent becomes free after completing a call (as in step 2 above, but considering the current state of the system).

4. When the waiting time of a type-j call hits a threshold $\theta_{i,j}$, look to see if agent i is available and if $P_{i,j} \geq 0$. If so, assign that call to agent i . Break ties among candidate agents by using the LIAR policy. Ties among waiting customers should occur only negligibly; to be specific, break them by numerical index of call type and then numerical index of position in queue. ■

3. Direct Value-Based Routing

We now propose a new value-based algorithm to replace the SBR algorithm defined above. We again consider a contact center with n agents, who respond to m types of calls. We assume that a *value* $v_{i,j}$ (a real number) has been assigned for agent i handling a type-j call.

In a *sales environment*, the value $v_{i,j}$ might be the *expected dollar value* of having agent i handle the type-j call. These expected dollar values might be estimated from data from the CRM system. On the other hand, these values $v_{i,j}$ might simply represent quantitative expressions of the "value" management attributes to having agent i handle a call of type j. For example, those value scores might be based on management estimation of agent skill levels. It is natural to go beyond simply recognizing whether or not an agent has the skill to speak French; we might try to assess how well the agent can handle French-speaking calls. Unlike expected revenue, values clearly apply in any contact-center environment, e.g., in emergency services as well as in retail. Finally, the value might be a direct expression of agent preferences or be a composite combined expression of management and agent call-handling values, as we will propose in Section 5 below. We believe that "value" is good terminology, because it encompasses many different dimensions we might want to consider.

3.1. A Performance Target

Before defining any specific routing algorithm, it is appropriate to consider the contact-center goals. Thus we start by formulating a *performance target* that can be used to judge candidate routing algorithms, including the ones we present. However, we regard what we do as only illustrative. In any contact-center application, it is natural to start by carefully considering the overall goals, and then try to formulate a meaningful performance target. Moreover, it is good to communicate the goals and the performance target to all employees, so that the direction is clear to one and all.

Given that we are starting with values, it is natural to base the target directly on the values $v_{i,j}$, but we may require that relatively few calls abandon and that answered calls be answered promptly. We think of value-based routing still aiming to meet traditional performance targets. Thus, we ascribe a cost (negative value) $c_{L,j}$ to each class-j call that is lost due to customer abandonment and a cost $c_{D,j}$ to each served class-j call that has to wait more than δ_j seconds. Again this is only meant to be illustrative; we might to pay attention to exactly how long abandoning calls waited before they abandoned, and we might pay attention to exactly how long served calls had to wait.

We now define a specific *total-value function.* To do so, we specify a time interval over which performance is to be judged, e.g., a typical day. Let $N_{i,j}$ be the number of type-j calls answered by agent i within γ_j seconds during the specified time period. Let L_j be the number of type-j calls to be lost because of customer abandonment within the designated time period, and let D_j be the number of answered class-j calls delayed beyond δ_j seconds ($\delta_j > \gamma_j$) within the designated time period. Clearly, the quantities $N_{i,j}$, L_j and D_j should be regarded as random variables, which depend on the unknown pattern of arrivals and service times as well as the routing policy used. For each routing algorithm, let the *total value* gained from the routing algorithm under consideration be the expected total value, i.e.,

$$\mathbf{V} \equiv \sum_{i=1}^{n}\sum_{j=1}^{m}(E[N_{i,j}] \cdot v_{i,j}) - \sum_{j=1}^{m}([E[L_j] \cdot c_{L,j}] + [E[D_j] \cdot c_{D,j}]) \ . \tag{3.1}$$

We may then take as our overall goal the maximization of the total value expressed in (3.1). We can estimate the expected values of the three random variables in (3.1) by first specifying a stochastic model for the arrivals of call types and their service-time distributions, and then applying computer simulation. For any given stochastic model, we can use computer simulation to evaluate the performance of each alternative routing algorithm, e.g., as in Anton et al. (1999), Brigandi et al. (1994) or Leamon (2004). We present candidate algorithms below.

Before going on, we observe that we might well want a more elaborate model. By including losses and delays in the performance target, we have accounted for important goals in traditional load-based routing, but in that direction we could provide more detail, e.g., by including service level. By including values depending on the (i, j) pair, we have accounted for important goals in skill based-routing, but we could go further: We have not directly evaluated whether or not each call was handled properly; we have not indicated whether or not the service request was properly resolved. Just assigning a call of type j to agent i clearly does not directly guarantee that we will receive a fixed value $v_{i,j}$ from each interaction of that kind. Nevertheless, we think of the performance target (3.1) as way to compare alternative call-routing algorithms. We aim to present algorithms that can perform well by that criterion, provided that the algorithm parameters are set appropriately.

3.2. The Parameters

Here we take the individual values $v_{i,j}$ as given. Moreover, as before, here we are concerned with routing, not staffing. For call routing, we must make the same two important decisions considered above: First, we must decide what to do *when a new call arrives*; second, we must decide what to do *when an agent becomes free* after completing a call. Since we introduce time thresholds, just as before, we also need to make decisions when these time thresholds are passed. We develop a class of algorithms depending on parameters to be specified. We believe that good performance can be achieved within the class we propose by an appropriate choice of these parameters. Unfortunately, however, our framework evidently does *not* contain the *optimal* routing algorithm (for our target). Identifying an optimal routing algorithm is evidently beyond current capabilities. We aim to develop an effective (reasonably good) routing algorithm. The performance of the algorithms can be judged by comparing them to other algorithms, using computer simulation.

As before, we have the agent-availability parameters β_i and λ_i. We also have *waiting-time threshold* $\theta_{i,j}$ for a call of type j, after which agent i, if available, will respond to that call, even if call type j is not especially appropriate for agent i. We also want to take extra measures to ensure that there is a good match between calls and agents. Hence we have lower and upper bounds on assignments, based on the values. For agent i, there are lower and upper *agent value bounds* α_i^L and α_i^U with $\alpha_i^L < \alpha_i^U$; for call type j, there are lower and upper *call-type value bounds* τ_j^L and τ_j^U with $\tau_j^L < \tau_j^U$. Their use will be specified below.

Thus, our direct-VBR algorithm has *eight kinds of parameters*: the $n \times m = nm$ values $v_{i,j}$, the n busy-time-proportion bounds β_i, the n lengths λ_i of the time intervals for the busy-time bounds, the $n \times m = nm$ waiting-time thresholds $\theta_{i,j}$, the $2n$ agent value bounds α_i^L and α_i^U, and the $2m$ call-type value bounds τ_j^L and τ_j^U.

3.3. The Direct VBR Algorithm

There are four situations in which action should be taken. Here is what to do in each case:

1. When a new type-j call arrives, route (assign) the call to the available agent i with the highest positive value $v_{i,j}$ among those agents i with $\tau_j^L \le v_{i,j} \le \tau_j^U$. Break ties with the *longest-idle-agent-routing* (LIAR) policy: route the call to that agent, among all those available agents with maximal value, that has been idle the longest. If there is no available agent i for which $\tau_j^L \le v_{i,j} \le \tau_j^U$, then put the arriving call at the end of a queue of waiting calls of type j. Let there be a separate queue for each of the m call types. Customers from each (call-type) queue are served in *first-in first-out* (FIFO) order.

2. When agent i becomes free, after completing a call, first check to see if agent i satisfies the availability requirement: If agent i has been busy more than a proportion β_i of the time during the last time interval of length λ_i, let agent i go idle. Otherwise, look for waiting calls to assign to agent i. Let the candidate calls to assign to agent i be from the front of the m call-type queues (the calls of each type that have been waiting the longest). Assign the waiting call of type j (in the front of its queue) if $v_{i,j}$ is the largest positive value among call types j with waiting calls satisfying $\alpha_i^L \le v_{i,j} \le \alpha_i^U$. Break ties by assigning the call that has been waiting the longest. If there is no waiting call for a call type j with $\alpha_i^L \le v_{i,j} \le \alpha_i^U$, then look to see if there are any waiting calls, of any type j, that have been waiting for more than time $\theta_{i,j}$. Among all these calls, answer the one for which $w_j - \theta_{i,j}$ is largest, where w_j is the amount of time spent waiting time by the type-j call that has been waiting the longest (necessarily associated with the call at the front of its queue).

3. When agent i becomes available, after sitting idle, assign a call to agent i, just as when the agent first becomes free after completing a call (as in step 2 above, but considering the current state of the system).

4. When the waiting time of a type-j call hits a threshold $\theta_{i,j}$, look to see if agent i is available. If so, assign that call to agent i. Break ties among agents by using the LIAR policy. Ties among waiting customers should occur only negligibly; to be specific, break them by numerical index of call type and then numerical index of position in queue. ■

That completes the specification of our direct VBR algorithm. Both an advantage and a disadvantage of this direct VBR algorithm is that it has a fairly rich collection of parameters that can be (needs to be) set in order to achieve good performance. Computer simulation can be used to identify good parameter settings. Initially we can ignore the value bounds α_i^L, α_i^U, τ_j^L and τ_j^U, but experience indicates that they can often be exploited in order to improve overall performance. They can prevent assignments that are feasible, but not desirable. In many scenarios, it is sometimes advantageous to let an agent sit idle when there are waiting calls. If the agent is highly skilled, we may want to save the agent in reserve for future important calls. On the other hand, if the agent is not highly skilled, then we may prefer to wait until a new more-highly-skilled agent becomes available to handle the call. We might want to restrict the calls that an agent handles, at least as long as customers are not waiting excessively.

4. An Indirect Value-Based-Routing Algorithm

In this section we introduce an alternative indirect VBR algorithm. We start with values as in Section 3, but then we convert those values into an agent priority matrix P as in Section

2. We thus show how to incorporate value-based routing into the SBR framework based on an agent priority matrix.

There are clearly many ways to transform values into priorities. A simple direct way is to just let $P_{i,j} = v_{i,j}$ for all i and j. However, we aim for a more elementary resulting routing algorithm. Specifically, here (i) there are fewer categories and (ii) each agent has only a few active skills. Here we work with four priority levels; in general, it might suffice to have fewer, or it might be better to have more. Here we assume that each agent has exactly one active skill at each of the top three priority levels. The agent may have the ability to handle other types of calls; the *active skills* are the skills that the agent will mostly be asked to handle during the time period those skills have been designated as the active skills. In Wallace and Whitt (2004) it was found that it often suffices for agents to have only two skills, in appropriate combinations. It may even suffice for only some of the agents to have two skills, with the rest having only one.

4.1. The Priority Matrix

Our indirect VBR algorithm can start from the same value function $v \equiv v_{i,j}$ in Section 3, but it produces a more elementary routing scheme based on only the four priority levels. In particular, routing will be determined by an $n \times m$ *priority matrix* P that assigns one of four priority levels to each agent-call-type pair (i, j), $1 \leq i \leq n$ and $1 \leq j \leq m$: For each agent i and call-type j, the matrix element $P_{i,j}$ gives the priority level for agent i handling a call of type j.

We assume that $P_{i,j}$ can assume one of the four different numbers: 0, 1, 2, and 3. The highest (top, preferred) priority level is 3; the second highest is 2 and so forth. If $P_{i,j}$ is not assigned a number or is assigned the number -1, then agent i is deemed unqualified to handle calls of type j. We can specify which call types the agents can handle with a $n \times m$ *skill matrix* $S \equiv (S_{i,j})$. If agent i can handle service request j, then $S_{i,j} = 1$; otherwise, $S_{i,j} = 0$. Some very simple skill-based routing algorithms in practice just let the $P_{i,j} = S_{i,j} - 1$ for all i and j. However, experience indicates that performance can be improved by imposing additional structure. That hypothesis can be tested with simulation.

A standard skill-based-routing (SBR) algorithm can be obtained by directly defining the priority matrix P. An *indirect VBR algorithm* is obtained from a direct VBR algorithm by mapping the set of nm values $\{v_{i,j} : 1 \leq i \leq n, 1 \leq j \leq m\}$ into the corresponding set of nm priorities $\{P_{i,j} : 1 \leq i \leq n, 1 \leq j \leq m\}$. A simple way to do this is to choose three *boundary* *numbers* b_1, b_2 and b_3 with $0 < b_1 < b_2 < b_3$, and let

$$\begin{aligned} P_{i,j} &= 3 \quad \text{if} \quad v_{i,j} \geq b_3, \\ P_{i,j} &= 2 \quad \text{if} \quad b_2 \leq v_{i,j} < b_3, \\ P_{i,j} &= 1 \quad \text{if} \quad b_1 \leq v_{i,j} < b_2, \\ P_{i,j} &= 0 \quad \text{if} \quad 0 < v_{i,j} < b_1 \ . \end{aligned} \tag{4.1}$$

If we stipulate that the set of possible values is $\{0, 1, 2, 3\}$, then we can directly identify $P_{i,j}$ with $v_{i,j}$, so we really have not done much yet. However, we continue to propose a more elaborate scheme that gives each agent exactly three non-zero (active) skills. We assume that each agent has one priority-3 skill, one priority-2 skill, one priority-1 skill and any number of priority 0 skills, where agent i having *skill* j at *priority level* k means that call type j is of priority level k for agent i. If $P_{i,j}$ is not assigned a value in the set $\{0, 1, 2, 3\}$, then agent i does not have the skill to handle call type j.

We are aiming to assign calls from the top three priority classes, but the agent can handle priority-0 calls as well, if necessary. We call the three high-priority skills *active skills*; those are the skills that are intended for the agent to be using. In restricting the number of active skills we are drawing on the results of Wallace and Whitt (2004), which show that it suffices for each agents to have only two skills, if the agents have appropriate skills in appropriate combinations. But when we limit the number of skills in this way, it becomes challenging to make a good skill assignment. To meet that goal, we suggest using mathematical programming; see Hillier and Lieberman (2003) and Winston and Venkataramanan (2002).

For the mathematical program, we assume that the priority levels 3, 2 and 1 each appear precisely once in each row, while 0 can appear any number of times. That is, each agent has one priority-3 skill, one priority-2 skill, one priority-3 skill and any number of priority 0 skills, where agent i having *skill* j at *priority level* k means that call type j is of priority level k for agent i. We construct the priority matrix P by applying a mathematical program, using the values as input parameters.

4.2. Constraining Parameters

In this setting with the agent-priority matrix P, we also use the busy-time-proportion bounds β_i, the lengths λ_i of the time intervals for the busy-time bounds, and the waiting-time thresholds $\theta_{i,j}$, but we no longer need the four value bounds α_i^L, α_i^U, τ_j^L and τ_j^U. With the indirect VBR algorithm, we directly specify the call types an agent can serve, so that we can prevent inappropriate feasible assignments.

We assume that we have specified staffing requirements in terms of: (i) the total number n of agents, (ii) lower and upper bounds n_j^L and n_j^U on the number of agents for which call type-j is priority 3 (top priority), (iii) lower and upper bounds $n_{j,k}^L$ and $n_{j,k}^U$ on the number of agents for which call type j is priority 3 and call type k is priority 2 (where it is assumed that j and k are distinct) and (iv) lower and upper bounds $n_{j,k,l}^L$ and $n_{j,k,l}^U$ on the number of agents for which call type j is priority 3, call type k is priority 2 and call type l is priority 1 (again where j, k and l are distinct).

For example, a systematic way to specify the staffing parameters n, $n_{j,k}$ and $n_{j,k,l}$ was specified in Wallace and Whitt (2004). First, we can obtain the total number of agents, n, by staffing as if all agents had all skills, or as if we had a multi-server queue with a single call type. Second, we can use a square-root-staffing rule to choose values of n_j: we can let $\bar{n}_j = R_j + x\sqrt{R_j}$, where R_j is the offered load for call-type j and x is chosen so that $\bar{n}_1 + \cdots + \bar{n}_m = n$; then round to get n_j. Third, we can let $\bar{n}_{j,k}$ be proportional to the product $n_j \cdot n_k$, and then round to get $n_{j,k}$.

Let the *decision variables* in our integer program be $x_{i,j,k,l}$; let $x_{i,j,k,l} = 1$ if agent i has call type j as priority 3, call type k as priority 2 and call type l as priority 1; let $x_{i,j,k,l} = 0$ otherwise. Let the value of such an assignment be $v_{i,j,k,l}$, which we assume is a nonnegative number. We assume that these values are specified. There are many possible ways to assign such values.

To establish a close connection to the previous section, we show how these values $v_{i,j,k,l}$ can be defined in terms of the more elementary values $v_{i,j}$ defined in Section 3 above. In particular, we can let

$$v_{i,j,k,l} = v_{i,j} + \epsilon_{i,j,k} v_{i,k} + \epsilon_{i,j,k,l} v_{i,l} \ , \tag{4.2}$$

where $\epsilon_{i,j,k}$ and $\epsilon_{i,j,k,l}$ are numbers satisfying

$$0 < \epsilon_{i,j,k,l} < \epsilon_{i,j,k} < 1 \quad \text{for all} \quad i, j, k \quad \text{and} \quad l \ . \tag{4.3}$$

We think of $\epsilon_{i,j,k}$ as being roughly equal to the ratio of the number of priority-2 calls of type k handled by agent i to the number of priority-3 calls of type j handled by agent i. Similarly, we think of $\epsilon_{i,j,k,l}$ as being roughly equal to the ratio of the number of priority-1 calls of type l handled by agent i to the number of priority-3 calls of type j handled by this agent.

For example, we might simply set $\epsilon_{i,j,k} = 0.20$ for all (i,j,k) and $\epsilon_{i,j,k,l} = 0.05$ for all (i,j,k,l). On the other hand, these ratios might be defined iteratively: We can initially estimate these proportions and then perform a simulation with the specified parameters, and see what proportions occur. We would then reset the proportions to the revealed values, and perform another simulation, and continue until there is close agreement. Because j is priority 3, k is priority 2, and l is priority 1, we expect the ratios to satisfy (4.3), but we could allow violation of condition (4.3) as well.

The first (main) point for the indirect VBR algorithm being developed here is that we have values $v_{i,j,k.l}$ for all four-tuples (i,j,k,l). The second point is that we have some way to convert the values $v_{i,j}$ defined in Section 3 into the more complex values $v_{i,j,k,l}$. We are strongly interested in being able to relate the indirect VBR algorithm in this section to the direct VBR algorithm in the previous section. Hence, it is important that here, too, we can start with the same values $v_{i,j}$ that were central to the direct VBR algorithm in Section 3.

Here we have the following additional parameters: the single parameter n (as before), the $2m$ parameters n_j^L and n_j^U, the $2 \times m \times (m-1) = 2m(m-1)$ parameters $n_{j,k}^L$ and $n_{j,k}^U$, the $2 \times m \times (m-1) \times (m-2) = 2m(m-1)m-2)$ parameters $n_{j,k,l}^L$ and $n_{j,k,l}^U$, the $m \times (m-1) = m(m-1)$ parameters $\epsilon_{j,k}$ in (4.2) and the $m \times (m-1) \times (m-2) = m(m-1)m-2)$ parameters $\epsilon_{j,k,l}$ in (4.2). (For simplicity, those parameters might be chosen to be independent of the indices; e.g., we might let $\epsilon_{j,k} = 0.20$ for all (j,k) with j and k distinct and $\epsilon_{j,k,l} = 0.05$ for all (j,k,l) with j, k and l distinct.

4.3. The Integer Program

Given the values $v_{i,j,k,l}$ for all four-tuples (i,j,k,l), we solve an integer program to find the priority assignments specified by the decision variables $x_{i,j,k,l}$. The integer program is a generalization of the classical *assignment problem*; see Hillier and Lieberman (2003), Winston and Venkataramanan (2002) and Wolsey (1998). We use standard mathematical programming algorithms to find an optimal solution or an efficient heuristic to find a good solution. In either case, the mathematical program gives a precise specification of the decision variables, which in turn directly produce the desired priority matrix P.

In particular, we maximize the total value subject to constraints on the number of agents of each type. That is, we solve the **optimization problem:**

$$\max_{\{x_{i,j,k,l}:\, 1\le i\le n, 1\le j\le m, 1\le k\le m, 1\le l\le m\}} \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} x_{i,j,k,l}\cdot v_{i,j,k,l} \tag{4.4}$$

subject to:

$$\begin{aligned}
x_{i,j,k,l} &\in \{0,1\} \quad \text{for all} \quad (i,j,k,l)\ , \\
x_{i,j,k,l} &= 0 \quad \text{if} \quad j,k \quad \text{and} \quad l \quad \text{are not distinct}\ , \\
\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} x_{i,j,k,l} &= n\ , \\
n_j^L \le \sum_{i=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{m} x_{i,j,k,l} &\le n_j^U \quad \text{for all} \quad j\ , \\
n_{j,k}^L \le \sum_{i=1}^{n}\sum_{l=1}^{m} x_{i,j,k,l} &\le n_{j,k}^U \quad \text{for all} \quad (j,k) \quad \text{with} \quad k\ne j\ , \\
n_{j,k,l}^L \le \sum_{i=1}^{n} x_{i,j,k,l} &\le n_{j,k,l}^U \quad \text{for all} \quad (j,k,l) \quad \text{with distinct} \quad j,k,l\ .
\end{aligned} \tag{4.5}$$

We apply available optimization techniques to solve this optimization problem. Given any optimal solution (there may be more than one) or a good (possibly suboptimal) solution, we then obtain the agent-priority matrix P be letting $P_{i,j} = 3$, $P_{i,k} = 2$ and $P_{i,l} = 1$ if $x_{i,j,k,l} = 1$. Otherwise, we let $P_{i,p} = 0$ if agent i has the skill to handle type p calls; if not, we assign no number (or -1).

The indirect VBR algorithm is the SBR algorithm developed in Section 2 using the special priority matrix P created in this section from specified values. The indirect VBR algorithm provides a way to implement value-based routing in the framework of an existing SBR algorithm. Both an advantage and a disadvantage of this indirect VBR algorithm is that it has fewer parameters than the direct VBR algorithm in Section 3. Even though this algorithm has a different structure than the direct-VBR algorithm in Section 3, we can compare the algorithms by using the same overall performance criterion (3.1), assuming that both algorithms are based on the same values $\{v_{i,j}\}$.

5. Preference-Based Staffing and Routing

For the direct-VBR and indirect-VBR algorithms developed in the previous two sections, we did not specify where the values came from. Following current practice, it is natural to assume that *management* assigns *both* the *values* and the additional constraining *parameters.* However, we instead obtain (agent-based) *preference-based-routing* (PBR) algorithms if we let the agents participate in assigning the values $v_{i,j}$, leaving management to assign the additional constraining parameters. We might even let the values be assigned entirely by the agents. By allowing these values to affect whether or not the agent works at all, we also obtain preference-based staffing.

5.1. Direct Agent Call Selection

If we have decided that we want to let the values be assigned entirely by the agents, we might also go further and let the agents directly select the calls. In some circumstances that may be feasible and desirable. Such a scheme requires that the agents receive the required information, but it evidently is possible to make information about the waiting customers available to the agents.

There are a number of concerns, however, which keep this approach from being our main proposal. First, it is problematic in the situation in which many agents are idle and a new call arrives. In that situation, it seems inappropriate to rely on the individual agents, because they would then be *competing* to answer (or not answer) each new call.

Thus, we might elect to use direct agent call-selection only when calls are waiting and an agent becomes available, either after handling a call or after being idle. However, there might be a great many calls waiting, so that the agent would have to do much information-processing. Moreover, management might be reluctant to relinquish control. However, there is an intermediate approach that can address both of those problems: The agents might be given the opportunity to directly select the call from among a *subset* of the waiting calls, designated by the system, under management guidelines.

5.2. Composite and Weighted Values

However, the main approach we propose is value-determination using input from *both* management and the agents. We could let the values be directly specified by the agents, but in many cases it will be desirable to compromise between management priorities and agent preferences, as expressed via their separate value functions. It may not be enough to let management specify the additional constraining parameters.

Thus we now define new PBR algorithms by combining *management-specified values* $v_{i,j}^m$ with associated *agent-specified values* $v_{i,j}^a$ in order to obtain *composite values* $v_{i,j}^c$. To do so, we define a function mapping the pair of functions (v^m, v^a) into the single function v^c. There are many ways to do that. We propose a simple specific method, the *product mapping*, yielding the *value product*:

$$v^c_{i,j} = v^a_{i,j} \cdot v^m_{i,j} \quad \text{for all} \quad i, j \ . \tag{5.1}$$

However, we also go beyond that and introduce a whole class of procedures for constructing composite values that satisfy certain reasonable properties. Let $I \equiv \{1, 2, \ldots, n\}$ be the set of n agents, and let $J \equiv \{1, 2, \ldots, m\}$ be the set of m call types. The value functions can be represented as

$$v^m : I \times J \to S_m \quad \text{and} \quad v^a : I \times J \to S_a \ , \tag{5.2}$$

where S_m is the allowed set of management values (preference scores) and S_a is the allowed set of agent values (preference scores).

We let our *composite value function* v^c be defined simply by

$$v^c : I \times J \to S_c \quad \text{with} \quad v^c_{i,j} = c(v^m_{i,j}, v^a_{i,j}) \ , \tag{5.3}$$

where c is a *composition function*, defined by

$$c : S_m \times S_a \to S_c \ . \tag{5.4}$$

Given the sets S_m, S_a and S_c, it is natural to simply require that the composition function be *monotone*, i.e., by having $c(x_1, y_1) < c(x_2, y_2)$ if either (i) $x_1 < x_2$ and $y_1 \le y_2$ or (ii) $y_1 < y_2$ and $x_1 \le x_2$. Two natural examples of monotone composition functions are the product above in (5.1) and the sum. The value product seems to be a nice simple choice.

In addition, we provide management with *controls* to continually adjust between these three kinds of values. Specifically, management can choose *weights* $w^m_{i,j}$, $w^a_{i,j}$ and $w^c_{i,j}$, which in generally we allow to depend on the agent i and the call type j, with the property that, for each pair (i, j),

$$w^m_{i,j} \ge 0, \quad w^a_{i,j} \ge 0, \quad w^c_{i,j} \ge 0 \quad \text{and} \quad w^m_{i,j} + w^a_{i,j} + w^c_{i,j} = 1 \ . \tag{5.5}$$

Then the final *weighted values* are

$$v^w_{i,j} = w^m_{i,j} \cdot v^m_{i,j} + w^a_{i,j} \cdot v^a_{i,j} + w^c_{i,j} \cdot v^c_{i,j} \ . \tag{5.6}$$

Note that the first three kinds of values - the management-specified values $v^m_{i,j}$, the agent-specified values $v^a_{i,j}$ the composite values $v^c_{i,j}$ - are each obtained as a special case; e.g., we get $v^m_{i,j}$ if we simply choose the weights $w^m_{i,j} = 1$. The generality of the weights (the dependence upon i and j) allows management to choose weights depending on the agent, the call type and the combination. For example, management might place more weight on agent values over time as the agent demonstrates good performance. In that way, the agent can acquire *earned empowerment*; see [3] for more discussion.

In summary, we let the individual agents express their preferences through their value function; i.e., agent i declares his call-handling values $\{v^a : 1 \le j \le m\}$. Management in turn has several controls: First, management specifies the remaining constraining parameters, as specified in Sections 3 and 4. Second, management declares its priorities through its own value function $v^m \equiv \{v^m_{i,j} : 1 \le i \le n, 1 \le j \le m\}$. Third, management specifies the composition function c in (5.4), which is required to be monotone. Finally, management specifies the weights $w^m_{i,j}$, $w^a_{i,j}$ and $w^c_{i,j}$ in (5.5) and (5.6). In that way, management contributes to the specification of the final weighted values $v^w_{i,j}$ in (5.6). Those weighted values can then applied with the algorithms in Sections 3 and 4 to obtain the corresponding PBR algorithms.

5.3. A Specific Simple PBR Algorithm

In this section we introduce a specific PBR algorithm, which we believe is appealing for its simplicity. We also believe it can be effective.

As before, we start with n agents responding to m types of calls. In this specific algorithm, we assume that management determines its values $v^m_{i,j}$ for all nm pairs (i,j), assuming that they each take one of the four values in the set $\{0,1,2,3\}$. Management then communicates these values along with appropriate incentives to the agents. In response, agent i for each i individually determines his values $v^a_{i,j}$ for each call type j, again assuming that they take one of the four values in the same set $\{0,1,2,3\}$. By design, we use a symmetric structure. In addition, we let the agent indicate that he prefers not to work at all; in addition to specifying the value $v_{i,j} = 0$, the agent sets a flag indicating that he prefers not to work at all during this period.

We then use the product function in (5.1) to form composite values $v^c_{i,j}$, which necessarily take one of the seven values in the set $\{0,1,2,3,4,6,9\}$. For the weighted value function in (5.6), we assume that $w_c = 1$, so that $v^w = v^c$; i.e., we work directly with the composite value function.

We now specify how to do the routing. We could use the direct PBR algorithm, which is just the direct VBR algorithm based on the composite values $v^c_{i,j}$. Instead, we propose using the indirect PBR algorithm to do the routing. In particular, we propose generating the priority matrix P in the manner specified in Section 4, allowing only three skills for each agent, and allowing each priority level to appear only once in each row of the priority matrix. As described in Section 4, we use the integer program to create the priority matrix P, using the composite value function v^c created above as input, where (4.2) provides the values $v_{i,j,k,l}$ from the values $v^c_{i,j}$. For simplicity, we simply set $\epsilon_{i,j,k} = 0.20$ for all (i, j, k) and $\epsilon_{i,j,k,l} = 0.05$ for all (i, j, k, l), as suggested in Section 2. To do the routing itself, we then use the indirect VBR algorithm based on the priority matrix P, which we have just specified.

To achieve transparency and mutual trust, management could communicate the full priority matrix to all the agents. For even greater transparency, management could communicate all the value functions. Such full disclosure might lead to further negotiations about the priority assignments. On the side, pairs of agents might agree to switch assignments.

Finally we discuss *performance evaluation.* We propose doing the performance evaluation and the routing *differently*: We suggest evaluating the performance by the composite values $v^c_{i,j}$, but doing the routing using the priority matrix P (which of course is based on the values $v^c_{i,j}$). For example, by this method, agent i has only a single highest-priority (priority-3) skill in the routing, and yet the composite values $v^c_{i,j}$ for agent i's skills could all be 9's.

We can use the composite value function v^c in the performance target (3.1) to evaluate the overall performance of the routing algorithm. That evaluation applies to any routing scheme we might use. However, a different goal seems very appealing: Instead (or in addition), we should pay attention to the proportions of calls receiving different composite-value scores. In favorable circumstances, we might want to have at least 80% of all calls with composite value 9, and at least 95% of calls with composite value 6 or higher. In doing so, we are assuming that greater customer satisfaction and agent satisfaction will be associated with higher composite values.

In other words, we suggest looking separately at the components of the overall performance criterion in (3.1). We want to achieve high composite value for the handled calls, but we also want low abandonment and low delay for the answered calls. It is natural to pay attention to each performance statistic as well as to combine them into one overall performance score.

6. A Dynamic Algorithm with Feedback

We envision the VBR and PBR algorithms being dynamic algorithms with feedback: The values and the constraining parameters in the algorithms should be dynamically adjusted in response to performance and input from the other parties. For example, adjustments in the algorithm might be done every half hour. However, it might be preferable to allow agents to express preference changes at any time. The system might be designed to immediately respond favorably to agent requests, providing current contact-center performance is good enough.

We envision the PBR algorithm working as follows: First, management determines its value function v^m and communicates it to the agents. Management may communicate its entire value function to each agent or it may only communicate $\{v^m_{i,j} : j \in J\}$ to agent i for each i. Then, in response, the agents express their preferences through their value functions: Agent i communicates its preferences $\{v^a_{i,j} : j \in J\}$ to management. Then, given all the management and agent values, the algorithm determines the composite value function v^c and the weighted value function v^w, using the management-specified composition function c and weights w^m, w^a and w^c. Then the final weighted value function $v^w \equiv \{v^m_{i,j} : 1 \leq i \leq n; 1 \leq j \leq m\}$ is used to generate the routing, following one of the algorithms in Sections 3 or 4, based on the constraining parameters set by management.

But that is not the end of the story: The algorithm dynamically adjusts through time. The algorithm based on the value v^w routes the calls. Then the results of this routing is observed. The algorithm, as directed by management (either automatically or manually) can update its value function v^m in response to network conditions (as communicated by input from the ACD) and the way that business objectives are being met (as communicated by input from the CRM system). Moreover, management can update other parameters such as the weights w^m, w^a and w^c used in forming the weighted values. As indicated in Section 5.2, agents may be experiencing earned empowerment in response to their performance.

Moreover, management can offer the agents *incentives*, external to the routing system, encouraging the agents to make desired choices. The overall procedure repeats, just as above, but with the understanding that at each decision point, the agents are responding both to updated management value function and incentives communicated to them by management. The agents may also be given information about network conditions and the way business objectives are being met, so that the agents too can see the overall contact-center needs.

The incentives described above may take various forms. They might involve direct compensation. Alternatively, they might be in the form of loyalty points, which are later redeemable for rewards or prizes. For example, incentives might be given for handling difficult customer-service calls involving complaints.

7. Conclusions

We have proposed a new strategy to empower contact-center agents via preference-based routing. To show that the proposed strategy is feasible, we have developed algorithms to perform value-based routing (VBR) and preference-based routing (PBR), with extensions to cover staffing as well. The PBR algorithms have been developed in the framework of the direct and indirect VBR algorithms in Sections 3 and 4. The indirect VBR algorithm in turn has been developed in the framework of the priority-based SBR algorithm in Section 2. Thus both a VBR algorithm and a PBR algorithm can be implemented using the priority-based SBR algorithm, using an existing algorithm on the ACD. It is not necessary to modify the algorithm on the ACD, provided that it is the priority-based SBR algorithm in Section 2. We anticipate that VBR and PBR algorithms can be developed within the framework of other existing SBR algorithms, in much the same way.

It remains to determine good parameter settings for these routing algorithms and evaluate their effectiveness. It also remains to more fully specify the dynamic-feedback aspects of the routing algorithms discussed in Section 6. Natural next steps are simulation experiments. For them, we should keep in mind that we not only want to ensure that the traditional goals of load-based routing and skill-based routing are still met, but that we want to realize more total value, as expressed by a target such as in Section 3.1. However, our ultimate goal is to go beyond that to achieve the desired behavioral changes among the agents and the customers: We want more satisfied agents and customers as a result, and thus less agent turnover and absenteeism, and more customer demand.

Preference-based staffing and routing was conceived as a way to empower contact-center agents, enabling them to participate more fully in contact-center decisions, and thereby experience more job satisfaction. That is intended to supplement other actions such as providing better compensation and better career paths. The ultimate proof of the pudding is in the tasting: Can it work in practice?

8. Acknowledgments

We thank Mohammed Asif for his assistance and Peter Bamberger and Michal Borin of the Technion for directing us to research literature related to job satisfaction, retention and performance in contact-centers.

References

[1] Anton, J., V. Bapat, B. Hall. 1999. *Call Center Performance Enhancement Using Simulation and Modelling.* Purdue University Press.

[2] Armony M, Gurvich I., A. Mandelbaum. 2004. Staffing and control of large-scale service systems with multiple customer classes and fully flexible servers. The Technion, Haifa, Israel.

[3] Bamberger, P. A. and Biron, M. 2005. Earned empowerment: motivating employees on the basis of incremental increases in job-based control. Technion Working Paper, Technion, Haifa, Israel.

[4] Batt, R. 2000. Strategic segmentation in front line service: matching customers, employees and human resource systems. *International Journal of Human Resource Management* 11, 540–561.

[5] Batt, R. 2002. Managing customer services: human resource practices, quit rates and sales growth. *Academy of Management Journal* 45, 587–599.

[6] Bliss, W. G. 2004. Cost of employee turnover. *The Advisor.*
Available at: http://www.isquare.com/turnover.cfm

[7] Bowen, D. E., E. E. Lawler. 1995. Empowering service employees. *Sloan Management Review* 36, 73–84.

[8] Brigandi, A.,D. Dargon, M. Sheehan, T. Spencer, III. 1994. AT&T's call processing simulator (CAPS): operational design for inbound call centers. *Interfaces* 24, 6–28.

[9] Chevalier, P., R. A. Shumsky, N. Tabordon. 2004. Routing and staffing in large call centers with specialized and fully flexible servers. Universite catholique de Louvain, University of Rochester and Belgacom Mobile/Proximus.

[10] Cleveland, B., S. Hash. 2004. (editors) *Call Center Agent Motivation and Compensation*, Call Center Press, ICMI, Annapolis, MD.

[11] Cleveland, B., J. Mayben. 1997. *Call Center Management on Fast Forward*, Call Center Press, ICMI, Annapolis, MD.

[12] Collins, D. 1996. Control and isolation in the management of empowerment. *Empowerment in Organizations* 4, 29–39.

[13] Corsun, D. L., C. A. Enz. 1999. Predicting psychological empowerment among service workers: the effect of support-based relationships. *Human Relations* 52, 205–224.

[14] Deery, S., N. Kinnie. 2004. *Call Centres and Human Resource Management*, Palgrave, Macmillan.

[15] Deming, W. E. 2000. *Out of the Crisis*, M.I.T. Press, paperback.

[16] Gans, N., Koole, G., A. Mandelbaum. 2003. Telephone call centers: tutorial, review and research prospects. *Manufacturing and Service Operations Management* (M&SOM) 5, 79–141.

[17] Glebbeek, A. C., E. H. Bax. 2004. Is high employee turnover really harmful? An empirical test using company records. *Academy of Management Journal* 47, 277–286.

[18] Hackman, J. R., G. R. Oldham. 1976. Motivation through the design of work. *Organizational Behavior and Human Decision Processes* 16, 250–279.

[19] Hillier, F. S., G. J. Lieberman. 2003. *Introduction to Operations Research*, seventh edition, McGraw-Hill.

[20] Holman, D. 2002. Employee wellbeing in call centres. *Human Resource Management Journal* 12, 35–50.

[21] Holman, D. 2003. Call centres. Chapter 7 in *The New Workplace: A Guide to the Human Impact of Modern Work Practices*, D. J. Holman, T. D. Wall, C. W. Clegg, P. Sparrow and A. Howard (editors), Wiley.

[22] Holtgrewe, U., C. Kerst, K. Shire. 2002. *Re-organizing Service Work: Call Centres in Germany and Great Britain*, Aldershot.

[23] Juran, J. M., A. B. Godfrey. 1999. *Juran's Quality Handbook*, fifth edition, McGraw-Hill.

[24] Karasek, R. A., T. Theorell. 1990. *Healthy Work: Stress, Productivity and the Reconstruction of Working Life*, Basic Books.

[25] Leamon, P. 2004. Workforce management for skill-based routing: the need for integrated simulation. IEX White Paper, IEX Corporation, Richardson, Texas.

[26] Mitchell, T. R., Holtom, B. C., Lee, T. W., Sablynski, C. J., M. Erez. 2001. Why people stay: using job embeddedness to predict voluntary turnover. *Academy of Management review* 44, 1102–1121.

[27] Quiñones, M. A., Ford, J. K., M. S. Teachout. 1995. The relationship between work experience and job performance: a conceptual and meta-analytic review. *Personnel Psychology* 48, 887–910.

[28] Rafig, M., A. K. Pervaiz. 1998. A customer-oriented framework for empowering service employees. *The Journal of Serices Marketing* 12, 379–387.

[29] Ruyter, K., Wetzels, M., R. Feinberg. 2001. Role stress in call centers: its effects on employee performance and satisfaction. *Journal of Interactive Marketing* 15, 23–35.

[30] Singh, J., Goolsby, J. R., G. K. Rhoads. 1994. Behavioral and psychological consequences of boundary spanning burnout of customer service representatives. *Journal of Marketing Research* 31, 558–569.

[31] Spraetz, B. 2004. Empowerment: a recipe for success; one part trust, one part technology. *InView*, IEX Corporation, November-December 2004.

[32] Wallace, R. B., W. Whitt. 2004. A staffing algorithm for call centers with skill-based routing. submitted for publication. Available at http://columbia.edu/∼ww2040.

[33] Whitt, W. 1999. Dynamic Staffing in a Telephone Call Center Aiming to Immediately Answer All Calls. *Operations Research Letters* 24, 205–212.

[34] Whitt, W. 2004. The impact of increased employee retention upon performance in a customer contact center. Available at http://columbia.edu/∼ww2040.

[35] Winston, W., M. Venkataramanan. 2002. *Introduction to Mathematical Programming: Applications and Algorithms*, third edition, Duxbury.

[36] Witt, L. A., Andrews, M. C., D. S. Carlson. 2004. When conscientiousness isn't enough: emotional exhaustion and performance among call center customer service representatives. *Journal of Management* 30, 149–160.

[37] Wolsey, L. A. 1998. *Integer Programming*, Wiley.