UNIVERSITY OF CALIFORNIA
SANTA CRUZ

ROUTING TO DEVELOP EXPERTISE IN CUSTOMER CONTACT CENTERS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Geoffrey S. Ryder

June 2009

The Dissertation of Geoffrey S. Ryder
is approved:

_____
Professor Alexandre Brandwajn, Chair

_____
Professor Kevin Ross

_____
Professor Vijay Mehrotra

_____
Professor John Musacchio

_____
Lisa C. Sloan
Vice Provost and Dean of Graduate Studies

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Abstract

Routing to Develop Expertise in Customer Contact Centers

by

Geoffrey S. Ryder

A call center often serves as the principal or only point of contact between a company and its customers, and the cost of staffing the center with suitably skilled agents is a large proportion of its operating costs. In this environment, the efficiency and quality with which agents handle customer encounters determines the economic productivity of the call center. This research describes the performance of routing policies for inbound call center traffic that trade off two conflicting objectives: minimizing the waiting time for customers, and specialized on-the-job skill acquisition by agents that improves the customer experience.

Empirical results for agent productivity trends are derived from individual summary data for 2.7 million calls taken over the course of 2007 by a financial service call center. This unique data set describes both service time and service quality for 178 separate types of inquiries, regarding diverse subjects such as credit card billing, tax advice, and sales of investment products. Service quality is measured independently of an agents own assessment by recording the resolution status of each call: does the record show this customer finds it necessary to call again about the same issue, or has this problem been resolved successfully?

Observations from this data show that the notion of expertise may be characterized by convex functions of cumulative production. Analytical results are then obtained for utility functions of the agents expertise: the customers utility, encouraging specialized expertise; and the supervisors utility, encouraging an even distribution of expertise among agents.

These utility functions guide the development of nonlinear programs that set expertise targets, in the form of work assignments, for call center staff who are observed to have varying abilities.

Routing rules to implement these work assignments are then analyzed, using simulations of call center operations in the presence of stochastically varying customer arrival and departure times. The research describes how well-targeted priority rules achieve higher levels of service experienced by customers, with minimal effects on the time spent waiting to be served.

I want to thank my committee, with whom it was a pleasure and privilege to work, and especially my advisor Kevin Ross, who gave generously of his time and expertise over the last three years. Kevin, Vijay, John, Alexandre, and visiting scholar Christoph Heitz deepened my understanding of computer engineering, operations research, and the emerging field of service science. My sincere thanks go to my family and friends for their many kindnesses; thank you Mac and Susan for always pushing me to aim higher. Most of all, I thank my wife Miyuki for her devoted, unyielding encouragement and support.

Acknowledgments

The text of this dissertation includes a reprinting of the following previously published material in Chapter 2:

Geoffrey S. Ryder, Kevin G. Ross, and John T. Musacchio. "Optimal service policies under learning effects." *Int. J. Services and Operations Management,* 4(6):631651, 2008.

Co-author Kevin Ross directed and supervised the research which forms the basis for this dissertation, and co-author John Musacchio served on the dissertation reading committee.

CHAPTER

ONE

Managing On-The-Job Expertise Development in Contact Centers

1.1. Expertise in a Call Center Setting

A major influence on a customer's satisfaction at a call center is the knowledge level of the agent who takes their call. Knowledge management, in particular maintaining or increasing the cumulative knowledge of the agents, is therefore a key issue for ensuring service quality. This is especially true when the call center operates within dynamic markets, and agents are required to keep pace with changes.

For the operational management, on the other hand, the knowledge of the employees is

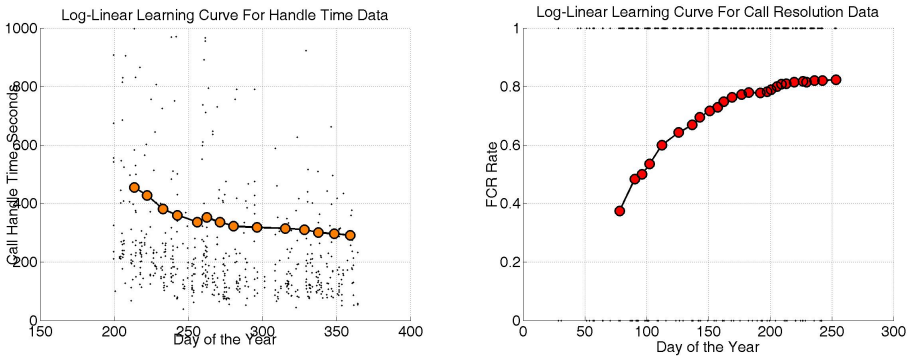


Figure 1.1: **Empirical Examples of Improving Performance.** Examples of cumulative average performance improving with experience, from the empirical study of Chapter 4. Left: an agent gets faster and decreases her average handle time from 8 minutes to 5 minutes, a 40% improvement, after handling about 560 calls over five months. Right: a different agent improves his cumulative average first call resolution rate from 40% to 80% after taking about 560 calls over six months.

usually treated as *exogenous* to the service delivery process. Knowledge is considered to be a given and fixed resource, and is treated as such for routing and call assignments. This makes sense when all training happens off-line, but does not account for the case where knowledge and expertise are actually gained on-the-job through the service process itself; and empirical data shows that agents learn by doing—see Figure 1.1.

If it is assumed that learning-on-the-job takes place, then the operational rules have an impact on knowledge, and knowledge is therefore an *endogenous* rather than an exogenous variable. In particular, routing policies determine which agents work on which jobs, and thus may have a major impact on the learning of the agents and their expertise level attained. This thesis examines how routing might influence the knowledge, how changing knowledge levels will affect customer experience, and how knowledge management and routing can be treated together.

Chapter 4 describes a new set of empirical performance data from a financial service contact center. The conclusions drawn from the data drive the theoretical approaches taken in other chapters. A key observation is the fact that agents are not busy all the time—only part of an agent's work day is spent in direct contact with customers. Figure 1.2 shows the pattern for a typical agent.

There are several reasons for this, including the helpful effects of automated customer sup-

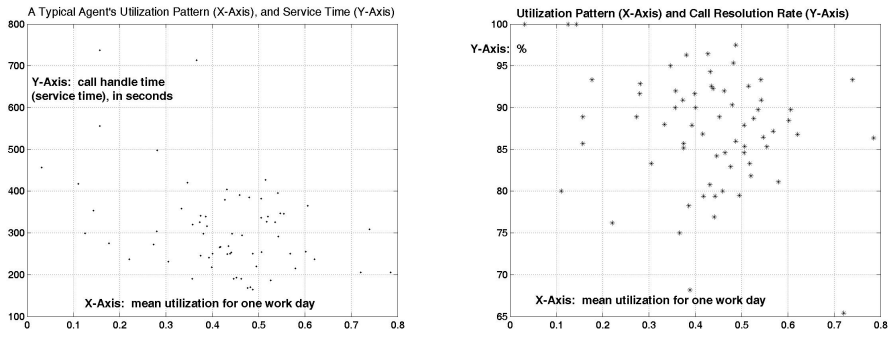


Figure 1.2: **Agent Utilization.** Plot of one agent's utilization rate over a two-month interval. Points with different horizontal coordinates represent different days, with different mean utilization values (different %-busy values). Each day is represented by two points, a dot(.) for mean call handle time on the left, and a star (*) for mean call resolution rate on the right. On a typical day, this agent spends less than 50% of his total work time directly engaged in customer encounters.

port systems, and management's desire to maintain a staffing level sufficient to provide responsive service even during unpredictable peaks in demand. In this system, management has taken the decision to staff for the *quality regime* described in Gans et al. [2003], page 100: "Waiting costs of customers dominate the cost of capacity, and the optimal staffing policy uses an asymptotically fixed utilization rate." Chapter 5 duplicates this policy in its simulations of call center agents, where a fixed utilization rate drives the choices for the other parameters. Note that agents remain productive, even when not directly interfacing with customers: they research customer inquiries, file reports, update automated help systems, attend staff meetings or training sessions, and so on. Time "not busy" is not wasted time.

For my research, these utilization levels point to the study of stochastic models where service time and customer waiting time are still important, and must be considered—but in which there is also some flexibility to route calls in ways that optimize other objectives besides waiting time. Here, the additional objective is the distribution of knowledge among agents. As shown in Pinker and Shumsky [2000], large systems tend to have more flexibility in routing different kinds of work to their agents. Some of the most interesting trade-offs between expertise development and responsive service occur in small to medium sized call centers, and so they will be my primary emphasis.

1.2. Thesis Outline

1.2.1 Chapter Organization

Chapters 2 through 5 each investigate a different aspect of expertise development through on-the-job learning among contact center agents. Taken together, these chapters provide a set of tools with which contact centers can incorporate expertise-driven productivity changes into capacity and quality management plans—a subject of fundamental interest in the field of knowledge management for service enterprises. The chapters are organized along the row and column dimensions of Table 1.1. Agent performance metrics define the rows, and the way in which experience drives expertise defines the columns in this scheme.

Contact center management aims to provide high quality service in an efficient way. But what do these concepts mean in a concrete sense? To answer this, the first dimension defines metrics used to evaluate performance. Froehle [2006] reported recent survey results for contact center customers, and characterized their expectations of the agents who helped them. The principal qualities requested were preparedness, avoiding wasted time; subject matter knowledge; and thoroughness. In Table 1.1, the total time taken by an agent to handle a customer encounter, or H, measures preparedness or efficiency. H captures the temporal effect of improved expertise on service quality—the greater the expertise, the less time needed to provide service.

The metrics in the next row express knowledge and thoroughness that accrue with experience, independent of improvements in handle time. The quantity X is used in Chapter 3 to represent an agent's increased store of knowledge following a service encounter. The first call resolution rate R measures the proportion of customers for whom an agent completes the customer's business at the time of first inquiry.

The three metrics H, X, and R are similar in that they increase in experience, and their rates of growth diminish as experience accrues. They are modeled here by bounded concave or convex functions in recent or in cumulative production. But these metrics also differ in ways that call for

Performance Metric	Expertise from Recent Experience	Expertise from Cumulative Experience
Call Handle Time H	**Chapter 2** Dynamic programming analysis of trends in call handle time (H) for a small queueing system in the presence of learning and forgetting effects.	**Chapter 4** Empirical observations of faster call handle times (H) with increasing experience. **Chapter 5** Designs and simulates optimal work assignments under learning. Applies customer's and supervisor's utility functions created in Ch. 3 to promote improvements in call handle times (H) over groups of agents.
General Expertise X, First Call Resolution Rate R	**Chapter 3** Defines utility functions in agent expertise (X). These functions are useful as metrics for evaluating the distribution of expertise among a group of agents.	**Chapter 4** Empirical observations of improving changes in first call resolution rates (R) with increasing experience.

Table 1.1: **Thesis Chapter Organization.** Chapters 2, 3, 4, and 5 together develop a toolkit of methods for making on-the-job expertise development an endogenous factor in the process of managing a call center.

special treatment.

- Learning-based improvements in the mean handle time H require analytically intractable modifications to standard queueing system models (Chapter 2), forcing such improvements to be modeled in randomized simulations (Chapter 5).

- Learning-based improvement in the mean first call resolution rate R enhances the customer experience and the firm's reputation, and in this thesis a higher rate R is desirable. Chapter 4 studies empirical trends in metrics that include R. A potential benefit of improving R that is *not* modeled here is any reduction in future traffic to the call center–when an agent often satisfies a customer on the first inquiry, fewer repeat visits may be needed.

- Chapter 3 presents a model of improving expertise that is compatible with the Markovian assumptions for M/M/1 queues. As such, it assumes the mean handle time H is constant. It also assumes the mean arrival rate is constant, and so ignores traffic fluctuations due to changes in the resolution rate R. A new quantity X is therefore introduced to represent the increase in knowledge with experience, and X is independent of metrics H and R.

Models of how expertise drives experience define the columns of Table 1.1. Both learning and forgetting have been measured in industrial settings, where learning curves contribute to improved productivity, but production breaks are then seen to relinquish some of those gains. Chapters 2 and 3 incorporate both learning and forgetting trends in new experimental models of agent expertise levels. Chapters 4 and 5 adopt a simpler model that uses only learning with cumulative production, without forgetting due to absences or breaks. The simpler model was found to be reasonable for both modeling agent learning trends from new empirical data, and for building optimal work assignments for groups of agents. Expanding on the ouline of Table 1.1, the motivation and modeling choices for each chapter follow.

Chapter 1: Introduction—an overview of research on call centers and agent productivity, placing these topics in the context of recent related results.

Chapter 2: ***Motivation.*** How do learning and forgetting mechanisms that drive expertise development affect the results of stochastic models of service operations? A Markov decision process model can explore this problem by obtaining steady-state results from dynamic program, showing that if learning and forgetting affect the mean call handle time, then the optimal policy for routing work to an agent will change. ***Model.*** The dynamic programming solver computes the optimal policy to follow for each state in a discrete Markov chain, and its performance is compared to the performance of four simple heuristic policies that might be used if the optimal policy were unknown. To evaluate performance, a transition matrix $p = [p_{ij}]$ is constructed in such a way that it is an irreducible, aperiodic Markov chain with all states positive recurrent, and so it has a stationary distribution $\vec{\pi}$ defining the steady-state probability of being in each state. The dynamic programming solver finds this distribution. Based on $\vec{\pi}$, the mean learning state value $\ell_{\vec{\pi}}$, mean queue backlog $(q_{x\vec{\pi}} + q_{y\vec{\pi}})$, and total cost for all states in the system $C_{\vec{\pi}}$ are computed for each policy. $\sum_{i=1}^{N} \pi(i) = 1$, so $\vec{\pi}$ serves as a means of weighting the values of $\ell(i)$, $q(i)$, and $C(i)$ according to the steady state likelihood of the system being in state i. (See page 25.)

Chapter 3: ***Motivation.*** How may the idea of expertise in service quality be captured for individuals, and for groups of agents? Here expertise is quantified as a continuous, monotonically increasing, concave function of the departure rate of customers served by an agent. Steady state agent expertise levels may be predicted for stochastic Markovian service systems where the mean service time is constant. Utility functions in agent expertise levels are useful as optimization objective functions—they allow the inclusion of on-the-job learning trends into the design of contact center routing rules. ***Model.*** Consider an agent who serves an M/M/1 queue of customers on a first-come, first-serve basis. A unitless value of expertse X, with $0 \leq X \leq 1$, increases with cumulative production n and decreases proportionally to time absent. These dynamics are modeled by the concave

function $X(t_{n-1}+\tau_n) = (X(t_{n-1}) + \alpha(1 - X(t_{n-1}))e^{-\beta\tau_n}$, where τ_n is the difference between two departure times from a server; α is a learning rate, and β is a forgetting rate. A proportion p, $0 \leq p \leq 1$ of the call center's customer arrival rate λ is routed to the agent. The steady-state value of the agent's expertise then becomes $X_\infty(p\lambda) = \frac{p\lambda\alpha}{\beta+p\lambda\alpha}$; by increasing or decreasing p, management increases or decreases $X_\infty(p\lambda)$. Utility functions in steady-state expertise of multiple agents are defined: U_c, the customer's utility, and U_s, the supervisor's utility; and in preparation for applications in Chapter 5 their convexity properties are explored. (See page 48.)

Chapter 4: ***Motivation.*** Do empirical observations show that agent performance improves with experience? Yes—statistical analysis of call-by-call data from a financial service contact center yields significant productivity trends, and suggest that expertise does improve with experience. Many individual agents demonstrate on-the-job learning, and specific task types demonstrate significant performance improvement with cumulative production over large groups of agents. ***Model.*** A data set of call-by-call records of one contact center for all of 2007 containing about 2.7 million calls, 1000 agents, and 178 call types yields trends in handle time performance(H), first call resolution performance (R), and performance on the combined metric μ-R = R/H. A set of standard clustering and regression tests show that cumulative production is more significant than tenure for predicting improvement, for this data; and 40% of all customer calls belong to a call type for which performance is better for high-volume agents than for low-volume agents, significant at the 95% level. (See page 75.)

Chapter 5: ***Motivation.*** How may expertise-building strategies be developed that optimize the distribution of skills among a workforce of agents? Approaches developed in prior chapters are brought together to modify work assignments to contact center agents in order to maximize operational quality and efficiency. This chapter describes a nonlinear program to

identify optimal work assignments, and a queueing system simulator that identifies the best routing rules to realize the optimal targets in contact centers. ***Model.*** A constrained nonlinear optimization program is presented to set routing targets of calls to agents in a system with I agents and K call types; and with a cumulative production log-linear learning model with learning exponent b_{ik} and starting handle time H_{ik} for every agent-type pairing ik. Both the customer's utility U_c and the supervisor's utility U_s are used as objectives to generate routing targets in separate trials, and their results compared. In discrete-event simulations, these targets are implemented with lower to higher degrees of fidelity by five types of routing rules: no-priority (NP), priority-no-idle-constant (PNI-C), priority-no-idle-swap (PNI-Swap), priority-with-idle-constant (PNI-C), and priority-with-idle-swap (PWI-Swap). The best rule to use changes depending on system conditions; and a trade-off exists between maximizing the expected value of expertise seen by a customer, and the time customers must spend waiting in queues for service. (See page 105.)

Chapter 6: The last chapter summarizes key results, and identifies avenues for future work in this area.

1.2.2 Contributions and Publications

Selected topics from this thesis have appeared before in the various journal papers, competitive and invited conference talks, and conference proceedings shown in Table 1.2.

1.3. Literature Review: Research on Agent Productivity in Call Centers

A key paper that helps establish this area of research is by Pinker and Shumsky [2000]. They analyze a Markov chain system model of learning and turnover that includes two types of specialist

Date	Venue	Subject
October, 2005	IASTED Computer and Communication Networks Conference	Clustering, Ch. 4; Optimization, Ch. 5
October, 2006	Applied Probability Section INFORMS Annual Meeting, Cincinnati	Dynamic Program/MDP, Chapter 2
October, 2007	Frontiers in Service Conference San Francisco, CA	MDP, Ch. 2; and Simulation, Ch. 5
November, 2007	Service Science Session SA25 INFORMS Annual Conference, Seattle	MDP, Ch. 2; Simulation, Ch. 5
November, 2007	Service Science Session WA27 INFORMS Annual Conference, Seattle	Empirical Data, Ch. 4; Simulation, Ch. 5
June, 2008	Ryder, Ross, and Musacchio. Article in the Intl. Journal of Operational Research (IJOR)	MDP, Ch. 2
June 2008	MSOM Conference/Annual Meeting University of Maryland	Asymptotic Expertise, Ch. 3
October, 2008	Service Science Session WB15 INFORMS Annual Conference, Washington D.C.	Asymp. Expt., Ch. 3; Empirical Data, Ch. 4
In preparation	Journal submission	Asymp. Expt, Ch. 3; Empirical Data, Ch. 4; Optim. and Simulation, Ch. 5
In preparation	Article on clustering of performance trends among agents	Empirical Data, Ch. 4

Table 1.2: **List of Publications.** Conference proceedings and a journal publication where selections of material related to this thesis have appeared.

workers, and a set of cross-trained or flexible workers that can perform either task. Each workers service quality improves with tenure, though specialists always provide the highest service quality. The optimal staffing arrangement, giving both high agent utilization and high average expertise, turns out to include a mix of specialists and flexible agents. They also specify how the staffing solution changed along the dimensions of arrival load and expertise development rate (or learning rate). High learning rates favor more specialists in large systems, and a precise, optimal mix of flexible and specialized agents in small systems. By contrast, low learning rates favor a staff mix with more flexible agents—the content of the work is simple enough that it can be mastered quickly, so agents can take on more tasks. The authors recommend the use of forgetting models in future work, and we build on their work here by exploring potential forgetting effects.

Gans and Zhou [2002] model learning and turnover effects using a Markov decision process, and demonstrate that the optimal hiring policy for each state of a firms agent roster is a "hire-up-to" policy similar to the "order-up-to" policies from the supply chain management literature. Whitt [2006] explores ways to characterize the employee retention distribution for call center agent populations. Improved retention results in a higher average expertise of agents in the center, because expertise improves with tenure. Among other findings, decreasing distributions such as the negative exponential are noted to be reasonable first approximations to real retention distributions. This is because in general employees are most likely to leave within a short time of starting, and the probability of leaving then decreases as tenure grows.

We share a key assumption with these three previous papers: we assume service quality, denoted here by variable X, improves with tenure; or more precisely with cumulative production. But we do not specify precisely what improved quality means to the customer. This allows us to reason about the relationship of routing rules and expertise in a general way that can be adapted to fit specific cases. In an application of our results to call center agent data, X may stand for the handle time of a call, which should decrease with expertise; or to the first call resolution rate (FCR), which should increase with experience. See de Vericourt and Zhou [2005] for a discussion of the FCR metric.

If SERVQUAL-type survey data is available, X may represent a measure of customer satisfaction (Parasuraman et al. [1988]). Froehle [2006] conducts a statistical analysis of such data, and finds that agent preparedness, subject matter knowledge, and thoroughness are most important to customers' perceptions of service—three qualities that can be expected to increase with experience. Froehle also describes the alternatives modern agents have for communicating with customers, such as email and instant messaging. The customer call center is now rightly termed a *customer contact center* as well. We will use the terms interchangeably.

Our work has been guided by results in the literature from several areas of service operations research. For more on learning and turnover in call centers, see Bordoloi [2004]. (Zohar et al. [2002] show that learning by customers impacts operations as well.) For an in-depth background on call center planning and operations, see Aksin et al. [2007], Brown et al. [2002], Cleveland and Mayben [2000], Gans and Zhou [2003], Gans et al. [2003], Hasija et al. [2005], Iravani et al. [2007], and Koole [1997]. Due to the inherent complexity of call center operational models, high quality simulations are becoming important (Mehrotra and Fama [2003], Avramidis and L'Ecuyer [2005]).

For related results on learning and forgetting at work, Shafer et al. [2001] present a detailed study of empirical learning and forgetting data in an industrial application with worker service times roughly equivalent to call handle times in a call center. Badiru [1992] presents a survey of applied learning models, and Nembhard and Osothsilp [2001] do the same for forgetting models. Sikstrom and Jaber [2002] explore new ways of measuring the impact of production breaks on productivity. Sayin and Karabati [2007] develop a detailed optimization model for solving a rostering problem with learning and forgetting effects in a corporate setting involving several departments. A similar problem is explored by Eitzen et al. [2004], who note that forgetting effects require that worker skill levels be maintained through repetition in work assignments.

We do not discuss the details behind learning and forgetting effects here, but there is a body of work from the behavioral sciences that supports our operational models. See especially Globerson and Levin (1987), Howick and Eden (2007), and Schilling et al. (2003). Behavioral scientists see new

opportunities opening up now for joint work with those in the operations field, in order to apply the growing catalog of behavioral results to service operations (Bodreau et al. 2003).

There is a growing set of work describing call center outsourcing contracts, and the competitive milieu faced by call center operators—see for example Aksin et al. [2008], Hasija et al. [2008], Ren and Zhou [2008], Shumsky and Pinker [2003]. Reflecting this reality, the ultimate goal of this research is to provide new avenues for productivity growth in call center operations, so that savvy operators may drive more profitable and lower cost service contracts, such as described in Reis [1991]. Although in practice measuring and acting on learning curves requires care, productivity gains have contributed to business success in a variety of settings (Ghemawat [1985]).

CHAPTER

TWO

Expertise Predicted by Dynamic Programming

In this chapter, we consider the impact of changing service rates in a small queueing system. The mean service rate is usually considered fixed in these kinds of stochastic models, so loosening this restriction changes the nature of the optimal control policy. Here we model changes in the mean as driven by learning and forgetting mechanisms, and apply dynamic programming to evaluate five policies for directing service in the system. The queueing performance—keeping queue lengths short, so as to provide prompt service to all customers—is computed for the optimal policy, and then that performance is compared to the performance of four simple heuristic policies that might be used if the optimal policy were unknown.

The table of chapter topics below shows how this study fits among the problems explored in other chapters. While this dynamic programming approach provides valuable insight and exact results for small systems, it is difficult to scale up, and so is probably not viable as a means to examine larger systems. Chapter 5 offers an alternative in the form of a two-step, optimization/simulation approach that is feasible for both small- and large-scale problems. For more context see also the full thesis outline, the motivations behind each chapter, and the separate chapter models that are described starting on page 5 in Chapter 1.

[1]

Performance Metric	Expertise from Recent Experience	Expertise from Cumulative Experience
Call Handle Time H	**Chapter 2** Dynamic programming analysis of trends in call handle time (H) for a small queueing system in the presence of learning and forgetting effects.	**Chapter 4, Chapter 5**
General Expertise X, First Call Resolution Rate R	**Chapter 3** (X)	**Chapter 4** (R)

2.1. Scenario: Learning Effects and the Optimal Work Assignment

2.1.1 The Effect of Work Allocation on Skill Development

Consider the situation where a customer service manager must assign tasks to one of her subordinates. The assignment must take into account the customers' needs and the agent's skill at

[1] Our analysis here has also been published in the Inderscience journal *The International Journal of Services and Operations Management* (Ryder et al. [2008])—see **http://www.inderscience.com** and select this journal title, Volume 4, Issue 6, and 2008.

handling them. In response to customer demand, she assigns the agent to process two classes of incoming jobs, as in Figure 2.1. Customers who cannot immediately enter service join the queue with other customers who share the same type of problem to be solved.

The managers of a service facility might often use a queueing model to assign jobs to agents, and these models typically assume that an agent's mean service time is static. In reality, the agent's skill fluctuates due to his experiences over time. For the purpose of this chapter, we equate his mean service rate with his skill level and investigate the impact of such experience-based fluctuations on the agent's performance over a time span of weeks to months.

As a concrete example, suppose company ABC sells a new nanotechnology chemical sensor to customers in two different industries. The company sells a bundled product containing the sensor hardware, cabling, software driver, and a data acquisition program. A service engineer (the *agent*) at the company's customer support centre handles incoming calls regarding this product.

Calls of type x come from equipment suppliers to medical diagnostic labs, which use the sensor to characterise patient fluid and tissue samples. Calls of type y come from biotechnology factories, which use the product as part of a control loop within the fragile manufacturing process for genetically engineered cancer drugs.

If the agent takes calls of type x exclusively, he will become very familiar with the medical users' situation: how often the sensor must be replaced for different uses, how the interface program can be used and customised, and what operating settings give the most accurate results. Over time, his knowledge of these matters accumulates and he becomes a medical customer support specialist.

However, type y callers face a much different situation, with higher volumes of different material flowing by their sensors, higher interference from surrounding equipment, and so on. Experience relevant to type x jobs is not transferable; separate experience with type y jobs is indispensable in order to be proficient at helping those customers.

Depending on the allocation of time and experience, the agent's ability to advise customers on these matters will fluctuate in both speed and quality. Here, we assume a constant unchanging

quality, and focus on the cost impact of the improvement in the speed of answering.

New service tools, new organizational arrangements, and learning effects may all drive changes in his mean service rate. Here we assume all changes are caused by the latter. There is a considerable literature describing the phenomenon of learning curves; among those are recent empirical results describing the effects of learning and forgetting on the job. They make clear that learning and forgetting effects seen in human performance data may play a key role over the short and medium terms; since the job categories themselves keep changing in modern dynamic service environments, learning effects may play important roles over long term time horizons as well.

We develop a simple stochastic model that accommodates such on-the-job learning and forgetting results, and in this paper we present conclusions drawn from our experiments with it that suggest how to assign agents to tasks in ways that improve system performance. A good assignment policy must resolve the following question: with respect to the backlog in the queues, the agent's skill levels, and potential experience-based changes in his skills, from which queue should the agent select his next job?

2.1.2 Operations Research Motivation for the Markov Decision Process Model: Learning and Forgetting Dynamics

Researchers and practitioners have recently highlighted the potential of applying a more fine-grained approach to stochastic learning models. Aksin et al. [2007], Bodreau et al. [2003] and Dietrich and Harrison [2006] examine areas of overlap between operations research (OR) and behavioural science, and call for more investigation of the effects of learning and forgetting. The phenomenon of improved performance through learning is observed in many settings, in both individuals and in organizations (Aksin et al. [2007], Mazzola and McCardle [1997], Reis [1991], Schilling et al. [2003]). Shafer et al. [2001] derived a learning curve model from empirical observations of manufacturing workers that includes both learning and forgetting. Thompson [2007] and Zamiska et al. [2007] con-

tribute new insights into how forgetting mechanisms do or do not manifest themselves, and support the idea that an agent's proficiency in a task drops when that task is neglected.

Several recent papers are notable for their inclusion of agent learning in the analysis of systems. Eitzen et al. [2004] and Sayin and Karabati [2007] consider worker skill development and maintenance as important factors when optimising staffing schedule assignments. Misra et al. [2004] explored the impact of experience effects on the structure of a sales force; Tucker et al. [2007] studied team learning in hospital intensive care units. Whitt [2006] develops analytical means to describe the distribution of performance over a population of agents as their experience increases.

The approach of Gans and Zhou [2002] is similar to ours in that they use a Markov decision process model to explore the effects of learning in a service organization. In their model, employees have a probability of learning and increasing their skill at each decision epoch, and at the same time a probability of leaving the firm due to turnover. As future research they recommended exploring ways to model stochastic learning effects more precisely, as they may significantly affect the firm's performance. In our model each epoch represents a single customer departure (a single job completed) in a queueing model, so we examine the issue at the level of the routing decision. We do not consider turnover.

Hasija et al. [2005], Iravani et al. [2007], Wallace and Whitt [2005], Hopp et al. [2007] and Pinker and Shumsky [2000] evaluate the tradeoffs involved in cross-training workers. In our paper the single service agent is servicing two queues, each with a different job type, requiring two distinct learning curves; in that respect we are also observing the effects of cross-training.

We would like to know the costs that result from the agent's choice of which job class to serve next. We investigate that choice by formulating the problem as a Markov decision process which can be solved using a dynamic programming solver (Bertsekas [1995], George and Harrison [2002], Mazzola and McCardle [1997]). The solver computes each state's cost, which is the sum of the holding costs of both queues for that state plus its cost-to-go to other states, and the agent's best choice may change from one state to the next.

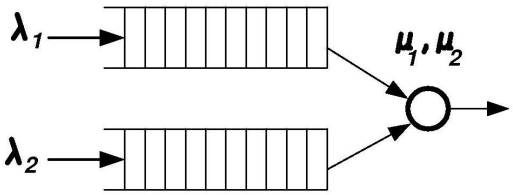


Figure 2.1: **One Server, Two Queues.** The optimal service policy is sought for the one server, two queue case in the presence of stochastic learning (increasing service rate) and forgetting (decreasing service rate).

The customers in one queue are of a different type than customers in the other, so they can be thought of as two separate job classes with different service rates. The system consists of two parallel Markovian FCFS queues with exponentially distributed service and interarrival times. In addition to learning while serving, an agent has a chance of forgetting skills learned for one queue during time periods spent serving the other queue. This behaviour is a simplified version of the *recency* effect discussed in Nembhard and Osothsilp [2001], and Shafer et al. [2001]. Following Hampshire et al. [2006], the extended Kendall notation $M/M_t/1/K$ can be used to describe queues with changing service times such as these. Caro and Gallien [2007] use multiple coupled dynamic programs that include approximation methods for switching costs and other hard-to-model phenomena; that may be a path to expanding upon the results we report here in future work.

If our model did not incorporate learning and forgetting, the optimal policy would be given by the well-known μc rule (Baras et al. [1985], Harrison [1975], Koole [1997], Mandelbaum and Stolyar [2004], Ryokov and Lembert [1967]), which says it is optimal to serve the queue for which the product of the service rate μ and the queue occupancy cost c is a maximum. We find that the μc rule is in fact not the best rule to follow for this problem when the parameters corresponding to learning curve effects are significant.

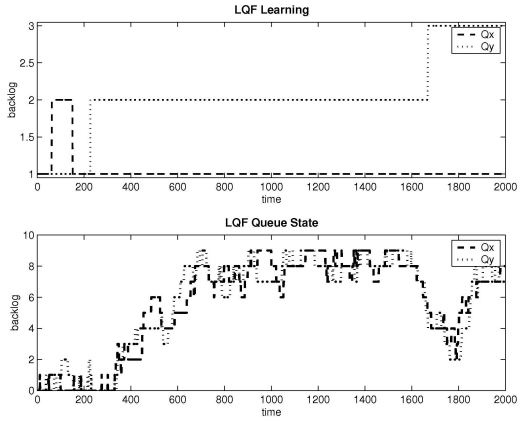


Figure 2.2: **Longest Queue First Time Series.** Time series sample for the stationary policy *LQF*, or "serve the Longest Queue First." Note that the *LQF* rule does not respond to changes in the service rate.

2.2. Model Formulation and Analysis

2.2.1 Adding Learning Effects to the Model

Gans and Zhou [2002]'s call centre learning study considers three levels of agent performance: baseline, a 40% improved service rate, and an 80% improved service rate. Following this we assign the vector of improving service rates for μ_x and μ_y to be $m \cdot [1, 1.4, 1.8]$, where the base service rate is $m \approx 0.25$ jobs per minute.

Shafer et al. [2001] provides a wealth of data, in particular the average learning and forgetting rates for a group of workers conducting a detailed assembly line operation. The testing station workers incurred forgetting effects during interruptions in their work assignments, although they did not switch between two different tasks, as we assume here. In order to incur forgetting losses in our Markov chain model with three learning curve states, forgetting transitions incur a performance penalty by pushing the agent back to less proficient states. We then analyse performance using metrics based on the long-run steady state distribution of the chain.

Their average worker might achieve his highest learning curve state after completing 1,000 jobs, while others learned much faster. Our learning curve has three states, and we allow the mean jump

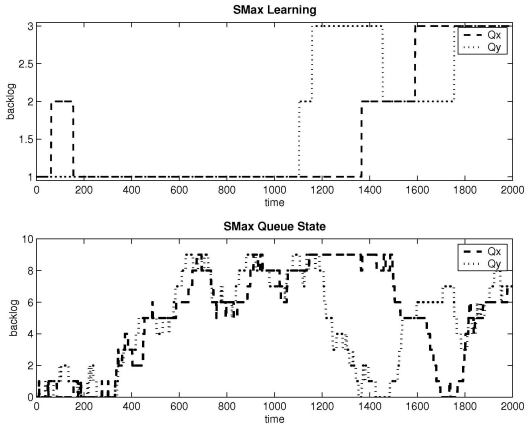


Figure 2.3: **SMax Time Series.** Time series sample for the stationary policy *SMax*, or "serve the job class for which the agent is fastest." Here long stretches of time can occur where the other job class is not served.

rate from a slower service rate to a faster one to range from $(\frac{1}{500} \cdot m)$ up to $(\frac{1}{2} \cdot m)$. So with regard to the data from the product testing study, we are focusing on workers with above average learning rates. We use various values for the forgetting rate that are within the range suggested by the study, with our default forgetting rate being one-third of the learning rate.

2.2.2 Policy Definitions and Time Series Examples

We analyse five policies that could be used to direct the agent's work.

1. Serve the longest queue first, or the *LQF* policy, Figure 2.2.

2. Serve the job class that the agent is most skilled (fastest) at handling, or *SMax*, Figure 2.3. This could be considered a policy of specialisation in Pinker and Shumsky [2000]. Note that such a policy is preferred in many settings. In policies for managing priority queues, this is the *shortest processing time* rule, or *SPT* (Gross and Harris [1998], Schrage and Miller [1966], Tezcan [2006]).

3. Serve the job class that the agent is slowest at handling, or *SMin*, Figure 2.4. This could be considered a policy of cross-training.

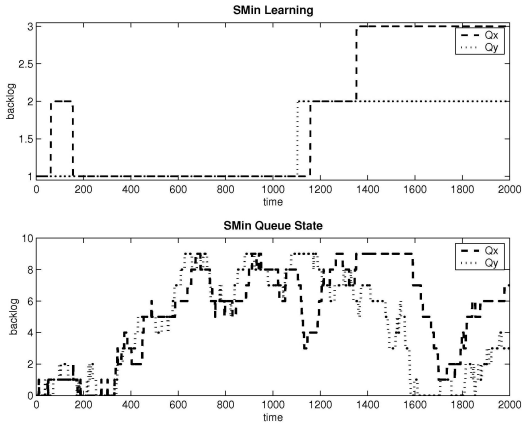


Figure 2.4: **SMin Time Series.** Time series sample for the stationary policy *SMin*, or "serve the job class for which the agent is slowest." It is the reverse of *SMax*. This policy accrues a higher cost than others, but maximises the average capacity of the system.

4. Serve the job class specified by the μc rule, Figure 2.5.

5. Serve the job class specified by the dynamically optimal cost service rule, or *Opt*, Figure 2.6. *Opt* is the only one of the five for which policy iteration is used to alter the policy choice at each state according to the step-by-step value iteration results from the dynamic program.

2.2.3 Dynamic Programming for Optimization

The dynamic programming solver computes the optimal policy to follow for each state in a discrete Markov chain. As a Markov decision process (MDP), a decision maker acts to minimize the expected cost of the present value at time zero of the stream of costs incurred. Assumptions made during the design of the solver include:

- Advancing in skill does not depend on how long the agent has been in a state, but only if the last job served was relevant to this learning curve. Therefore learning and forgetting are memoryless state transitions, which together with the queueing processes form a four-dimensional Markov chain.

- The agent attends one of the two queues at all times, with no breaks or server vacations.

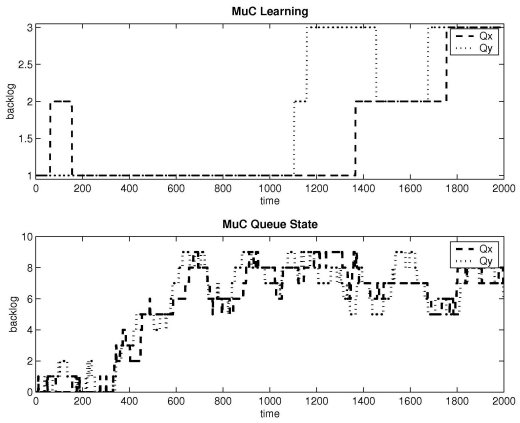


Figure 2.5: **μ-c Time Series.** Time series sample for the stationary policy μ-c, or the μ-c rule: serve the job class for which the product of the queue length and service rate is a maximum. It is similar to but more flexible than *LQF* in permitting some jobs to accumulate in one queue while the other is served.

- There is a finite number of states and an infinite planning horizon. Discount factor β (where $0 < \beta < 1$) discounts the future cost of visiting other states.

- i denotes one of the states, where each i is an index number for a uniquely valued tuple $\{q_x, q_y, \ell_x, \ell_y\}$. Here q_x is the number of customers in the first queue, q_y is the number of customers in the second queue, ℓ_x is the learning state (specifying a service rate μ_x) of the server for type x jobs, and ℓ_y is the learning state of the server for type y jobs.

- The number of valid transitions and their relative probabilities are determined by the policy u adopted for that state, $u \in \{u_x, u_y, u_{tie}\}$. Under u_{tie}, ties in the DP solver are resolved by including transition probabilities for serving both queues and renormalizing.

- The static cost is the sum of the number of customers waiting in the two queues at each state. The holding cost of the backlogs is assumed to be \$1 per waiting job per unit time. Customers to not balk or renege. Arrivals to full queues are turned away, and a one-time penalty $M = \$100$ is charged to the system to represent the impact of dropping a customer. The number of waiting positions in the two buffers is $K_x = K_y = 9$.

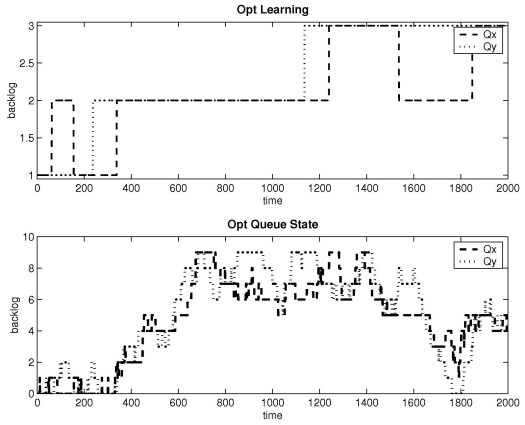


Figure 2.6: **Optimal Policy Time Series.** Representative time series sample for the optimal policy, as computed by the dynamic program using a combination of value and policy iteration.

- $J(i,u)$ is Bellman's equation, the cost functional equation for state i and policy u. The dynamic program iterates until $|(J^k(i,u) - J^{k-1}(i,u))| < \epsilon$, for every state i and policy u, and reports the final value vector $\vec{J}^*$ for each of the five policies.

The process of uniformisation is used to convert the continuous time queueing problem to discrete time. Five exponential processes govern state transitions, and the components of the uniformisation constant ν are the largest values of those processes found among all of the model states. Let a '*' next to a parameter denote its largest value anywhere in the model. The five parameters are: λ_x, the arrival rate to x; λ_y, the arrival rate to y; μ, the departure rate from one of the queues (not both); ϕ, the learning rate; and ψ, the forgetting rate. Then the fastest exponential race transition out of any state is $\nu = \lambda_x^* + \lambda_y^* + \mu^* + \phi^* + \psi^*$. In the experiments the arrival rates for both queues are fixed and equal to each other, so $\lambda_x = \lambda_y = \lambda_x^* = \lambda_y^*$. For each individual state, we have $\nu_i(u) = \lambda_x + \lambda_y + \mu_i(u) + \phi_i(u) + \psi_i(u)$.

2.2.4 Structure of the Transition Matrix $p(u)$

Consider Figure 2.7, which illustrates the state transition "serve queue x" for u_x in the model state $J^k(q_x, q_y, \ell_x, \ell_y, u_x)$. The diagram shows three of the waiting positions, plus empty, for four

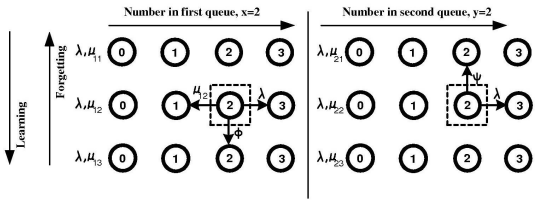


Figure 2.7: **MDP State Transitions.** Joint model state $J(q_x, q_y, \ell_x, \ell_y, u_x)$, with $q_x = 2$, $q_y = 2$, $\ell_x = 2$, and $\ell_y = 2$. The probabilistic transitions under the policy "serve queue one", u_x, are shown. Learning, service, and an arrival are possible for queue x, while forgetting and an arrival are possible for queue y. Self-transitions at each queue are also possible. The activities of the two queues are considered to be independent.

queue states on each side, and it shows three learning states for each job type. Learning, service, and an arrival are possible for queue x, while forgetting and an arrival are possible for queue y. Self-transitions at each queue are also allowed due to uniformisation. The activities of the two queues are considered independent. Bellman's equation then becomes

$$\begin{aligned} J^k(q_x, q_y, \ell_x, \ell_y, u) &= \frac{1}{\beta+\nu}\left[q_x(i) + q_y(i) + (\nu - \nu_i(u)) \cdot J^{k-1}(q_x, q_y, \ell_x, \ell_y, u^*)\right] \\ &\quad + \frac{1}{\beta+\nu}\left[v_i(u) \cdot \Big(\sum_j p_{ij}(u) J(j)\Big)\right]. \end{aligned}$$

Here $q_x(i)$ and $q_y(i)$ are the fixed queue holding costs in state i. To minimize the right-hand side of this equation we seek a policy in each state such that the cost of the policy-dependent terms is a minimum. In the cost-to-go term, the transition probabilities p_{ij} take the form $\left(\frac{\lambda_x}{\lambda_x+\lambda_y+\mu_x+\phi_x+\psi_y}\right)$. This particular p_{ij} value is for an arrival of type x, for a policy "serve queue x"; the other p_{ij}s are similar but with the appropriate value in the numerator.

2.2.5 Steady State Measures

The transition matrix $p = [p_{ij}]$ is constructed in such a way that it is an ergodic Markov chain, and so it has a stationary distribution $\vec{\pi}$ defining the steady-state probability of being in each state. The dynamic programming solver finds this distribution.

Based on $\vec{\pi}$, the mean learning state value $\ell_{\vec{\pi}}$, mean queue backlog $(q_{x\vec{\pi}} + q_{y\vec{\pi}})$, and total

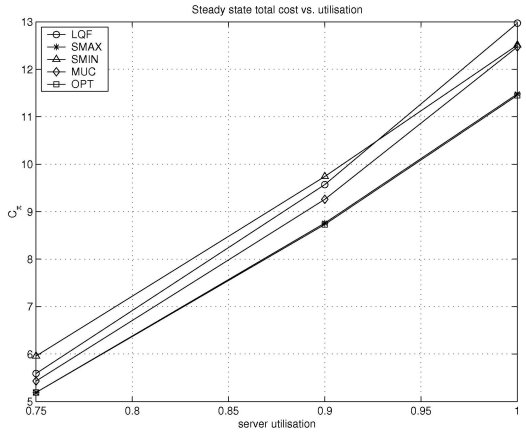


Figure 2.8: $C_{\vec{\pi}}$ **for Five Policies.** Plot of the steady state total cost $C_{\vec{\pi}}$ values from Table 2.3 for all five policies. Section 2.2.5 defines $C_{\vec{\pi}}$ as $C_{\vec{\pi}} = \sum_{i=1}^{N} \pi(i) \cdot J^*(i)$; it represents the steady-state cost of state i's queue backlog, and the discounted cost of state i's neighboring states.

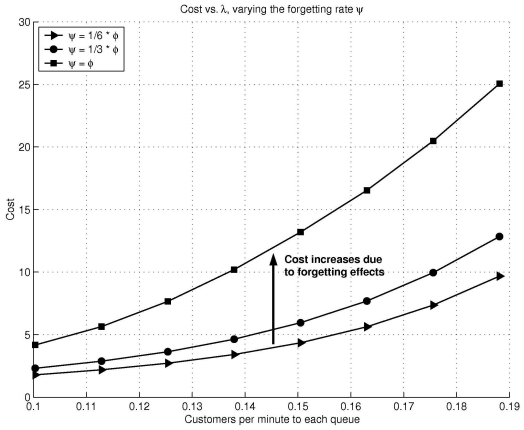


Figure 2.9: $C_{\vec{\pi}}$**, Varying Forgetting.** Plot of the combined steady state cost $C_{\vec{\pi}}$ vs. the arrival rate into the system, using the optimal dynamic policy *Opt*, while varying the forgetting rate ψ. Increased forgetting increases the holding costs in the queue through lower average service rates. The learning rate for all cases is set to *high*.

cost for all states in the system $C_{\vec{\pi}}$ are computed. $\sum_{i=1}^{N} \pi(i) = 1$, so $\vec{\pi}$ serves as a means of weighting the values of $\ell(i)$, $q(i)$, and $C(i)$ according to the steady state likelihood of the system being in state i.

Recall that the fixed cost portion of each state's cost is the sum of the two queue backlogs; $(q_{x\vec{\pi}} + q_{y\vec{\pi}}) = \sum_{i=1}^{N} \pi(i) \cdot [q_x(i) + q_y(i)]$. $C_{\vec{\pi}}$ includes this fixed cost, and also includes the cost-to-go to neighbours at each state, with penalties if applicable: $C_{\vec{\pi}} = \sum_{i=1}^{N} \pi(i) \cdot J^*(i)$. So the performance metric $(q_{x\vec{\pi}} + q_{y\vec{\pi}})$ is a part of the metric $C_{\vec{\pi}}$, and typically $C_{\vec{\pi}}$ will be roughly twice as large.

Note that in the cases we consider here, the learning, forgetting, and service mechanisms affect each job class equally. Thus the steady state metrics that describe experimental outcomes in the next section are symmetric in job classes x and y. In the following sections the terms *long run, steady state, $\vec{\pi}$-weighted,* and *mean* are equivalent.

Each experiment is considered an exact computation for a hypothetical agent who has those characteristics. Truncation error from limiting the number of DP solver iterations is approximately 1e-9 for the value function J in each state, and roundoff error incurred computing $\vec{\pi}$ is near 1e-16 for each state.

2.3. Experimental Results

2.3.1 Experimental Parameters

Simulation inputs are a specific, unchanging dynamic programming reward structure, Table 2.1, combined with specific values of learning parameters that are allowed to vary, Table 2.2. The experiments we present have three learning curve states and ten queue states (including empty) for two job classes.

Note that the learning rate ϕ varies from $(\frac{1}{500} \cdot m)$ up to $(\frac{1}{2} \cdot m)$ in these experiments. The agent therefore steps up the learning curve roughly at rate $\frac{\phi}{m}$, or from once every 500 to once every two jobs completed. Unless stated otherwise, the forgetting rate is $\psi = \frac{\phi}{3}$. The value $\phi = \frac{1}{500} \cdot m$ is referred to as a *moderate* learning rate case; $\phi = \frac{1}{32}$ is *high* learning; and $\phi = \frac{1}{2}$ is *very high* learning.

Table 2.1: **Chapter 2 Fixed Simulation Parameters.** Parameter settings for all simulation runs, which define the fixed reward structure.

Parameter	*Value*
Number of service agents	1
Number of job (queue) types	2
Number of queue buffer states $K = K_x = K_y$	9
Number of learning curve states	3
Discount factor β	0.1
Total # of dynamic programming states N	1800
Minimum service rate m	0.25 departures per minute
Improving service rates μ_x, μ_y	$[m \;\; 1.4m \;\; 1.8m]$

Server utilisation is computed as the fraction $\frac{\lambda_x+\lambda_y}{E[\mu]}$. Here the expected service rate $E[\mu]$ is the service rate at the mean learning curve state for this trial, as weighted by the stationary distribution $\vec{\pi}$.

2.3.2 The Character of the Five Policies

Table 2.3 gives metrics for evaluating the performance of the five policies. The first row under each utilisation value gives the mean queue length in steady state, $q_{x\vec{\pi}} + q_{y\vec{\pi}}$. Performance on this measure usually (but not always) matches the total simulation cost weighted by the stationary distribution, $C_{\vec{\pi}} = \vec{\pi} \cdot \vec{J}^*$, shown on the fifth row. The second row gives the long run service rate $\mu_{\vec{\pi}}$; this is the long run system capacity. Because of the symmetry in the characteristics of job classes x and y, we have $\mu_{x\vec{\pi}} = \mu_{y\vec{\pi}}$. The third and fourth rows measure the impact of buffer overflow on the system. Define $o_{\vec{\pi}} = \lambda\vec{\pi}_{\text{over}}$, where $\vec{\pi}_{\text{over}}$ contains the values of the stationary distribution for states with one or both buffers full. $C_{o\vec{\pi}}$ measures the contribution of those full buffer states towards the total cost $C_{\vec{\pi}}$. Figure 2.8 shows plots of the $C_{\vec{\pi}}$ values from Table 2.3.

Table 2.2: **Variable Simulation Parameters.** Learning and forgetting parameters that were varied in simulations.

Learning Parameter	*Symbol or Abbreviation*	*Sample Value* (m = 0.25 jobs per minute)
Arrival rate	λ_x, λ_y	$0.4 \cdot m$ to $0.75 \cdot m$, or 50% to 100% capacity
Moderate learning rate	ϕ_x, ϕ_y	$\frac{1}{500} \cdot m$
High learning rate	ϕ_x, ϕ_y	$\frac{1}{32} \cdot m$
Very high learning rate	ϕ_x, ϕ_y	$\frac{1}{2} \cdot m$
Default forgetting rate	ψ_x, ψ_y	$(\phi/3) \cdot m$

Each of the five policies does well on at least one of the metrics. *LQF* is the best at minimizing the costs due to buffer overflow. *SMax* is best at minimizing the mean queue backlogs. *SMin* is best at maximizing the system's long run capacity. μ-c is similar to *LQF*, but trades off some of the ability to minimize overflow for a lower total cost, $C_{\vec{\pi}}$, than *LQF*. Finally, *Opt* uses the advantage of policy iteration to achieve the lowest total cost among all of them.

In Table 2.3, with the learning rate set to *moderate*, the characteristics of the five policies start to become significant, giving performance differences of between 0.5% to 10% on the various metrics. This is roughly at the level of the average learner case from Shafer et al. [2001]. At higher learning rates these policy differences are even more pronounced.

2.3.2.1 Learning and Forgetting Effects

The fundamental dynamic driving the results is that increased learning decreases the holding costs in the queue by achieving higher average service rates. Figure 2.9 illustrates this by showing a plot of the long run cost $C_{\vec{\pi}}$ against the arrival rate into the system, using the optimal cost policy *Opt*. Note that if forgetting is absent, learning pushes the service rates to their maxima for every policy in steady state. That trivial result does not occur when forgetting effects are present, and Figure 2.9 shows

Table 2.3: **Table of Policy Comparisons.** Comparison of policies by different $\vec{\pi}$-weighted measures, which are defined in Section 2.2.5. Data is for a *moderate* learning rate; $\psi = \phi/3$; and the system faces arrival rates into both buffers that give 75%, 90%, and 100% server utilisation. The best two policies by each metric are shown in bold. Trends seen here are accentuated at *high* and *very high* learning rates.

75% utilisation						
	Units	*LQF*	*SMax*	*SMin*	μ-c	*Opt*
$q_{x\vec{\pi}} + q_{y\vec{\pi}}$	jobs	2.83	**2.44**	2.67	2.73	**2.48**
$\mu_{x\vec{\pi}}$, $\mu_{y\vec{\pi}}$	jobs/min	1.62	1.62	1.62	1.62	1.62
$(o_{x\vec{\pi}} + o_{y\vec{\pi}}) * 1000$	jobs/min	**1.30**	2.52	3.93	**1.51**	2.25
$C_{o\vec{\pi}}$	cost	**0.76**	1.16	1.76	**0.82**	1.07
$C_{\vec{\pi}}$	cost	5.59	**5.19**	5.96	5.43	**5.19**
90% utilisation						
	Units	*LQF*	*SMax*	*SMin*	μ-c	*Opt*
$q_{x\vec{\pi}} + q_{y\vec{\pi}}$	jobs	4.75	**3.84**	4.22	4.54	**3.96**
$\mu_{\vec{\pi}}$, $\mu_{y\vec{\pi}}$	jobs/min	**1.64**	1.63	**1.65**	**1.64**	1.63
$(o_{x\vec{\pi}} + o_{y\vec{\pi}}) * 1000$	jobs/min	**5.85**	9.06	11.87	**6.48**	8.05
$C_{o\vec{\pi}}$	cost	**2.32**	3.19	3.95	**2.43**	2.88
$C_{\vec{\pi}}$	cost	9.57	**8.76**	9.74	9.26	**8.73**
100% utilisation						
	Units	*LQF*	*SMax*	*SMin*	μ-c	*Opt*
$q_{x\vec{\pi}} + q_{y\vec{\pi}}$	jobs	6.63	**5.23**	5.58	6.33	**5.26**
$\mu_{\vec{\pi}}$, $\mu_{y\vec{\pi}}$	jobs/min	**1.65**	1.64	**1.66**	1.64	1.64
$(o_{x\vec{\pi}} + o_{y\vec{\pi}}) * 1000$	jobs/min	**12.79**	17.83	21.13	**14.09**	17.51
$C_{o\vec{\pi}}$	cost	**4.89**	5.42	6.50	**4.91**	5.34
$C_{\vec{\pi}}$	cost	12.97	**11.48**	12.51	12.47	**11.45**

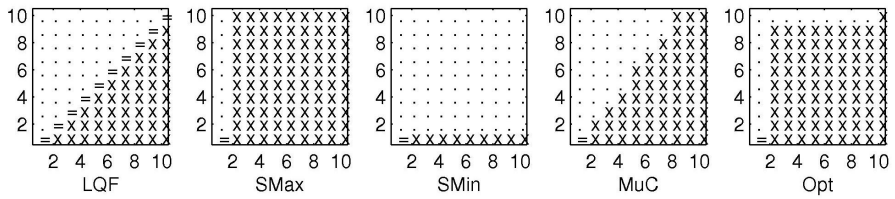


Figure 2.10: **Policy Results Over Queue States.** Side-by-side comparison of policies by observing queue service rules for the same subset of states. Each of the five squares represents the state of queue x on the horizontal and queue y on the vertical axis. The policy in question serves x if an x appears, y if a dot appears, and serves either if an equals sign appears. All five plots show the same unbalanced learning curve state: $\mu_x = 1.8m$, $\mu_y = 1.4m$, and x has a service rate advantage. Here *Opt* behaves like *SMax*, except at the border where *Opt* reacts to the overflow penalty. For balanced learning curve states with $\mu_x = \mu_y$ (not shown), all policies look like *LQF*.

how increasing the forgetting rate ψ increases the cost.

While forgetting can occur in any state, learning-based service rate improvement is not allowed when the system is empty. Higher arrival and utilisation rates thus benefit the agent by giving him a chance to practice his skills and ascend the experience curve.

2.3.2.2 Overflow Penalty Effects

When a customer arrives to a full buffer, she is turned away and denied service. This event triggers a cost penalty of $M = \$100$. The penalty's effect increases if λ_x and λ_y are large in proportion to the other rates. For instance, if queue x is full, the penalty becomes $\left(\frac{\lambda_x \cdot M}{\lambda_x+\lambda_y+\mu_x+\phi_x+\psi_y}\right)$. Its impact on $C_{\vec{\pi}}$ is further adjusted by how often the state is visited and the mean length of each visit (from $\vec{\pi}$, ν and $\nu(i)$).

Figure 2.10 illustrates the effect of the buffer overflow penalty over the five policies. Only *Opt* is able to react to the increased cost in full buffer states and reduce its cost by choosing a different queue. This is reflected in Table 2.3 where $o_{\vec{\pi}}$, $C_{o\vec{\pi}}$, and $C_{\vec{\pi}}$ are consistently greater for *SMax* as compared to *Opt*. This is noticeable at higher values of ϕ as well. For instance, at a *high* learning rate and 75% utilisation, *SMax* incurred a 1.5% higher cost $C_{\vec{\pi}}$ than *Opt*.

Table 2.4: $\vec{\pi}$ **Over Learning States.** Values of the stationary distribution $\vec{\pi}$ for policies *SMax*, *SMin*, and μ-c. They are organized according to the probability mass found at each learning curve state. Subscript *1* indicates the lowest service rate, and subscript *3* the highest. The optimal policy follows the *SMax* pattern below closely; *LQF* follows the μ-c pattern closely. The system utilization was set to 90% of capacity, ϕ was *very high*, and ϕ was set equal to ψ. When $\phi = \frac{1}{3}\psi$, similar distributions are obtained, but with a skew toward the northeast corners because the forgetting rate is smaller.

SMax				**SMin**				**μ-c**			
y_3	**0.24**	0.08	0.05	y_3	0.05	0.09	**0.14**	y_3	0.17	0.09	0.07
y_2	0.09	0.06	0.08	y_2	0.11	**0.16**	0.09	y_2	0.11	0.08	0.09
y_1	0.09	0.09	**0.24**	y_1	**0.20**	0.11	0.05	y_1	0.11	0.11	0.17
	x_1	x_2	x_3		x_1	x_2	x_3		x_1	x_2	x_3

2.3.2.3 Approximating the Optimal Policy

The dynamic optimal cost policy *Opt* is found at each state by a combination of value and policy iteration in the DP solver. *Opt* finds the globally optimal policy within a reasonable time for policies of this size, balancing the different costs in the best possible combination.

For all the learning parameter ranges we study, the (static) policy *SMax* turns out to be a very good approximation to *Opt* for the purpose of minimizing the total cost. However, *SMax* does not respond well to cost changes at boundaries (Figure 2.10). Section 2.3.2.5 describes cases where μ-c is closer to *Opt* by the metric $C_{\vec{\pi}}$.

2.3.2.4 Specialisation Versus Cross-Training

Figure 2.11 and Table 2.4 give more insight into the difference between *SMax* and *SMin*, or the difference between specialising and cross-training.

On the left side of Figure 2.11, *SMin* gives up to a 5% improvement over *SMax* in long run system capacity $\mu_{\vec{\pi}}$ as the learning rate is varied, for a system at 100% utilisation. The forgetting rate is held constant at $\psi = \frac{1}{12} \cdot m$, or one-sixth times the *very high* learning rate. While *SMin* was the best policy for boosting $\mu_{\vec{\pi}}$ over the long run, *LQF* never differed by more than 3% from *SMin* by this metric; *LQF* itself can be considered a good cross-training policy. In fact *LQF* may be preferred in

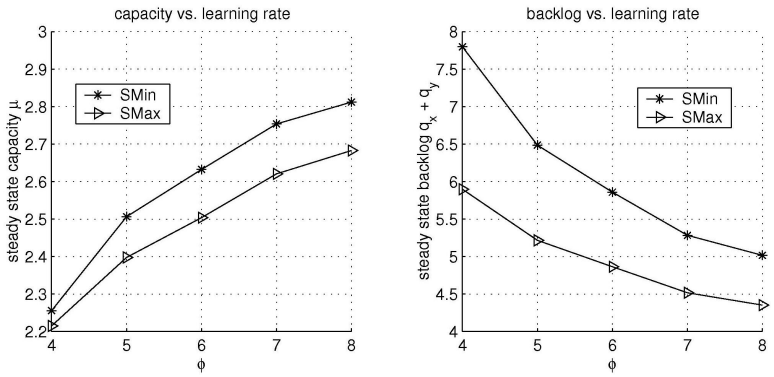


Figure 2.11: **Long Run System Capacity, Backlog.** Left: plot of long run system capacity $\mu_{\vec{\pi}}$ (note that $\mu_{x\vec{\pi}} = \mu_{y\vec{\pi}}$) at 100% utilisation, showing a metric for which *SMin* is superior to *SMax*. Right: plot of long run system backlog $(q_{x\vec{\pi}} + q_{y\vec{\pi}})$ at 100% utilisation, showing a metric for which *SMax* is superior to *SMin*. Learning is swept from *high* to *very high* rates. Forgetting is held constant at $\psi = \phi_{vh}/6$, where ϕ_{vh} is the *very high* learning rate.

practice since it does not require an estimate of the agent's skill level.

The right side of Figure 2.11 shows the system steady state backlog, $(q_{x\vec{\pi}} + q_{y\vec{\pi}})$, a metric for which policy *SMax* performs better. *SMin* generates a 25% to 40% worse value for the backlog, a metric for which the trends are usually aligned with the total system cost $C_{\vec{\pi}}$. Here ψ is held at the same value as on the left.

SMax generates a centrifugal trend in service rate improvement that discourages a balanced skill set, while *SMin* promotes balanced skills. These trends are evident in the steady state distributions $\vec{\pi}$ for those policies, which are described in Table 2.4. In all our experiments *SMax* performed better than the other policies as a means of minimizing $(q_{x\vec{\pi}} + q_{y\vec{\pi}})$.

2.3.2.5 Performance Penalties from Service Level Agreements

Up to now we have used a simple holding cost for the queue that is linearly increasing in the size of the backlog, and in fact is equal to the size of the backlog in dollars per minute. But organizations such as call centres may be contractually obligated to provide a certain service level, with financial penalties for nonperformance. These penalties will be assessed by observing if customer

waiting times are too long.

For this example consider queue backlogs to be proxies for customer waiting times. Then a cost penalty is assessed whenever the queue length goes beyond a threshold. In Table 2.5, our two buffers are each segmented into low cost states (from zero to threshold K_{th}) and high cost states (from $K_{th}+1$ to K). An additional fixed cost penalty $M = \$25$ per minute is assessed for being in each of the high cost states. The learning rate is set to *high*, and utilisation is 75%. In all cases policy *Opt* has the lowest cost $C_{\vec{\pi}}$, and we would like to know which of the simpler policies best approximates this cost. With these parameters and no thresholds, we found that this was *SMax*.

When the threshold K_{th} is low, and most of the queue states accrue penalty costs, then all the policies suffer about equally. When the threshold is four fifths or more of the buffer size, μ-c and *LQF* have lower costs than *SMax*. Yet note that *SMax* still gives the lowest mean steady state queue length $q_{\vec{\pi}}$.

The problem for *SMax* is that the agent leaves one queue alone for long periods, and the unattended job class slips into penalty-accruing states often during his absences. The more equitable policies μ-c and *LQF* reduce these absences – visible in the time series of Figure 2.2 and Figure 2.3 – and so perform better than *SMax* here. This example demonstrates that the best steady state cost $C_{\vec{\pi}}$ may not agree with the lowest mean steady state backlog ($q_{x\vec{\pi}} + q_{y\vec{\pi}}$); $C_{\vec{\pi}}$ is a better measure of the impact of server absences when those absences are costly.

The act of reducing the buffer sizes will produce a similar result to the imposition of performance penalties described above; μ-c and *LQF* become more attractive. On the other hand, preliminary data indicate that *SMax* retains its superiority in runs with larger buffer sizes.

2.4. Conclusions from the MDP Model

In summary, we have examined a model of one agent serving two parallel queues, each queue holding a separate and unrelated type of customer. We seek the best policy to follow, or in

Table 2.5: **Effects of Service Level Agreements.** The best simple policy by the total cost metric $C_{\vec{\pi}}$ may not agree with that given by the sum of the backlogs ($q_{x\vec{\pi}} + q_{y\vec{\pi}}$) when the server is absent for long stretches of time while away serving the other job class.

K_{th}	Lowest ($q_{x\vec{\pi}} + q_{y\vec{\pi}}$)	Lowest $C_{\vec{\pi}}$	Second Lowest $C_{\vec{\pi}}$
4	*SMax*	*SMax*	μ-c
5	*SMax*	μ-c	*SMax*
6	*SMax*	μ-c	*SMax*
7	*SMax*	μ-c	*LQF*

other words which of the two should be served, for each combination of queue states. These queue states impute holding costs for the service system in proportion to the sum of the queue occupancies and penalties for missed jobs. An additional factor in this model is that the agent may increase his service rate through learning-by-doing, and decrease his service rate through forgetting when ignoring a job class. The choice of which to serve thus affects future performance, and makes the policy choice more complicated. We found that policies *SMax* and μ-c were the best approximations to the optimal cost policy in terms of their steady state cost in different situations. *SMin* maximized the steady state capacity of the system, and *LQF* minimized the costs due to buffer overflow.

Here we only study job classes with symmetric learning curves. Issues of asymmetric learning curves and class priorities will lead to different outcomes, due to the inherent nonlinear behaviours in queueing systems. Switching costs in real-world queueing systems are another factor that is complicated to model. There also exist analytical results in applications of queueing theory that could be adapted to investigate the properties of our model in future work (see Fischer and Meier-Hellstern [1993], Grassmann [2003], Meier-Hellstern [1987], Nunez-Queija [1998], Rossiter [1987], and Iravani et al. [2007]). Finally, we would like to scale up to larger systems, with more agents and job classes. Because our dynamic program suffers from the curse of dimensionality when modeling large systems, we explore a more scalable nonlinear programming approach to setting optimal job routing targets in Chapter 5, together with discrete-event simulations to find good routing rules to achieve those targets.

CHAPTER

THREE

Utility Functions in Agent Expertise

Performance Metric	Expertise from Recent Experience	Expertise from Cumulative Experience
Call Handle Time H	**Chapter 2**	**Chapter 4, Chapter 5**
General Expertise X, First Call Resolution Rate R	**Chapter 3** Defines utility functions in agent expertise (X). These functions are useful as metrics for evaluating the distribution of expertise among a group of agents.	**Chapter 4** (R)

A myopic rule for skill-based routing would consider an agent's skill level to be fixed, as determined by formal training or certification. However, the routing rule itself has an impact on skill levels: through on-the-job learning, the development of the agents' expertise depends on the calls they take. In this chapter we develop new tools that allow us to reason qualitatively about the process of acquiring on-the-job expertise by call center agents. First, we show how to link routing decisions to expertise level outcomes. By including forgetting and agent turnover effects along with learning-based improvement, we place natural limits on the asymptotic level of an agent's expertise, and obtain steady-state expressions that may be used to analyze stochastic models of expertise development.

Denoting an agent's expertise by the dimensionless quantity X, $0 \leq X \leq 1$, we then define **utility functions** that depend on expertise, and show how polices of evenly shared routing and extreme specialized routing affect them. Two of these utility functions will be developed further in Chapter 5, which shows a scalable approach for optimizing distributions of expertise in call centers. The first function is the expected value of expertise seen by a customer, which we call the customer's utility, or U_c. The second is the sum of expertise in the system, which we call the supervisor's utility, or U_s. Both functions would seem to be worth maximizing, but we see that optimizing the value of one may mean sacrificing the value of the other. Section 3.4 provides a basic analysis of their convexity properties in preparation for further development as optimization program objectives.

The table of chapter topics above shows how this work fits among the problems explored in other chapters. For more context, also see the full thesis outline, the motivations behind each chapter, and the separate chapter models that are described starting on page 5 in Chapter 1.

3.1. Modeling the Asymptotic Value of Expertise

Consider the evolution of expertise in an agent answering calls to a call center. Let the expertise $X(t)$ of the agent at time t be on a scale $0 \leq X(t) \leq 1$, where $X(t) = 0$ indicates a novice, and $X(t) = 1$ corresponds to an expert. Define the average time between completed jobs to be

τ_d, including the receiving and processing of a job, followed by some time until the next job arrives. Denote the mean steady state *departure rate* of customers from the system by λ, so that $\lambda = 1/\tau_d$. In Section 3.2 we develop a model that takes advantage of the reversibility property of M/M/1 queues to equate λ to both the mean steady state *arrival rate* of jobs to an agent as well as departures from an agent. We assume that the agent learns while processing the job (on-the-job), thus increasing his expertise level X(t), and forgets while not processing, leading to a decrease of X(t).

First, consider the learning dynamic. The agent's expertise by processing one job increases through

$$X(t_n) \mapsto X(t_{n-1}) + \alpha(1 - X(t_{n-1})) \tag{3.1}$$

where α is a learning parameter, and improvements in expertise are recorded at the instant of each *customer departure time* t_n. Then the experience gain is proportional to $(1-X(t_{n-1}))$, and so becomes geometrically smaller as $X(t_n)$ approaches expert status. In the absence of forgetting, an agent will move from novice to roughly half of her maximum possible level by completing 1/α jobs. Figure 3.1 illustrates this accrual of expertise for an agent responsible for a single queue of customers desiring service.

Here expertise is increasing in an agent's relevant *cumulative production*. We define this as the index n of the n-th departure handled by the agent. Of course, a true accounting of the knowledge-building process for a person working as a call center agent would involve a complex nonlinear function. Fortunately, empirical studies justify a simpler model of concave increasing expertise functions with different parameters for each agent and task type (Badiru [1991], Shafer et al. [2001]). Our stylized expertise model develops as a concave increasing function of cumulative production by awarding an expertise increase proportional to the proximity of current expertise $X(t_n)$ to a long-run expected expertise value X_∞.

Suppose that skills need to be maintained through reinforcement; in the absence of work to occupy an agent, forgetting ultimately reduces the expertise of the agent to zero (novice level). Thus expertise depends on *recent production*, as illustrated in Figure 3.2. We assume that forgetting occurs

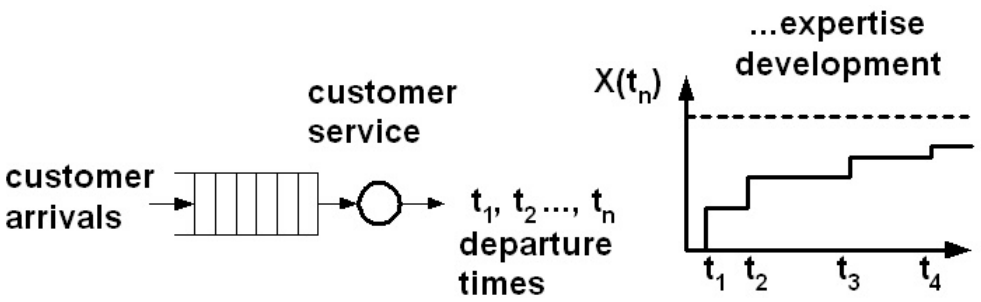


Figure 3.1: **Expertise Accrues With Cumulative Production.** Plot of expertise development for a single agent responsible for a queue of customers. The plot on the right illustrates Relation (3.1). Here, the agent's expertise develops according to the irregular stair-step plot, with an increase recorded at the instant of each customer departure t_n. The amount of this increase diminishes as the production record grows, and is bounded above by the dotted line, which represents a perfect expertise value of one. The dotted line also represents the asymptotic limit of the learning-only model (a model without forgetting).

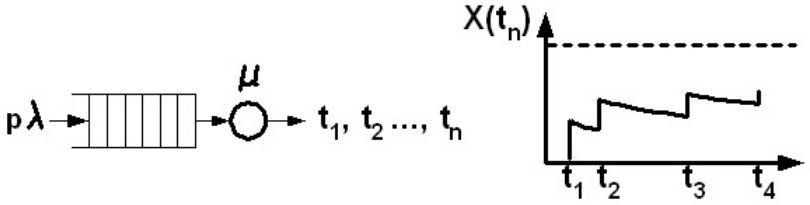


Figure 3.2: **Expertise Accrues with Recent Production.** As Figure 3.1, but for a single agent responsible for a queue of customers under both learning and forgetting mechanisms. The plot on the right illustrates Equation (3.2). Here, the agent's expertise develops according to the irregular stair-step plot, with an increase recorded at the instant of each customer departure t_n, and a slow but steady drop in expertise in the gaps between departure times. We refer to the asymptotic limit as X_∞. The arrival rate $p\lambda$ is under management's control, as defined in Equation (3.3).

at a continuous rate β, so that for a period of length $\tau_n = t_n - t_{n-1}$, the expertise is discounted by $e^{-\beta\tau_n}$. This is one type of function among several that have been used to fit the measured effects of forgetting (Nembhard and Osothsilp [2001]); we apply it here for its simplicity and tractability. Taking learning events and continuous forgetting together, we get:

$$X(t_{n-1} + \tau_n) = (X(t_{n-1}) + \alpha(1 - X(t_{n-1}))e^{-\beta\tau_n}. \tag{3.2}$$

Learning is designed to be a geometrically decreasing concave function of time, and the forgetting exponential function is convex in time for positive τ_n, which holds for the cases we consider. From prior studies of learning and forgetting rates, and our own observations of call center data, we

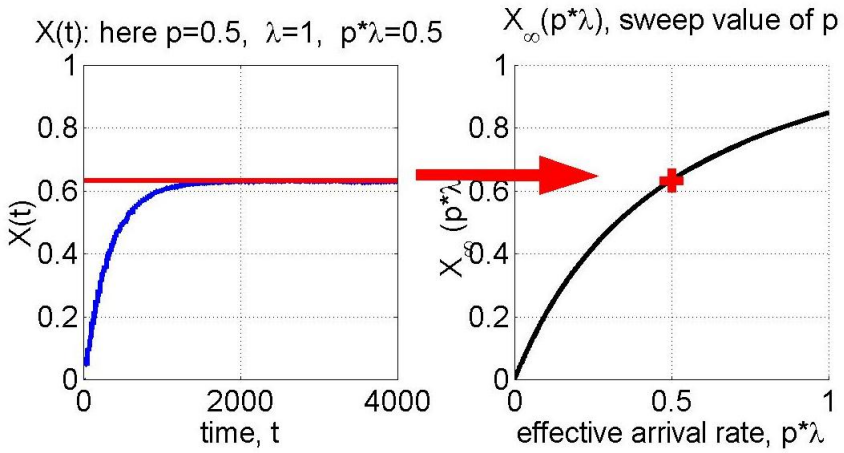


Figure 3.3: **Long-Run Expertise as a Continuous Function of the Arrival Rate.** The long-run value of expertise. The key advantage of our formulation: expertise becomes a continuous function of the routing proportion p. Here $\alpha = 2e - 3$, $\beta = 8e - 4$, $\mu = 10$, $\lambda = 1$, and $p = 0.5$.

expect a reasonable range of interest for parameter α to be between 1e-4 and 0.1. We take $\beta \leq \alpha$, so that forgetting happens at a slower rate than learning.

A key issue still remains at this point: what processes govern the customer arrival and service rates? As we noted in Chapter 2, decisions by individual customers about when to call often appear random to the agent. Empirical studies show that customer arrivals to contact centers are well modeled by nonhomogeneous Poisson processes; and over specific time intervals an analytically tractable homogeneous Poisson process can be a good model (Cleveland and Mayben [2000], page 58; Brown et al. [2002], page 39; Gans et al. [2003], pages 125–126). In some call centers the service time may also be acceptably modeled by an exponential distribution, although that is not always the case (Gans et al. [2003], page 127; Mehrotra and Fama [2003], page 138). In this Chapter, we will take both the arrival and service processes to be Poisson. These are not the best distributional assumptions for every facility, but they are very useful as theoretical approximations that allow values of agent expertise to be computed with the aid of steady-state results for Markovian queueing models. Section 3.2 provides an expression for X_∞ as a steady-state expected value under Markovian assumptions (Equation (3.8), page 46).

Figure 3.3 depicts the evolution of our expertise equations over time: on-the-job experience grows through serving a sequence of customers. The left-hand plot of Figure 3.3 depicts this trend. A longer interdeparture time will incur more forgetting, a short one less, but over many customer visits the variations average out and $X(t_n)$ settles to a long-run expertise level X_∞. The right-hand plot shows the range of X_∞ for this agent. In this example, the long-run value of expertise maps from the red line on the left plot to the red cross on the right. The position of the cross depends on the agent's learning and forgetting characteristics, and on the arrival rate of customers that management sees fit to sent to this agent for service. This illustrates a key feature of our expertise formulation: *we establish a link between the rate of jobs completed by agent and his expertise level.* Management can take advantage of this linkage to design routing rules that optimize the distribution of expertise among the workforce.

Consider a random process model of an agent's work in which *the steady state average rate of departures (λ) equals the steady state average rate of arrivals.* Section 3.2 below will give a precise description of such a model. Accepting that premise for the moment, in Figure 3.3 we let λ be the rate of Poisson arrivals coming into the entire contact center, which may employ many agents. Let the rate to any single agent be a thinned Poisson stream of arrivals, with a mean arrival rate that a will be a fixed fraction of λ. Let this fraction of λ arriving at a single agent be denoted by p, with $0 \leq p \leq 1$. Note that τ_n in Equation 3.2 is the time between two departures; here assume the agent is a server in an M/M/1 system in equilibrium, so that $\tau_n \sim \text{Exp}(p\lambda)$. Then we may modify the definition of the expected interdeparture time to be

$$E[\tau_n] = \tau_d = \left(\frac{1}{p\lambda}\right). \tag{3.3}$$

Management controls the proportion p for each agent through routing rules, hence controlling the intensity of on-the-job learning experiences, and ultimately the agent's expertise level. We will show the expected long-run value of the agent's expertise is $X_\infty(p\lambda)$. Where the value of λ is understood to be a certain value, we can just write $X_\infty(p)$. Note that although $X(t_n)$ was a function of discrete departure times t_n, $X_\infty(p\lambda)$ may be a continuous function of a continuous real variable p,

and is thus convenient for analysis.

With this definition of p, the left side of the example in Figure 3.3 shows $X(t_n)$ increasing as customers are served, until the steady state value $X_\infty(p\lambda)$ is reached—the horizontal line at about 0.63. The right side shows the value of $X_\infty(p\lambda)$ as the routing proportion p to this agent is swept from zero to one.

Now we would like to estimate expertise levels for agents where the operating environment is modeled by a stochastic queueing system. In the next section we derive an explicit link between the level of asymptotic expertise and the arrivals to an agent $p\lambda$ when those arrivals are Poisson.

3.2. Expected Value of Expertise When Arrivals and Service are Exponentially Distributed

Here we apply the assumption underlying the most commonly used system capacity model: we let interarrival times be exponentially distributed, $\sim$ Exp(λ); and we let service times be $\sim$ Exp(μ). We determine the expected value of expertise—which is also the asymptotic value of expertise—as a function of these random variables, the agent's learning and forgetting characteristics, and management's assigned routing proportion p.

3.2.1 Definitions

Consider the situation where an agent provides service to customers one at a time. We observe each customer leaving the agent at the moment their service encounter is completed.

Definition 3.2.1. *Definition of departure times t_n.*

Let departure time t_n denote the moment in time when the nth customer leaves following service. Here t_n is a positive real number, $t_n \geq 0$; and $t_1 < t_2 < ... < t_n$. □

Definition 3.2.2. *Definition of the learning rate α.*

Let $0 < \alpha < 1$ be known as the *learning rate*. $\square$

Definition 3.2.3. *Definition of the forgetting rate β.*

Let $0 \leq \beta \leq \alpha$ be known as the *forgetting rate*. $\square$

Definition 3.2.4. *Definition of the forgetting factor θ_n.*

Let $\tau_n = t_n - t_{(n-1)}$ be the difference between successive departure times. Then let the *forgetting factor* be the quantity $\theta_n = e^{(-\beta\tau_n)}$.

Note that $0 \leq \theta_n \leq 1$. The forgetting factor θ_n will take values close to one when the product of τ_n and β is small, and forgetting is not significant. $\square$

Definition 3.2.5. *Definition of the routing proportion p.*

Here $0 \leq p \leq 1$ is a parameter that acts to thin a Poisson process. In other words, assume we are given $N \sim$ Poisson(λ); then let $M \sim$ Binomial(N, p), so that $M \sim$ Poisson$(p\lambda)$. Therefore $\lambda \geq p\lambda$, and we know that Poisson process M is characterized by a slower rate $p\lambda$ that is a proportion of the original rate λ. In our models p is a control parameter, and we will refer to it as the *routing proportion*.

Definition 3.2.6. *Definition of expertise $X(t_n)$ that increases due to learning, and decreases due to forgetting.*

Let $X(t_n)$ be known as the *expertise* of the agent. This expertise function maps the sequence of discrete departure time differences to a sequence of expertise values between zero and one, $\mathcal{R}^+ \rightarrow [0, 1]$, according to the following definition.

$$X_n = [X_{n-1} + \alpha(1 - X_{n-1})]\,\theta_n. \tag{3.4}$$

3.2.2 The Asymptotic Value of Expertise

Here we find the long-run expected value of expertise, or the *asymptotic value of expertise*, for an agent who serves an M/M/1 queue of customers on a first-come, first-serve basis. We take expertise to develop according to Equation (3.4) with learning and forgetting mechanisms.

We will need the following three results to find the asymptotic value of expertise. First, Lemma 3.2.7 and Lemma 3.2.8 show that the time sequence of departing customers has a convenient analytical description.

Lemma 3.2.7. *In an M/M/1 queueing system with arrival rate $p\lambda$ and service rate μ, where $(p\lambda)/\mu < 1$, the sequence of departures is a Poisson process with rate $p\lambda$.*

Proof. This is known as the reversibility property of the M/M/1 queue. See for example Durrett [1999], page 184. Note that we define the arrival rate to this queue as a thinned Poisson process, $p\lambda$, implying that a stable percentage p of a greater traffic stream is directed to this queue for service by this agent. □

Lemma 3.2.8. *In the system of Lemma 3.2.7, the interval between successive departure times is an exponential distribution with parameter $p\lambda$.*

Proof. The proof examines the conditional expectation of the last departure time as it depends on the next-to-last time, and shows they are statistically independent, leading to an exponential distribution for the time between the two events. For the full details see Stirzaker [2003], page 366. This fact is known as the independent increments property of the Poisson process. □

Now we know that $\tau_n \sim \text{Exp}(p\lambda)$, which allows the derivation of a closed-form expression for the expected value of the forgetting factor. (Recall our concise representation for the forgetting factor, $\theta_n = e^{(-\beta\tau_n)}$.)

Lemma 3.2.9. *When $\tau_n \sim \text{Exp}(p\lambda)$, then $E[\theta_n] = \frac{p\lambda}{\beta+p\lambda}$.*

Proof.

$$\begin{aligned} E[e^{-\beta\tau_n}] &= E[\theta_n] = \int_{\tau_n=-\infty}^{+\infty} e^{-\beta\tau_n} p\lambda e^{-p\lambda\tau_n} d\tau_n \\ &= \frac{-p\lambda}{\beta+p\lambda} \int_{\tau_n=0}^{+\infty} e^{-\tau_n(\beta+p\lambda)} \cdot (-1) \cdot (\beta+p\lambda) d\tau_n \\ &= \frac{p\lambda}{\beta+p\lambda}. \end{aligned} \tag{3.5}$$

□

Now we can state the main result.

Theorem 3.2.10. *The long-run expected value of expertise,* $X_\infty = \lim_{n\to\infty} E[X_n]$*, is given by* $\frac{p\lambda\alpha}{\beta+p\lambda\alpha}$.

Proof. We start with difference equation for expertise based on departure times, X_n, and work to find its expected value in the long run, $\lim_{n\to\infty} E[X_n]$.

$$\begin{aligned} X_n &= [X_{n-1} + \alpha(1 - X_{n-1})]\,\theta_n \\ X_n &= X_{n-1}(1-\alpha)\theta_n + \alpha\theta_n \end{aligned}$$

Note that by the independent increments property of the Poisson process, θ_n and X_{n-1} are statistically independent, so that $E[\theta_n X_{n-1}] = E[\theta_n]E[X_{n-1}]$. Also, since values of τ_n are i.i.d. exponential random variables, we know by the monotone convergence theorem for random variables (Stirzaker [2003], page 201) that $E[\theta_n] = E[\theta]$.

$$E[X_n] = (1-\alpha)E[\theta]E[X_{n-1}] + \alpha E[\theta]$$

By defining constants $K_1 = (1-\alpha)E[\theta]$, and $K_2 = \alpha E[\theta]$, we may write $E[X_n]$ as a first-order difference equation.

$$E[X_n] = K_1 E[X_{n-1}] + K_2.$$

See Elaydi [2005], page 17 for a discussion of solutions for this class of equations. Using

the closed-form solution for our difference equation, we can find $\lim_{n\to\infty} E[X_n]$:

$$\begin{aligned} E[X_n] &= \frac{K_2}{1-K_1}\left(1-K_1^n\right) \\ \lim_{n\to\infty} E[X_n] &= \frac{K_2}{1-K_1}. \end{aligned} \tag{3.6}$$

$$\lim_{n\to\infty} E[X_n] = \frac{\alpha E\theta}{1-E\theta+\alpha E\theta} = \frac{\alpha}{E\theta^{-1}+\alpha-1} \tag{3.7}$$

Substituting $\frac{p\lambda}{\beta+p\lambda}$ for $E[\theta]$ from Lemma 3.2.9 gives the result.

$$\boxed{X_\infty = \lim_{n\to\infty} E[X_n] = \frac{p\lambda\alpha}{\beta+p\lambda\alpha}} \tag{3.8}$$

□

Remark. Substituting for K_1 and K_2 in Equation (3.6) gives a closed-form solution for transient expertise levels as well:

$$\boxed{E[X_n] = \frac{p\lambda\alpha}{\beta+p\lambda\alpha}\cdot\left(1-\left[(1-\alpha)\cdot\left(\frac{p\lambda}{\beta+p\lambda}\right)\right]^n\right).} \tag{3.9}$$

To review, Equation (3.8) gives the asymptotic value of expertise when the arrival and service events are Poisson processes—the interarrival time between customers has a negative exponential distribution with rate $p\lambda$. We expect the routing proportion $0 \le p \le 1$ to be fixed and under management's control. Note that expertise tends towards the maximum value 1 when $p\lambda\alpha \gg \beta$.

3.2.3 Embedding Expertise Development in Queueing Models—Some Limitations

Contact centers employ a large workforce of agents. To use Equation (3.8) and (3.9) for such multi-agent systems, we must turn to a model like that on the left of Figure 3.4, where each agent is responsible for service to a separate M/M/1/∞ queue. Yet in practice contact centers assign multiple agents to the same queue to keep waiting times low. To model waiting times accurately, we

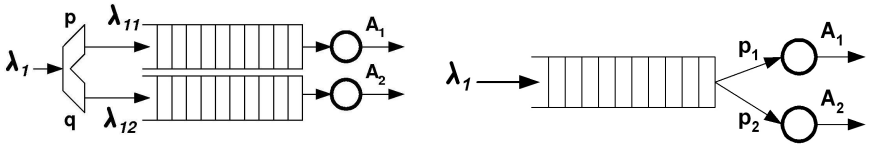


Figure 3.4: **A Queue Modeling Problem.** Left: an analytically tractable arrangement where each agent handles her own M/M/1 queue, used here in Chapter 3. Right: a more realistic model where agents share responsibility for a single queue. But because of the difficulty of finding analytical results for priority M/M/c systems, we turn to randomized simulations in Chapter 5 to investigate this model.

must turn to the M/M/c system on the right (Cleveland [2000], pages 89–90), or to related models such as M/G/c—in that case the general service time distribution G is often taken to be the lognormal distribution, following empirical results (Gans et al. [2003], page 127).

M/M/c systems are the most tractable of the family of multi-server queueing system models. But as with all the others, it is difficult to compute and write down expressions for steady state results where routing proportions p are different for every agent; further, the mean service time must remain stationary (Gross and Harris[1999], page 156). But in conflict with this requirement, in Chapter 4 we will see that mean service times may change with the experience our agents. For these reasons, we turn to randomized discrete-event simulations in Chapter 5 to generate results for different routing rules. In this light, the analytical results of Section 3.2 are best used to develop intuition—the concave increasing, bounded nature of the model follows the empirical data of Chapter 4 and other sources—and in Section 3.3 we will apply this intuition when designing utility functions in expertise levels of agents.

Note that forgetting occurs at the instant of departure, so in this model the degradation of expertise occurs even during the service time interval $1/\mu$—whereas expertise is expected to be improving by an amount $\alpha \cdot (1 - X_{n-1})$ during the service time. Thus it may be argued that forgetting should be modeled as only occurring when an agent is idle.

Unfortunately, forgetting based on idle times greatly complicates the derivation of an analytical result similar to Theorem 3.2.10. When interarrival times T_a and service times T_s are exponentially

distributed, the idle time is given by the difference function $g(T_a, T_s) = T_a - T_s$. But the distribution function of this difference is a mixed random variable, and the independent increments property fails to hold.

Empirical studies suggest that in a call center context, the duration of a call is in minutes; but the time it takes for the forgetting factor to significantly impact the process of expertise development is a matter of tens of hours, days, or even weeks. In that case—in empirical data we have observed—the forgetting rate β takes a very small value, and the forgetting process can reasonably be approximated by a constant process whether the agent is utilized or not. The service time duration is insignificant compared to the time needed for forgetting effects to be noticed the data. Therefore the current definition 3.2.4 for the forgetting factor is a very good approximation to a forgetting model based on idle times.

3.3. Expertise Utility Functions of the Customer and the Firm

3.3.1 The Customer's Utility U_c, and the Supervisor's Utility U_s

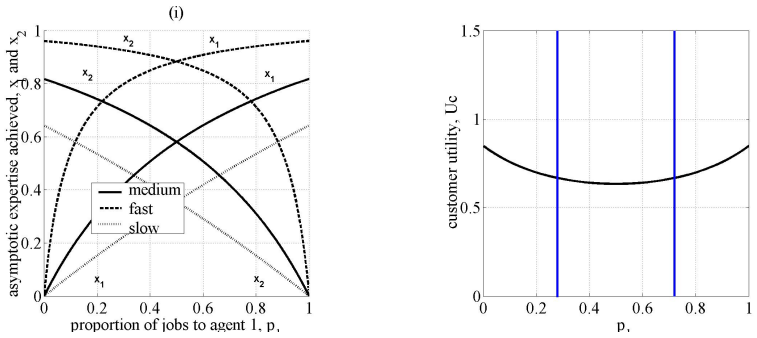


Figure 3.5: **Customer's Utility, Two Agents.** The customer's utility function U_c for the two agent case. Note the convex shape, with two optimal solutions residing at extreme values of the routing proportion p: $p = 0$, or $p = 1$. The vertical lines represent possible system constraints that limit the maximum utilization of an agent. Note that with the constraints, extreme points are still optimal, but the extreme values have been reduced.

The observation of a correlation between arrival rates and expertise leads to the natural question of how a call center should route calls to different agents. Consider the situation where all incom-

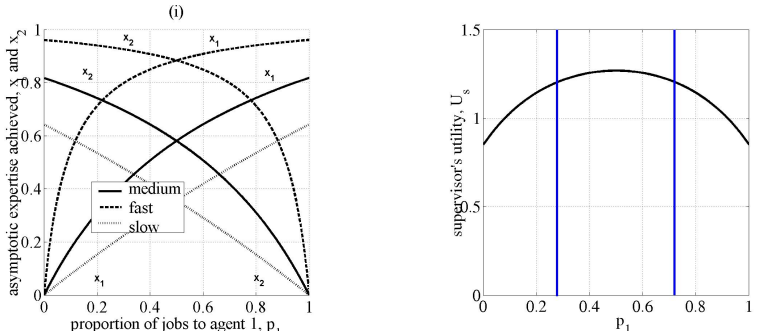


Figure 3.6: **Supervisor's Utility, Two Agents.** The supervisor's utility function U_s for the two agent case. Note the concave shape, with the optimal solution residing at the value where the routing proportions are equal: $p_1 = p_2 = 0.5$. The vertical lines represent possible system constraints that limit the maximum utilization of an agent. Note that with the constraints, the middle or even routing point is still optimal, and its value has been unaffected.

ing jobs are divided between two agents, A_1 and A_2. Take λ to be the arrival rate of all jobs into the system. Parameter p_1 is the fraction of jobs routed to A_1, and $(1 - p_1)$ is the fraction routed to A_2.

We see that the value of p_1 chosen by our decision rule thus determines the two asymptotic expertise levels of the agents—and we can introduce the notation $X_1(p_1)$ and $X_2(p_1)$ to denote the dependence of asymptotic expertise on p_1. Given this situation, we would like to know how one might select the ideal value for p_1. Here we consider the multiple M/M/1 system model of the left side of Figure 3.4, and asymptotic expertise given by Equation (3.8), page 46.

Customers and shift supervisors have different objectives with respect to knowledge of the agents. Customers may prefer to have the maximum available service expertise; we will call a utility function that maximizes this objective the *customer's utility*, or U_c. This is a function of agent expertise as determined by the routing policy, so we will indicate that dependence using the notation $U_c(p)$.

Management, particularly those in charge of shift staffing, are on the other hand also interested in the overall knowledge and expertise available within the company. For example, having more than one trained agent mitigates the risk of one agent leaving (and taking their expertise with them). We refer to a utility function that maximizes the experience available as the *supervisor's utility*, or $U_s(p)$. Having agents with similar knowledge level leads to quality assurance whereby each customer receives equivalent service, which might be desirable. This argument also favors $U_s(p)$.

Figures 3.5 and 3.6 illustrate the trade-off between customer and supervisor perspectives. We let the customer's utility be $U_c(p_1) = E[X]$, where $E[\cdot]$ denotes the expectation value; following the notation used in the figure, this is

$$U_c(p_1) = E[X] = p_1 X_1(p_1) + (1 - p_1) X_2(p_1) \tag{3.10}$$

Let the supervisor's utility be

$$U_s(p_1) = \sum_{i=1}^{I} x_i(p_i) = x_1(p_1) + x_2(p_1) \tag{3.11}$$

corresponding to the total knowledge of all the agents.

On the left of Figure 3.5 we see the asymptotic expertise attained by each of the two agents over the range of p_1. Here the forgetting rate for all pairs of curves is $\beta = 0.001$, and the learning rates from the top pair to the bottom pair are $\alpha = 0.011, 0.002$, and 0.0008—consider these *fast, medium,* and *slow* learning cases, respectively. The curve for the first agent grows with p_1, similar to the right side plot of Figure 3.3. As expected, the asymptotic expertise curve for agent 2 decreases in p_1. The right-hand plot shows the resulting customer's utility.

The left-hand side of Figure 3.6 repeats the two-agent expertise plot for reference, and the right-hand plot shows the supervisor's utility. Compare the plots of U_c and U_s, and note that the maximum of the supervisor's utility U_s is a minimum of the customer's utility U_c. Further, note that if the firm chooses solely to increase the utility function for the customer, it destroys its own cumulative expertise.

In Figures 3.5 and 3.6 blue lines indicate the effect of hypothetical system capacity constraints. That is, to keep customers from waiting too long, both agents have to pitch in and take some calls—the blue lines show the limit of what management will accept in the form of unbalanced routing assignments. The customer's utility U_c increases as the routing becomes unbalanced; **therefore, capacity constraints reduce the center's ability to pursue specialized routing and boost** U_c. This is a fundamental trade-off in the design of routing rules to develop specialized expertise.

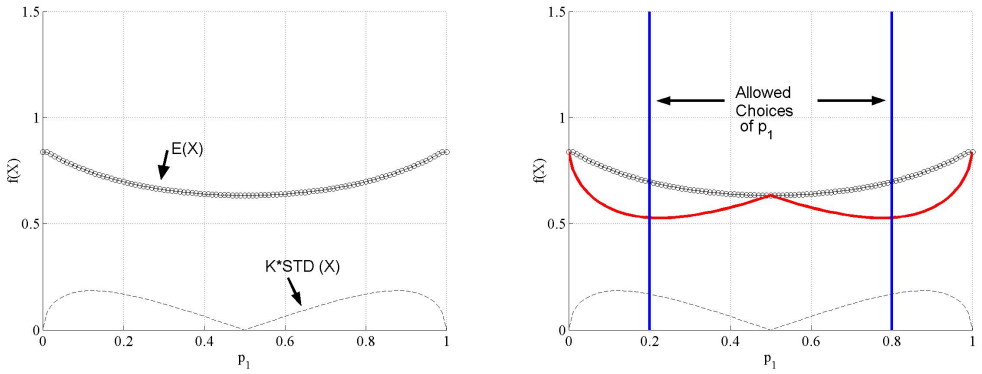


Figure 3.7: **A Utility Function Including Variance.** The brand manager's utility U_b rewards a high expected value of expertise $E[X]$, and penalizes variance in expertise. U_b is the solid red curve with two minima at the right.

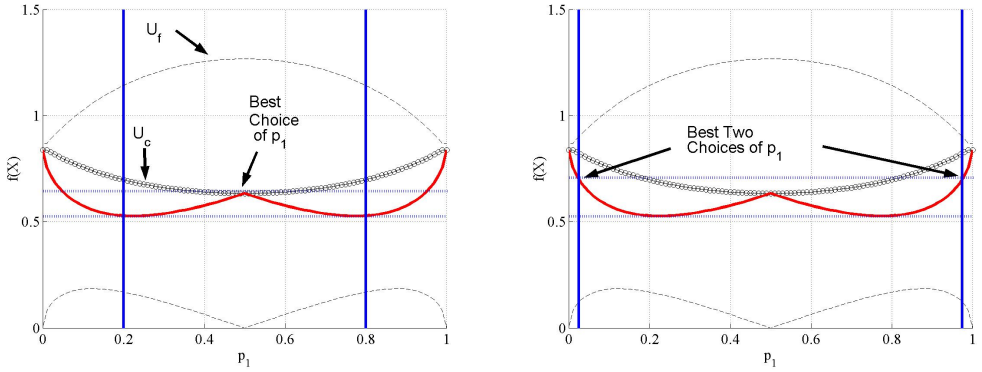


Figure 3.8: **Capacity Constraints Affect** U_b If capacity constraints are tight (left), U_b behaves like the supervisor's utility. If they are loose (left), U_b behaves like the customer's utility.

3.3.2 Incorporating Variance: the Brand Manager's Utility

A third utility function in asymptotic expertise that may be of interest to contact centers we describe as the *brand manager's utility*, or U_b. This is named for the person directly charged with maintaining the center's reputation for service quality. We make the simplifying assumption that the service quality level for a call is equivalent to the expertise level of the agent answering the call.

The brand manager wants both a high customer utility value—a high value of the expected

value of expertise—and a low value for the variance of that expertise. For our two-agent example:

$$\begin{aligned} E[X] &= p_1 X(p_1) + p_2 X(p_2), \;\; p_2 = 1 - p_1 \\ Var[X] &= \sum_{i=1}^{2} p_i \cdot (x_i(p_i) - E[X])^2 \qquad (3.12) \\ U_b(p_1) &= E[X] - K_b \cdot \sqrt{Var[X]} \qquad (3.13) \end{aligned}$$

Equation (3.12) is a standard expression for variance, with Bernoulli random variable p_i selecting one of two agents for each call in steady state, as shown on the left side of Figure 3.4. (This variance is different from the variance of the underlying Poisson stream of arrivals.) Equation (3.13) defines the brand manager's utility. To keep consistent units we take the square root of the variance, and multiply it by the manager's choice of a weighting factor, K_b.

Figures 3.7 and 3.8 illustrate the behavior of U_b for our two-agent example, and how system capacity constraints affect it; this case is more complicated than that of U_c and U_s. First, the left side of Figure 3.7 shows plots of $E[X]$ and $K_b \cdot \sqrt{Var[X]}$ separately as p_1 is swept from 0 to 1, with $K_b = 1$. The right side plots the resulting values of U_b as a dark red solid line, and adds two capacity constraints that limit the allowable choices of p_1.

The left side of Figure 3.8 shows that, with these same capacity constraints, the optimal choice of p_1 is 0.5—exactly in the center, at the point that optimizes U_s. But on the right side of Figure 3.8, we see that if the capacity constraints are loosened, the optimal value of p_1 becomes an extreme point—a solution that optimizes U_c. Thus the brand manager's utility may behave like U_c or U_s, depending on whether the capacity constraints are loose or tight.

3.4. Maximizing Utilities U_c and U_s in the Many-Agent Case

The following sections characterize policies for expertise development in multi-agent systems using convexity arguments. The main tools for proving results are the properties of strictly convex and strictly concave functions; see Boyd and Vandenberghe [2004], pages 70–71, or Bertsekas [2003],

pages 29–35 for a discussion of these properties.

3.4.1 Extreme Routing Optimizes the Customer's Utility

Consider a service system that is staffed by a number of agents, I, each having the same mean service rate μ. Let customers arrive to the system at a fixed rate, λ. Customers are all of the same type, and are not divided into priority classes—every customer receives the same treatment as all other customers with respect to service time and service quality, once he has entered service.

In this system, learning/forgetting curves for expertise development exist for every agent, as defined below. Each agent's curve, defining her potential to learn (improve expertise) and forget (reduce expertise) is the same as that of every other agent.

Definition 3.4.1. *Definition of the routing proportion p_i.*

For each agent i, let p_i be the proportion of customers routed to i, so that agent i handles an arrival rate of $p_i\lambda$. Let $0 \leq p_i \leq 1$, so that $\sum_{i=1}^{I} p_i = 1$. □

Definition 3.4.2. *Definition of agent expertise, $X(p_i)$.*

Let the asymptotic expertise value of X of each agent i be a concave function of the customer traffic $p_i\lambda$ that is routed to that agent, so we write $X(p_i\lambda)$. In our system λ is a constant value, while p_i may differ among agents. We write $X(p_i)$ to reflect the fact that asymptotic expertise varies among agents according to the independent variable p_i. Let $0 \leq X(p_i) \leq 1$, and assume the first and second derivatives of $X(p_i)$ with respect to p_i exist. [1] □

Theorem 3.4.3. *For the system described above, if $pX(p)$ is strictly convex in p, then the expected value of expertise seen by the customer in steady state, $E[X] = \sum_i^I p_i X(p_i)$, is always increased by shifting work from an agent with less expertise to an agent with more expertise.*

[1] A more descriptive notation for the quantity of expertise is $X(\alpha, \beta, p_i\lambda, \mu)$, indicating the dependence of X on four parameters: the learning rate parameter α; the forgetting rate parameter β; the mean arrival rate of customers to this agent, $p_i\lambda$; and the mean service rate μ. But in this system, α, β, and μ are fixed values that are the same for all agents, so we suppress them in the notation for expertise.

Proof. Let $g(p_i) = p_i \cdot X(p_i)$. Then the expected value of expertise seen by an arriving customer in this system's steady state, $E[X_s]$, is given by

$$E[X_s] = p_1 \cdot X(p_1) + p_2 \cdot X(p_2) + ... + p_n X(p_n). \tag{3.14}$$

$$= g(p_1) + ... + g(p_n) \tag{3.15}$$

Consider two agents i and j who receive nonzero routing proportions p_i and p_j, with $p_j \geq p_i$. Now, perturb the system to a new state using the routing rule to remove a small proportion of arrivals $\epsilon = \Delta p$ from agent i's assignment, and add those arrivals to agent j's assignment. The new expected value of expertise $E[X_{new}]$ becomes

$$E[X_{new}] = g(p_1) + ... + g(p_i - \epsilon) + g(p_j + \epsilon) + ... + p_n X(p_n) \tag{3.16}$$

Now we can analyze the difference $\Delta E = E[X_{new}] - E[X_s]$. If this difference is positive, the expected value of expertise increased due to the perturbation. We construct a first-order approximation to ΔE as follows:

$$g(p_i - \epsilon) \approx g(p_i) - \epsilon \cdot g'(p_i) \tag{3.17}$$

$$g(p_j + \epsilon) \approx g(p_j) + \epsilon \cdot g'(p_j) \tag{3.18}$$

$$\Delta E = E[X_{new}] - E[X_s] \approx \epsilon \cdot (g'(p_j) - g'(p_i)). \tag{3.19}$$

Note that $g(p) = pX(p)$ is strictly convex in p, so its first derivative is increasing in p. Therefore since $p_j > p_i$, we have that $g'(p_j) - g'(p_i) > 0$. The value of ΔE from (3.19) is positive: the expected value of expertise has increased, because we shifted work from agent i who has less expertise i to agent j who has more expertise. We improve $E[X]$ by giving more work to the busiest agent from any other agent who has nonzero work.

□

Corollary 3.4.4. *To maximize the expected value of expertise seen by a customer, $E[X]$, the optimal routing policy under the conditions of Theorem 3.4.3 is to send all work to a subset of fully utilized*

agents, and no work to the others. There may be one agent who is partially utilized, but never more than one.

Proof. By contradiction. Assume the optimal policy is to route work to two partially utilized agents. But then by Theorem 3.4.3, $E[X]$ will be improved by increasing one agent's workload until her utilization rate is 100%—culminating in one fully utilized agent, and another partially utilized or unutilized agent.

This property scales to larger systems of I agents: $E[X]$ improves as work is shifted to the busiest agent, until she is 100% utilized; $E[X]$ is improved again as work is shifted to the next busiest agent, until he is 100% utilized; and so on. □

3.4.2 Even Routing Optimizes The Supervisor's Utility

Here we use convexity arguments similar to those of the last section to show that even routing is optimal for the supervisor's utility, $U_s(p)$.

Theorem 3.4.5. *Consider again the system described in Section 3.4.1. Assume the asymptotic expertise of an agent X(p) is concave in p. Then the sum of asymptotic expertise in the system, S[X], is always increased by shifting work from an agent with more expertise to an agent with less expertise.*

Proof. The sum of asymptotic expertise present in the firm, $S[X]$, is given by

$$S[X] = X_1(p_1) + X_2(p_2) + ... + X_i(p_i) + ... + X_j(p_j) + ... + X_n(p_n) \tag{3.20}$$

Consider agents i and j who receive routing proportions p_i and p_j, with $p_j > p_i$. Perturb the system to a new state using a routing rule to remove a small proportion of arrivals $\epsilon = \Delta p$ from agent j's assignment, and add those to agent i's assignment. The new value of $S[X]$, or $S[X_{new}]$, becomes

$$S_{new}[X] = X_1(p_1) + X_2(p_2) + ... + X_i(p_i + \epsilon) + ... + X_j(p_j - \epsilon) + ... + X_n(p_n) \tag{3.21}$$

Now we can analyze the difference $\Delta S = S[X_{new}] - S[X]$. If this difference is positive, the sum of expertise in the firm increased due to the perturbation. We construct a first-order approximation to ΔS as follows:

$$X_i(p_i + \epsilon) \approx X_i(p_i) + \epsilon \cdot X_i'(p_i) \tag{3.22}$$

$$X_j(p_j - \epsilon) \approx X_j(p_j) - \epsilon \cdot X_j'(p_j) \tag{3.23}$$

$$\Delta S \approx \epsilon \cdot (X_i'(p_i) - X_j'(p_j)). \tag{3.24}$$

Note that $X_i(p_i)$ and $X_j(p_j)$ are strictly concave in p, so their first derivatives are decreasing in p. Here $p_i < p_j$, so $X_i'(p_i) > X_j'(p_j)$, and Equation (3.24) is positive due to the change in routing assignment ϵ. The sum of expertise $S[X]$ increases from shifting work from an agent with more expertise to an agent with less expertise. □

Corollary 3.4.6. *To maximize the sum of expertise over all agents in the system, $S[X]$, the optimal routing policy under the conditions of Theorem 3.4.5 is to share all work as evenly as possible among the agents.*

Proof. By contradiction. Suppose we claim the optimal policy for routing customers to agents is to route more to one of the agents, and fewer to a second agent. But by Theorem 3.4.5, $S[X]$ will improve if some work is shifted to the second agent. All agents share identical learning curves, so this fact is true for any two agents in the system; thus a policy of routing customers evenly to all workers maximizes $S[X]$. □

Remark. Corollary 3.4.6 does not hold for the more general case where agents learn at different rates. Still, over a variety of contact center simulations in Chapter 5, we find a policy of even routing usually achieves a high value for the metric $S[X]$.

3.4.3 Example Using a Specific Expertise Function

This section shows that the results of the last two sections apply to a function we developed previously to model asymptotic expertise levels in stochastic service systems. Recall the expression for the value of asymptotic expertise $X_\infty(p) = \frac{\alpha\lambda p}{\beta+\alpha\lambda p}$ given by Equation (3.8), where the arrival rate to an agent $p\lambda$ is a Poisson process, and the service rate μ is ignored in the forgetting calculation.

Lemma 3.4.7. *Let the value of asymptotic expertise be given by $X(p) = \frac{\alpha\lambda p}{\beta+\alpha\lambda p}$, from Equation (3.8). Then (i) the expected value of expertise seen by the customer in steady state, $E[X] = \sum_i^I p_i X(p_i)$, is always increased by shifting work from an agent with less expertise to an agent with more expertise. (ii) To maximize the expected value of expertise seen by a customer, $E[X]$, the optimal routing policy under the conditions of Theorem 3.4.3 is to send all work to a subset of fully utilized agents, and no work to the others. There may be one agent who is partially utilized, but never more than one.*

Proof. We show that the function $X(p) = \frac{\alpha\lambda p}{\beta+\alpha\lambda p}$ fulfills the requirements placed on the expertise function by the conditions of Theorem 3.4.3. The routing proportion p ranges between zero and one, $0 \le p \le 1$, so $0 \le X(p) \le 1$. Further, $X(p)$ has continuous first and second derivatives with respect to p :

$$\begin{aligned} X(p) &= \frac{\alpha\lambda p}{\beta+\alpha\lambda p} \\ X'(p) &= \frac{\alpha\lambda}{\beta+\alpha\lambda p} - \frac{(\alpha\lambda)^2 p}{(\beta+\alpha\lambda p)^2} \\ X''(p) &= \frac{-2\alpha\lambda^2}{(\beta+\alpha\lambda p)^2} + \frac{2p(\alpha\lambda)^3}{(\beta+\alpha\lambda p)^3}. \end{aligned}$$

Therefore $X(p)$ is an expertise function according to Definition (3.4.2).

It remains to show that $g(p) = pX(p)$ is a strictly convex function of p. Let p_1 and p_2 be two routing proportions, with $p_1 \neq p_2$. Use parameter ω, $0 \le \omega \le 1$, to form linear (convex) combinations of p_1 and p_2, and of $g(p_1)$ and $g(p_2)$; these combinations are written as $\omega \cdot p_1 + (1 -$

$\omega) \cdot p_2$ and $\omega \cdot g(p_1) + (1-\omega) \cdot g(p_2)$, respectively. Strict convexity occurs when

$$\begin{aligned} g\left(\omega \cdot p_1 + (1-\omega) \cdot p_2\right) &< \omega \cdot g(p_1) + (1-\omega) \cdot g(p_2) \\ 0 &< \omega \cdot g(p_1) \ + \ (1-\omega) \cdot g(p_2) \ - \ g\left(\omega \cdot p_1 + (1-\omega) \cdot p_2\right) \end{aligned} \tag{3.25}$$

After simplifying the notation by letting $v = \omega p_1 + (1-\omega)p_2$, we have:

$$\begin{aligned} g(v) &= \frac{\alpha\lambda v^2}{(\beta + \alpha\lambda v)} \\ \omega g(p_1) &= \frac{\omega\alpha\lambda p_1^2}{(\beta + \alpha\lambda p_1)} \\ (1-\omega)g(p_2) &= \frac{(1-\omega)\alpha\lambda p_2^2}{(\beta + \alpha\lambda p_2)}. \end{aligned}$$

Now we may use these three terms to build an expression that is the equivalent of the right side of Equation (3.25):

$$\omega g(p_1) + (1-\omega)g(p_2) - g(v) = \frac{\omega\alpha\lambda p_1^2}{(\beta + \alpha\lambda p_1)} + \frac{(1-\omega)\alpha\lambda p_2^2}{(\beta + \alpha\lambda p_2)} - \frac{\alpha\lambda v^2}{(\beta + \alpha\lambda v)}.$$

Strict convexity holds if this expression is positive for all $p_1 \neq p_2$. Note that the terms in the denominator are all positive. After expanding over a common denominator, the numerator becomes

$$\omega\alpha\lambda p_1^2(\beta + \alpha\lambda v)(\beta + \alpha\lambda p_2) + (1-\omega)\alpha\lambda p_2^2(\beta + \alpha\lambda v)(\beta + \alpha\lambda p_1) - \alpha\lambda v^2(\beta + \alpha\lambda p_1)(\beta + \alpha\lambda p_2).$$

After canceling like terms, this reduces to $\omega \cdot (1-\omega) \cdot \beta^2 \cdot (p_1 - p_2)^2$. Here all four quantities are positive; therefore $\omega \cdot (1-\omega) \cdot \beta^2 \cdot (p_1 - p_2)^2 > 0$ for all $p_1 \neq p_2$, and $pX(p)$ is strictly convex.

Now we have shown that $X(p) = \frac{\alpha\lambda p}{\beta + \alpha\lambda p}$ is an expertise function meeting the requirements of Theorem 3.4.3, and therefore property (i) holds. Property (ii) follows from Corollary 3.4.4.

□

Lemma 3.4.8. *Let the value of asymptotic expertise be given by $X(p) = \frac{\alpha\lambda p}{\beta + \alpha\lambda p}$, from Equation (3.8).*

(i) Then the sum of asymptotic expertise in the system, S[X], is always increased by shifting work from

an agent with more expertise to an agent with less expertise. (ii) To maximize the sum of expertise over all agents in the system, $S[X]$, the optimal routing policy under the conditions of Theorem 3.4.5 is to share all work as evenly as possible among the agents.

Proof. Following the proof of Lemma 3.4.7, the only requirement we have not yet demonstrated for this function is whether or not $X(p)$ is strictly concave. The test for strict concavity is a modification of Equation (3.25), for all values of p_1 and p_2 where $p_1 \neq p_2$:

$$X(v) - \omega \cdot X(p_1) \; - \; (1-\omega) \cdot X(p_2) \;\; > \;\; 0. \tag{3.26}$$

As before, $v = \omega \cdot p_1 \; + \; (1-\omega) \cdot p_2$. The three terms of interest are:

$$\begin{aligned} X(v) \;\; &= \;\; \frac{\alpha\lambda v}{(\beta + \alpha\lambda v)} \\ \omega X(p_1) &= \frac{\omega\alpha\lambda p_1}{(\beta + \alpha\lambda p_1)} \\ (1-\omega)X(p_2) &= \frac{(1-\omega)\alpha\lambda p_2}{(\beta + \alpha\lambda p_2)}. \end{aligned}$$

Arranging these in the form of the left-hand side of Equation 3.26 gives:

$$\frac{\alpha\lambda v}{(\beta + \alpha\lambda v)} - \frac{\omega\alpha\lambda p_1}{(\beta + \alpha\lambda p_1)} - \frac{(1-\omega)\alpha\lambda p_2}{(\beta + \alpha\lambda p_2)}$$

Again the terms in the denominator are positive, so we observe the sign of the numerator over a common denominator to determine concavity:

$$\alpha\cdot\lambda\cdot v\cdot(\beta+\alpha\lambda p_1)(\beta+\alpha\lambda p_2) \; - \; \alpha\cdot\lambda\cdot p_1\cdot((\beta+\alpha\lambda v)(\beta+\alpha\lambda p_2) \; - \; \alpha\cdot\lambda\cdot p_2\cdot((\beta+\alpha\lambda v)(\beta+\alpha\lambda p_1)$$

Evaluating this expression and cancelling like terms results in $\omega\cdot(1-\omega)\cdot\alpha\cdot\beta\cdot\lambda\cdot(p_1-p_2)^2$. All terms are positive, so

$$\omega \cdot (1-\omega) \cdot \alpha \cdot \beta \cdot \lambda \cdot (p_1 - p_2)^2 \;\; > \;\; 0. \tag{3.27}$$

Thus $X(p) = \frac{\alpha\lambda p}{\beta+\alpha\lambda p}$ from Equation (3.8) is strictly concave, and it meets the requirements for an expertise function in Theorem 3.4.5. Therefore, Theorem 3.4.5 holds, and claim (i) is satisfied. Claim (ii) follows from Corollary 3.4.6. □

3.5. Conclusions from the Asymptotic Expertise Model

In this chapter, we describe a method to quantify how task routing rules influence the long-run expertise of agents in a call center through on-the-job learning effects. We then prove how to obtain the optimal solutions for two conflicting objectives: the expected value of expertise seen by customers, or the *customer's objective*, and the sum of all expertise within the firm, or the *supervisor's objective*. When all agents in a population share the same potential to learn, a policy of extreme uneven routing optimizes the expected value, while a policy of even routing optimizes the sum. These routing-driven productivity trends guide our expectations when designing routing rules in Chapter 5, where we quantify trade-offs between knowledge management and waiting time goals over groups of agents and tasks.

CHAPTER

FOUR

Empirical Measurements of Agent Expertise

4.1. Empirical Research Goals

Here we analyze empirical performance data for agents in a financial service call center, and find evidence of on-the-job learning. As would be expected, learning occurs more clearly in a few call types, and in the earlier stages of an agent's career. We focus on a description of some of these observations. This data set is particularly valuable because it provides a rare opportunity to observe trends in both handle time (abbreviated HT, or just H), and first call resolution rate (abbreviated FCR, or just R). It has been suggested that a combination of these two metrics (the quantity μ-R = R/H) is

useful for implementing skill-based routing rules in call centers, and in this chapter we take advantage of this data to examine trends for the combined metric as well.

Performance on service quality metrics such as HT and FCR varies per call type and per agent. By observing service trends with respect to explanatory variables—such as the call type, the agent ID, the length of the agent's tenure, or his total cumulative production—we may generate models that predict future productivity.

Performance Metric	Expertise from Recent Experience	Expertise from Cumulative Experience
Call Handle Time H	**Chapter 2**	**Chapter 4** Empirical observations of faster call handle times (H) with increasing experience. **Chapter 5**
General Expertise X, First Call Resolution Rate R	**Chapter 3** (X)	**Chapter 4** Empirical observations of improving changes in first call resolution rates (R) with increasing experience.

We confirm the observation made about other industries that cumulative production may be used to define learning curves, such that we know who the best agents will be in the future: they are the ones who handle the most tasks. Then since management controls the agents' work assignments, management may shape the skills of its workforce over time through work assignment policies—Chapters 2, 3, and 5 evaluate these policy choices. In particular, the rates of expertise improvement seen among agents in Chapter 4 greatly influences our expertise optimization model in Chapter 5, both in terms of the learning model, and the optimization time period.

The table above shows how this study fits among the problems explored in other chapters. For more context see also the full thesis outline, the motivations behind each chapter, and the separate

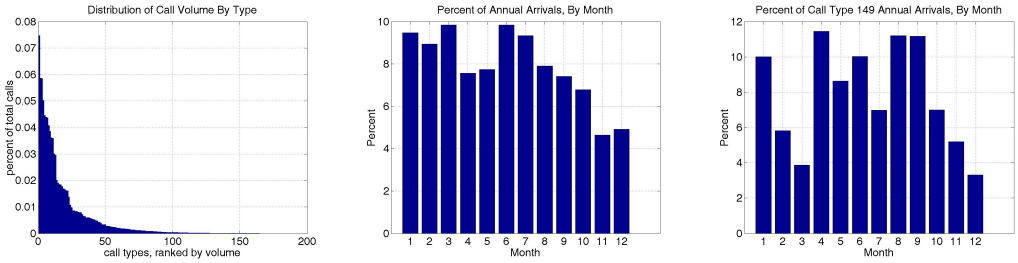


Figure 4.1: **Volume of Calls.** Left: volume of calls by call type, where types are numbered from 1 to 178. Most of the call volume is generated by the top 25 call types. Middle: aggregated annual traffic of all call types to the center as distributed by month. Right: Traffic of call type # 149 as distributed by month.

Call Type Identifier	Description	Call Volume, % of Total Sample
15	address change	7.5%
16	agent inquiry	5.9%
149	sales request inquiry	5.9%
152	securities transfer inquiry	5.0%
88	legal transfer instructions	4.5%
150	sales request taken	4.4%
154	shareholder confirmation	4.4%
127	price history inquiry	4.1%
61	duplicate investor id letter inquiry	3.9%

Table 4.1: **Examples of Call Types.** Large-volume call types in our sample. See also the right side of Figure 4.1.

chapter models that are described starting on page 5 in Chapter 1.

4.1.1 Contents of the Data Set

Our entire data set consists of call-by-call records of one contact center for all of 2007, and contains about 2.7 million calls. We present the following description of these records to help the reader understand the scope of the raw data. Each record has six fields:

- The name (individual agent ID number) of the agent who handled the call.

- The reason for the call— also the *call type*, or *task type*—which falls into one of 178 categories.
- The date and time of the call.
- The start date of the agent who handled the call; her tenure is computed as the difference between this field and the call date above.
- The time needed to handle the call, or *handle time* (HT), in seconds.
- The resolution status of the call—see below.

The resolution status may itself take one of six different values:

1. The customer's inquiry was resolved to a satisfactory level, such that the customer need not call again regarding this particular issue.
2. This call is the first of two or more transactions needed to satisfy the customer.
3. This is one of potentially multiple intermediate customer follow-up calls about this issue.
4. This is the last call out of a repeated series of customer calls regarding the same issue.
5. The calling party was not recognized as a customer.
6. "Other," a catch-all category for odd call types.

In our study, we define resolution status levels 1, 4, 5, and 6 to be successful from the point of view of service quality, while levels 2 and 3 are service failures. Thus the service quality record simplifies to a binary set, where a one in a call's resolution field is a success, and a zero is a failure.

Figure 4.1 gives some high-level call distribution data for this center. At left, the call volume by call type shows that most customer inquiries fall into the top 25 out of 178 categories. Table 4.1 also gives some details about the largest call types by total volume.

The middle bar graph of Figure 4.1 shows the arrival rate per month, as a percentage of the total annual call volume. Every month experienced substantial volume, though there was a decline at

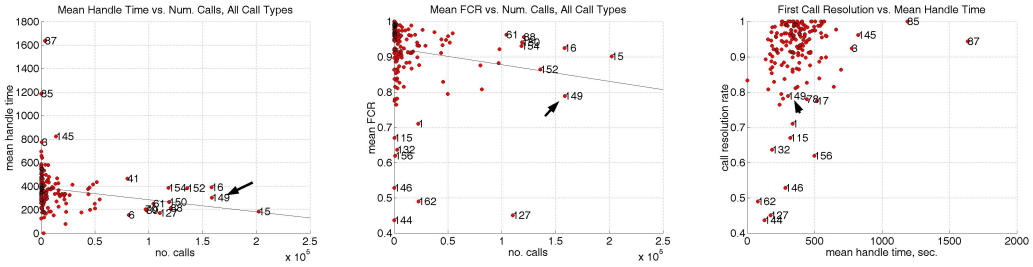


Figure 4.2: **Performance Trends by Call Type.** Left: scatterplot of HT versus call volume, where each point is a different call type, and the HT value is an average computed over all agents who took calls of that type. Middle: same as left, but for the FCR metric. Right: scatter-plot of FCR vs. HT. On all three plots, call type # 149 is indicated by an arrow (see Section 4.3).

the end of the year. The final bar graph at right shows the call volume per month of a specific call type, # 149, "sales request inquiry." (Note that a call type ID number has no relation to the total call volume of that type.) Later, Section 4.3 gives a detailed look at performance trends for call type # 149.

Among the various ways of classifying groups within this data, the call type proved to be the most significant; that is not surprising, because each call type defines a fundamentally different kind of task to be performed by agents. Thus our analyses in later sections generally start by segmenting the performance data into groups defined by call type.

Figure 4.2 provides scatter plots of performance metrics, averaged over all agents, broken down by call type. At left is handle time (HT) versus volume. A light linear regression trend line is shown on the data, showing a general trend of faster service with volume of calls. The middle plot shows the first call resolution rate (FCR) versus the volume of calls, with the trend line showing a slight worsening with volume. The right-hand plot shows FCR versus HT. Note that if successful efforts were made to improve performance through additional training, adjusting rosters, reassigning work, and so on, then those efforts will tend to move this cluster of points up and to the left.

The HT and FCR values form another useful quantity when combined together. Gans and Zhou (2003) showed that the product of resolution rate and service rate, or $\rho\mu$, is the best metric to use for assigning work to agents under certain conditions. Here μ measures how fast an agent works, and ρ measures the success rate in resolving calls. Higher values of this product indicate an agent both serves

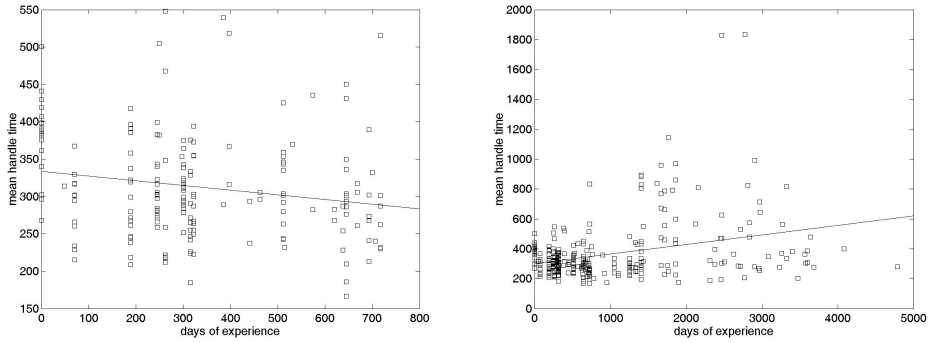


Figure 4.3: **Handle Time Trends with Tenure.** Left: average handle times, in seconds, for agents with up to two years' tenure in the call center show a slight decreasing trend. Each marker gives the value for one agent, averaged over all call types she handled. Right: the aggregate trend for all agents with up to 13 years' tenure shows an increasing trend. Note the large variation in mean HT starting at about three years out. The right-hand plot includes the data from the left-hand plot; the x-axis and y-axis scales on the right are larger.

customers more quickly, *and* reduces the number of incoming calls by providing correct service, thus avoiding repeated inquiries and lowering future traffic (and call center costs).

Here we refer to this metric as the μ-R product, or metric, and compute it as a mean FCR value divided by a mean HT value. A virtue of the μ-R product as a tool for managerial decisions is that it places an agent's speed of answer in a quality context. For example, some agents complete their customer encounters quickly, but fail to resolve problems—and basing judgments solely on HT data would reward such dysfunctional behavior. Sections 4.3 and 4.4 emphasize trends using the μ-R metric.

4.1.2 Plan of Experiments: *Data-Tenure*, and *Data-All*

While records for 1006 agents exist in our database, we only have the tenure start date for 310 of them. Further, Figure 4.3 shows that agents with greater than about three years' tenure exhibit a wider variation of mean handle time than newer agents. That variation may be driven by factors we cannot observe; our data set spans one year, so work assignments for previous years that could drive trends among veteran workers remain unknown to us.

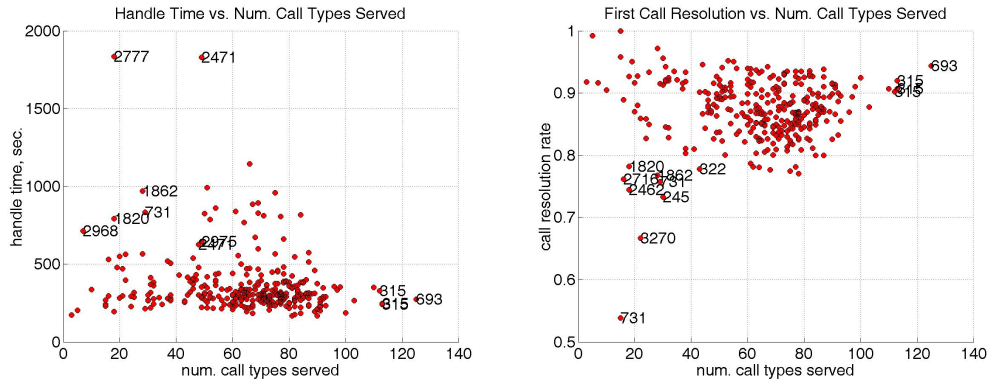


Figure 4.4: **Characteristics of Low and High Tenure Agents.** Scatter-plots of agents' average performance over all call types served: HT (left) and FCR (right). The data used is for all 310 agents for whom we have tenure records; tenure in days is plotted next to selected agents.

Figure 4.4 shows evidence for one possible explanation of the veteran agents' performance. These are scatter-plots of agents' average HT (left) and FCR (right) over all call types served. The data used is for all 310 agents for whom we have tenure records; tenure in days is plotted next to selected agents. Most agents of fall in the middle in terms of the number of call types served (40 to 100).

But a number of high-tenure agents serve a very small number of call types, and also demonstrate poor performance in terms of handle time and call resolution rate. That is consistent with a policy of making veteran agents specialists in certain call types, and giving those specialists the toughest calls from the most demanding customers. These veteran agents belong to a qualitatively different group and should be studied separately.

In response to these factors, we define data set *Data-Tenure* to be the subset of the main database containing all calls handled by agents with less than 15 months' tenure. In other contexts performance improvement is most pronounced among new workers, because newer workers are in the midst of ascending the steepest portion of their learning curves. Later sections examine *Data-Tenure* to test whether tenure and cumulative production have a significant relationship with agent performance.

Finally, define data set *Data-All* to be all records in the database. We describe some results using *Data-All* as well, in order to to try to boost the power of statistical tests, and see how results are

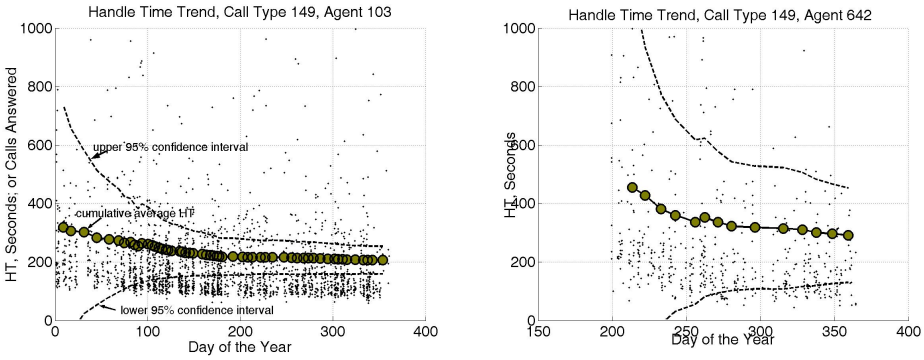


Figure 4.5: **Individual Handle Time Trends.** Left: One agent's handle time trend for a specific call type, showing improved performance (faster service) as cumulative production increases. Here $-\hat{b} = -0.04$. Right: A different agent's handle time trend. The rate of improvement is quite high—this agent reduced service time by almost 40% over three months, from an average of eight minutes per call to five minutes, after serving 560 calls. In this case $-\hat{b} = -0.07$.

influenced by larger sample sizes. Tests and results involving *Data-All* ignore the tenure field.

4.2. Performance Trends at the Level of Individual Agents

The left side of Figure 4.5 shows the handle time performance of Agent #103 on call type #149 over all the days in 2007. This comes from data set *Data-Tenure*. This agent begins work at the end of 2006, and so begins taking calls just before the series of measurement starts. The olive circles show the cumulative average HT over all the calls to that point—the trend with cumulative production. The converging dashed lines give 95% confidence intervals for the cumulative average HT.

Let N stand for cumulative production (the number of discrete tasks of this type done to date); and let $H(1)$ stand for the handle time of the first call. Then the traditional log-linear learning curve is given by the expression

$$H(N) = H(1) \cdot N^{-\hat{b}} \tag{4.1}$$

where exponent $\hat{b}$ is estimated from the data. For Agent #103, the best fit is $\hat{b} \approx 0.04$; and the learning curve follows the cumulative average HT trend line shown by the circles. This is a substantial trend in which the long-run average drops from about 300 seconds to 200 seconds per call over the first half of

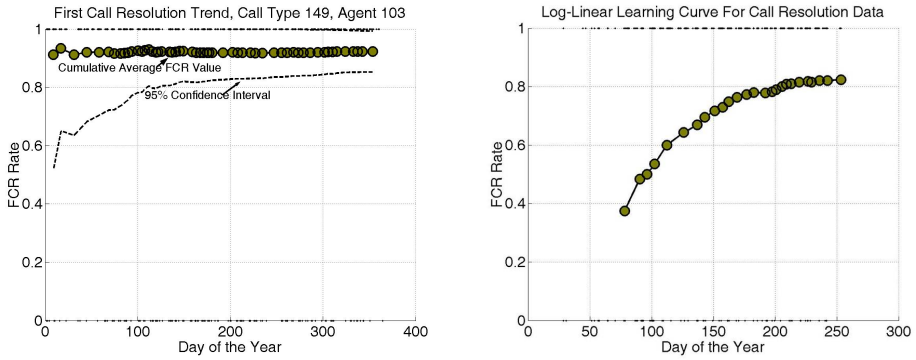


Figure 4.6: **Call Quality Trends.** Left: Agent #103's first call resolution (FCR) trend for call type #149. Here FCR performance does not change as cumulative production increases. Right: Some agents demonstrate on-the-job learning trends for call resolution. Here a different agent improves his cumulative average first call resolution rate from 40% to 80% after taking about 560 calls over six months.

the year. The right side of Figure 4.5 shows a different agent hired late in the year (Agent #642) who demonstrates a rapid learning rate, $\hat{b} \approx -0.07$.[1]

Note that in contrast with data from manufacturing settings, there is considerable variance among the call time values due to the inherent variability of customer service demands. For individual learning curve estimates, we only recorded $\hat{b}$ values for agents who handled 120 or more calls of the type in question.

Figure 4.6 gives the same agent's performance history for the FCR metric. One means a call is resolved successfully, and zero means the customer's issue is unresolved and he must call back. By contrast with the HT curve, this cumulative average FCR curve is flat, and no significant learning curve appears. For many agents, changes in the mean resolution rate with cumulative production are less significant than changes in the mean handle time. Even when the mean resolution rate is a flat, fixed value, that value is different from agent to agent, and agents may still be differentiated by their FCR performance. Some agents do demonstrate learning trends with respect to call resolution—the agent on the right of Figure 4.6 doubled productivity on this measure after serving about 560 calls.

[1] The estimation routine finds the best least-squares fit of a log-linear trend line, characterized by $H(1)$ and b. Before fitting, the call data is averaged such that one data point submitted to the estimator represents 40 calls. Those calls may be spread out in time—the averaging operation is done with respect to volume, not time.

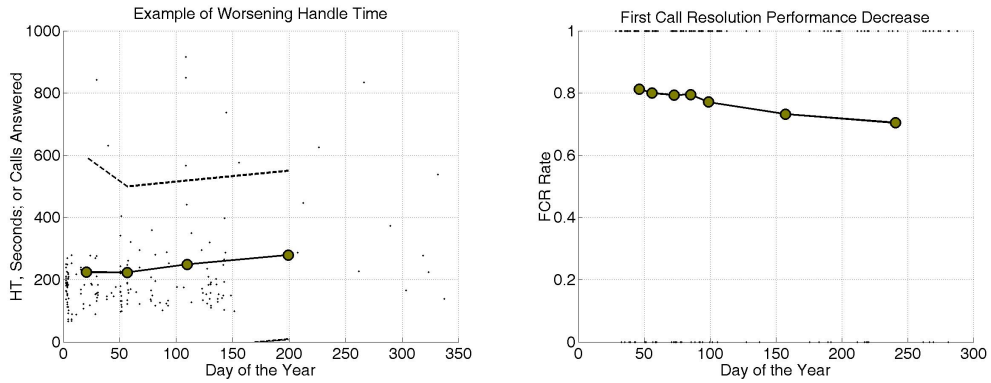


Figure 4.7: **Example of Performance Getting Worse.** Left and right plots are examples of two different agents who are assigned only a few calls of call type #149–most of their work involves handling other kinds of calls. As their production rate falls, their cumulative average HT and FCR values get worse.

Our review of this contact center's procedures leads us to assume that the difficulty level of a particular call is not screened before the call is routed to an agent.[2] All agents have the same chance of facing an unusual inquiry that is very difficult to handle in one encounter; when comparing work histories spanning hundreds or thousands of calls of a specific type, differences observed among FCR rates may thus be fairly attributed to the background, training, and talents of the individual agents.

As an interesting side note, we have investigated whether FCR gets worse when agents are highly utilized; in general, no significant correlation between peak busy periods and poor FCR values exists among these agents. On a typical day these agents as a group are 32% utilized, based on an 8-hour shift. The maximum utilization is about 86%, but in the record the mean utilization for agents in the *Data-Tenure* set rises above 50% for only 18 days out of the year.

The trends in the plots of HT and FCR repeat in data set *Data-Tenure* among the other agents as well. Exceptions occur when only a few calls appear in an agent's record; then the small sample size with high variance gives an erratic trend for the mean, and larger confidence intervals.

As an example, Figure 4.7 gives the performance history for an agent whose work assignment changes mid-year, and who takes less than 40 calls of type #149 for the last half of 2007—most of

[2]One exception to this is the small set of returning calls to the most veteran agents, who are routed the most unusual or specialized cases. When we are aware of them, we exclude these cases from our analysis.

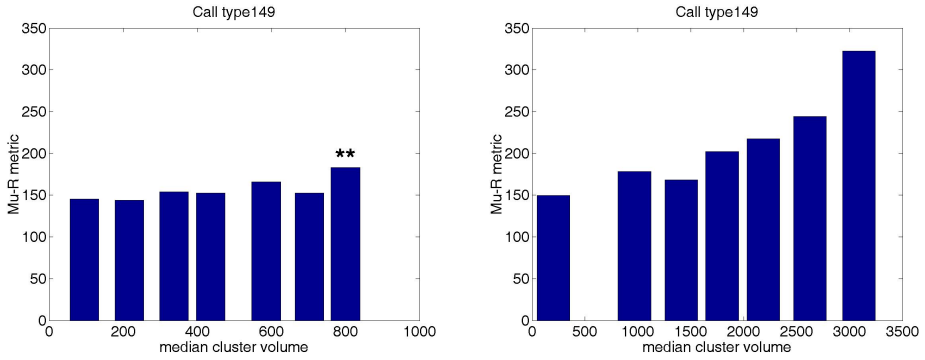


Figure 4.8: **Clusters Assigned by Call Volume, Plotting μ-R Metric, for Call Type *Sales Requests*.** From data set *Data-All*: performance on the μ-R metric for call type #149. Left: to obtain even cluster sizes, agents who served more than 800 calls join the last cluster (indicated by **). Right: cluster sizes are uncontrolled. Single agents are allowed to form clusters, so the trend among very high-volume agents influences the general trend. The best agent achieves over twice the μ-R score of the lowest-volume cluster.

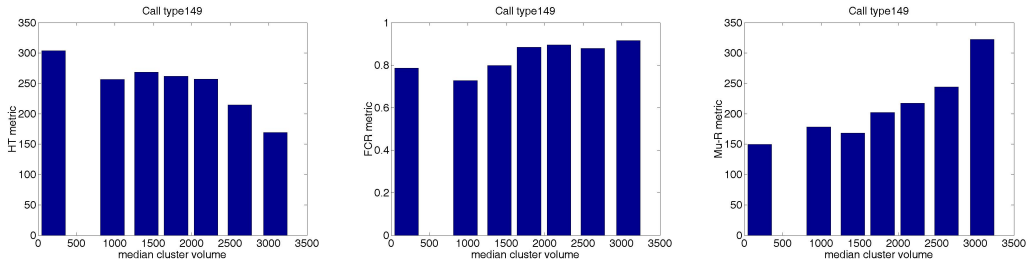


Figure 4.9: **Clusters Assigned by Call Volume, Plotting Handle Time and Call Resolution, for Call Type *Sales Requests*.** From data set *Data-All*, for call type #149. Left: the median call handle time decreases with the volume of calls handled. Middle: the FCR rate is high throughout, but does show improvement in later clusters. Right: again for reference, the μ-R metric improves with volume.

his time is then spent on other call types. This agent's production rate slows down over time, and his performance levels show evidence of worsening over time, making him a potential subject for a forgetting rate study. For now, we just indicate that possibility for future work, and maintain focus on trends of increasing productivity in the data.

4.3. Performance Trends at the Level of a Single Call Type

Call type #149, or "sales request inquiries," exhibits significant cumulative production-based performance trends. Section 4.2 takes examples of agents serving this call type to illustrate productivity changes at the individual level. Here we broaden the scope of our analysis: we seek to know how significant production volume trends are among all agents serving this type.

As the first step, consider the following approach. Given a specific call type Y found in data set *Data-Tenure*, consider all the agents who served that type during the year. Put those agents into a list, and sort them by the number of type Y calls they took. Those at the bottom of the list served tens or hundreds of type Y calls, while those at the top served thousands. May we expect a significant performance difference between those at the bottom of the list, and those at the top?

To test call type #149, we split the list into two agent groups of equal size, low-volume and high-volume, and compare their median μ-R values (computed as mean HT divided by mean FCR for each agent and call type). Using a standard t-test and a Wilcoxan rank test, we find trends with respect to volume that are significant at the 95% level. The high-low group difference for median μ-R values is about 25%, and for median handle times is about a minute per call.

Some agents in this contact center are in the active portion of their learning curves—consider them *active learners*—and some have already traversed their learning curves with respect to cumulative production: they have reached a performance plateau. For *Data-Tenure*, call type #149, the mean estimated learning exponent for handle time is $\hat{b} = .018$, with standard deviation $\hat{\sigma} = 0.033$. 70% of these agents were active learners, and 54% had a learning exponent more active than the mean, with $\hat{b} > 0.018$. Note that values of $\hat{b}$ close to zero indicate a plateau, while larger values indicate active improvement—the larger, the more active. For instance, an agent for which $\hat{b} = .018$ will demonstrate a mean handle time that is 11% faster after serving 500 customers of this type.

Figure 4.8 illustrates another approach for observing performance trends with respect to volume. We use a standard, tree-based clustering algorithm to group agents into clusters based on

the volume of calls they take. The effect is to reduce the natural variation that exists in the customer encounter data, and expose underlying trends.

Here the left-hand bar graph gives the median μ-R value for each of seven clusters, with the median volume of each cluster increasing from left to right on the abscissa. Comparing the first and last clusters, we see the median μ-R value has increased about 20%—see Table 4.2, Part (I).

In this plot, size limits keep the clusters from becoming too small. That forces the last cluster (marked by two stars, **) to accept several elite agents, who handled extremely large numbers of calls of type #149. Note that we are measuring median values, so even if the top ten elite agents are removed, we still see the same 20% increase in μ-R levels.

For the right-side bar graph, the clustering algorithm runs with no restrictions on cluster size. Now it defines clusters for the elite agents, even though some clusters only hold one agent. The new trend is striking: *the μ-R value more than doubles when the best cluster is compared to the lowest-volume cluster*.

We note the following evidence of the relationship between call volume and μ-R performance level for type #149:

- The Wilcoxan rank test shows that the higher-volume half of the agent population performs better.
- The size-restricted clustering results show a 20% improvement between the first and last clusters.
- The unrestricted clustering results show a performance doubling between the first and last clusters.

We also see evidence at the individual level of agents improving their operational performance by handling more calls. Agents 103 and 642 from Section 4.2 demonstrate improvement with increased production. In the unrestricted clustering results, agent 103 is in the third-highest cluster; and agent 642 after only three month's service moved from the bottom 25% to the top 25% of agents by μ-R level.

(I.) Call Type #149 Volume-Based Clusters, Cluster Size Restricted. Data Set *Data-All*				
# Agents	Median Tenure	Median Volume	Median μ-R Metric	Median HT
174	—	95	.15	328
82	—	217	.14	313
43	—	337	.15	298
55	—	436	.15	294
25	—	586	.17	295
14	—	703	.15	290
45	—	800	.18	259
(II.) Call Type #149 Volume-Based Clusters, Unrestricted. Data Set *Data-All*				
402	—	201	.15	304
19	—	967	.18	257
12	—	1409	.17	268
1	—	1795	.20	262
2	—	2184	.22	257
1	—	2626	.24	215
1	—	3087	.32	169
(III.) Call Type #149 Volume-Based Clusters, Unrestricted. Data Set *Data-Tenure*				
42	318	154	0.12	379
24	292	364	0.16	321
23	261	612	0.14	312
1	208	1013	0.14	363
2	314	1534	0.15	327
1	362	1786	0.25	214
2	235	2183	0.26	221

Table 4.2: **Characteristics of Agent Clusters That Are Assigned Based on Call Volume, for Call Type *Sales Requests*.** (I.) Assignment of agents into seven clusters for call type # 149, for Figure 4.8. The clustering criterion is call volume. Note that the μ-R metric tends to get better with cumulative production for the very high volume clusters—and we can see that a decreasing handle time is negatively correlated with increasing μ-R levels. (II.) Size is unrestricted—the algorithm may form individual clusters for elite agents. (III.) Clusters are recomputed using *Data-Tenure*, showing no clear trend with respect to agent start dates.

4.4. Performance Trends at the System-Wide Level

For almost every call type, some individual agents demonstrate increased productivity as a function of cumulative production. Repeating the high volume-low volume group segmentation of the previous section for the other call types, we find that about 40% of all customer requests belong to a call type for which performance is better—statistically significant at the 95% level—for high-volume agents than for low-volume agents.[3]

We would also like to know if cumulative production or simple tenure better predicts performance changes in this setting. A repeat of the Wilcoxan rank test for tenure shows 15% of all the customer requests belong to call types for which tenure predicts improving performance at the 95% level of significance.

This section goes further and applies our clustering method to other call types as well. Agents serving each type join one of seven groups based on volume. In separate trials, they join based on their tenure. After the clusters are formed, the median μ-R values from each group are compared with the clustering variable to see how well they are correlated. No limits exist for cluster sizes, so single-agent clusters are allowed. We find that about 30% of the calls fall into categories that resemble type #149, where μ-R performance levels are well predicted by volume-based clusters. Only three call types demonstrate tenure-based clusters that predict μ-R well.

Table 4.3 shows the cluster analysis results, for clusters whose correlation and $R2$ values we consider significant. Part (A) lists results for volume-based clustering using data set *Data-Tenure*. Part (B) shows the results for tenure-based clustering. Part (C) repeats (A) using the larger data set, *Data-All*.

We apply the following five metrics, all of which indicate higher correlation or quality the closer they approach one.

- CC is the *cophenetic correlation coefficient*, which measures the internal consistency of the

[3]Some call types with very low arrival rates were excluded from this analysis. For those types, so few calls were handled that the median values of low and high volume agent groups were almost the same.

clusters.

- *Kendall's tau correlation coefficient* is a unitless, nonparametric estimator of the trend among clusters for μ-R to improve with volume, Parts (A) and (C), or tenure, Part (B). Section 4.A.2 provides more details.

- *R2, volume* is a metric from a linear least-squares regression, showing how well volume predicts the μ-R trend among clusters.

- *R2, tenure* is similar, but uses tenure as the predictor instead of volume.

- *R2, Vol.+Ten* uses both volume and tenure to predict μ-R.

4.5. Conclusions from the Empirical Study

In this chapter we have analyzed a data set of almost three million calls handled over the course of a year by one thousand agents. Those calls are classified into 178 distinct types. For many call types, there is at least one example of a learning curve by which an agent improves his handle time performance, or his call resolution performance, as a function of his cumulative average production of that type. Over the set of individual agent learning curves, regardless of call type, we estimated the fastest learning rate in Equation (4.1) to be $b = 0.1$.

At a higher level, we see that increased cumulative production can drive performance across large groups of agents. For several call types there was a statistically significant improvement in performance for agents who took a high volume of that type, compared to agents who only took a few calls of that type. A cluster analysis of one of these types shows that the μ-R metric doubled when we compare the most experienced worker to the median of the cluster of the least experienced workers. In Chapter 5, we draw upon these observations to set the parameters in a two-step, optimization–simulation approach for finding the best work assignments to groups of agents.

(A) Volume-Based Clusters, Data Set *Data-Tenure*						
Task ID	# Agents	CC	Kendall's Tau	R2, Volume	R2, Tenure	R2, Vol.+Ten.
39	95	0.80	0.71	0.65	0.81	0.83
61*	97	0.83	-0.71	0.64	0.34	0.86
70	82	0.75	0.81	0.65	0.19	0.76
127	78	0.80	0.62	0.82	0.06	0.85
145	16	0.97	0.81	0.92	0.24	0.93
149	95	0.93	0.62	0.70	0.02	0.73
150	77	0.90	0.71	0.73	0.08	0.75
166*	29	0.95	-0.52	0.61	0.36	0.68
174	52	0.81	0.81	0.76	0.10	0.77
(B) Tenure-Based Clusters, Data Set *Data-Tenure*						
Task ID	# Agents	CC	Kendall's Tau	R2, Volume	R2, Tenure	R2, Vol.+Ten.
49*	22	0.86	-0.81	0.05	0.66	0.73
61*	97	0.81	-0.71	0.66	0.78	0.92
153	16	0.89	0.71	0.06	0.72	0.75
(C) Volume-Based Clusters, Data Set *Data-All*						
Task ID	# Agents	CC	Kendall's Tau	R2, Volume	—	—
14	212	0.89	0.73	0.84	—	—
70	299	0.78	0.52	0.50	—	—
88	315	0.88	0.81	0.87	—	—
127	347	0.93	0.81	0.83	—	—
149	438	0.91	0.90	0.84	—	—
150	284	0.89	0.81	0.57	—	—
152	331	0.87	0.81	0.74	—	—
174	158	0.86	0.81	0.71	—	—
178	179	0.78	0.62	0.78	—	—

Table 4.3: **Characteristics of Agent Clusters That Are Assigned Based on Call Volume, for Selected Call Types.** Clustering trends that predict the μ-R metric. There are no restrictions on cluster sizes. Section 4.4 provides definitions of the column headings. Agents serving each call type in this table were divided into seven clusters, and cluster median values were compared. These call types demonstrate statistically significant performance differences from one cluster of agents to the next cluster, where agents are assigned to clusters based on how many calls of that type they took during 2007—*volume-based clusters*. (*) Call types 49, 61, and 166 are interesting because of their high negative correlation with μ-R performance. For type 61, the reason is that FCR rates fall with increasing tenure and volume.

4.A. Clustering and Ranking Methods*

4.A.1 An Algorithm for Forming Agent Clusters*

The clustering results discussed in Chapter 4 are obtained using the following three-step procedure.

First, we compute the Euclidian distance between each pair of observations in the set. In these experiments the distance would be in units of time, for tenure-based clusters, or in units of customers served, for volume-based clusters.

Next, the procedure builds a hierarchical cluster tree. All agent records start as leaf nodes, and they join clusters in such a way that the increase (due to the join) in the sum of the squares of the distances within a cluster is minimized. Small clusters are then joined to form larger ones, until a predefined number of clusters is achieved—here, between seven and ten. To evaluate whether or not to join small clusters a and b, we note the membership size of each, n_a and n_b, and then compute $n_a \cdot n_b \cdot d^2/(n_a + n_b)$, where d is the distance between the centroids of the two clusters.

Third, we compute the Cophenetic clustering coefficient, which measures how well the hierarchical tree preserves the original distances between the observations: a value close to one means the clustering solution represents the original structure of the data well (Martinez and Martinez [2008], page 436).

4.A.2 A Note on *Kendall's Tau*, A Nonparametric Ranking Comparison of Tenure and Volume Effects*

Preliminary regression tests indicate that volume is a better predictor than tenure, but it is difficult to directly compare the effects of tenure and cumulative production because of differences in the units of the regression coefficients. As a means of validation, we may use Kendall's tau coefficient τ to provides a unitless comparison of these trends (Higgins [2004], page 159). Consider a two-column, n-row matrix of pairs of observations (X_i, Y_i) that we assume have been drawn randomly

from a continuous bivariate population—in this case, X_i will represent tenure in weeks or the call volume, and Y_i will be a performance measure, such as handle time. In the tests implemented here, these values are positive integers. Then, sort the rows in ascending order of the X-values, keeping pairs of observations together. We define the rank correlation coefficient τ as follows:

$$\tau = 2 \cdot P[(X_i - X_j)(Y_i - Y_j) > 0] - 1.$$

So τ is a function of the probability P that the product $(X_i - X_j)(Y_i - Y_j)$ is greater than zero, where index j is a row further down than i. τ takes values from 1 to -1; if $P = 1/2$, then $\tau = 0$.

We can compute an estimate of τ as follows:

$$U_{ij} = \begin{cases} 1, & (X_i - X_j)(Y_i - Y_j) > 0 \\ 0, & (X_i - X_j)(Y_i - Y_j) < 0 \end{cases} \tag{4.2}$$

Then define $V_i = \sum_{j=i+1}^{n} U_{ij}$, so that V_i is the number of pairs (X_j, Y_j) that are concordant with (X_i, Y_i) for $j \geq i + 1$. Then $\binom{n}{2}$ is the number of pairs in the data set, and

$$V_i = \frac{\sum_{i=1}^{n-1} V_i}{\binom{n}{2}}$$

is the fraction of concordant pairs. Then the estimate of the expected value of τ becomes

$$\hat{\tau} = 2 \cdot \frac{\sum_{i=1}^{n-1} V_i}{\binom{n}{2}} - 1.$$

The variance is

$$\mathrm{Var}[\hat{\tau}] = \frac{4n + 10}{9(n^2 - n)}.$$

So the estimation process for $\hat{\tau}$ is straightforward, though some technical corrections are needed when ties are present among the data.

4.A.3 The Wilcoxon Rank-Sum Test*

Another non-parametric test used in Chapter 4 is the Wilcoxon rank-sum test, also known as the Mann-Whitney U test (Higgins [2004], page 37). We compare two agent groups to test the

hypothesis that they both come from the same population—in this case, whether agents X_i that handle low call volumes achieve the same performance as those Y_j who handle high volumes of a certain type. Our results is that the median of the low-volume group is significantly different from the high-volume group at the 95% level.

For groups such as ours with more than 20 members, the test is computed simply as follows. Define the Mann-Whitney statistic U as the number of pairs (X_i, Y_j) for which $X_i < Y_j$. A large value of U means the larger rank values—say, ranked in increasing order by μ-R value—occur among the high-volume Y_j agents. Then consult a lookup table for U that defines the critical values at the desired significance.

Tables of the standard normal may be used for large sample sizes, such as ours. In this case $Z = \frac{U-\mu_u}{\sigma_u}$, where $\mu_u = \frac{N_X \cdot N_Y}{2}$, and $\sigma_u = \sqrt{N_X \cdot N_Y \cdot (N_X + N_Y + 1)/12}$; N_X is the number of agents in the low-volume group, and N_Y is the number in the high-volume group.

The derivation of the statistic U is similar to that of Kendall's τ, and the two statistics are equivalent in specific cases.

CHAPTER

FIVE

Planning and Simulation of Expertise Development

5.1. Introduction: An Optimization/Simulation Approach

5.1.1 Motivation and Models

Suppose an operations manager determines that on-the-job learning is significant among the workforce of agents she is responsible for. She observes her agents developing expertise with additional exposure to specific calls, exhibiting trends such as those of Figure 5.1 . How may she best take advantage of those trends to improve her group's performance? In this chapter, we provide a two-part answer to this question. This model assumes a call center with agent utilization rates between 30% and 60%, belonging to the *quality-driven regime* (see Gans et al. [2003], page 100).

As the table below shows, Chapter 5 focuses on improvements in call handle time with

cumulative expertise, where handle time is assumed to start slow and to get faster with experience. We have several reasons for selecting the handle time–cumulative experience model here. First, note that trends of handle time getting faster with experience were common among the agents and call types observed in Chapter 4. From the data, we saw that forgetting may be a factor for some agents, but that forgetting trends were not reliably identified for larger groups. By contrast, trends of learning with cumulative production were significant on a wide scale. Improvements in handle time may also have several interpretations, making them useful in different contexts. For instance, they may be taken to be improvements in service quality—a fast response may indicate a knowledgeable agent. They also have a noticeable operational impact because they increase the call center's capacity, and lower customer waiting times, which can be measured in queueing simulations. For more context on our modeling choices, see also the full thesis outline, the motivations behind each chapter, and the separate chapter models that are described starting on page 5 in Chapter 1.

Part one, starting at Section 5.2, discusses mathematical programming approaches for generating optimal work assignments. These assignments are in the form of call routing targets for each agent that depend on their skill and expected learning curve in cumulative production for each call type. To be useful, the routing targets need accurate assessments of agents' potential to improve—derived for instance from human resources data about an agent's education, background, and training period observations—and an accurate demand forecast for incoming calls. We will assume sufficient accuracy on both measures for the purpose of this chapter.

Performance Metric	Expertise from Recent Experience	Expertise from Cumulative Experience
Call Handle Time H	**Chapter 2**	**Chapter 4,** **Chapter 5** Design and simulation of optimal work assignments under learning, adapting Ch. 3 utility functions to evaluate improvements in call handle times (H) over groups of agents.
General Expertise X, First Call Resolution Rate R	**Chapter 3** (X)	**Chapter 4** (R)

Part two, starting at Section 5.3, describes routing rules that implement the routing targets in a stochastic model of a small contact center, where customer arrival times and service times are randomly distributed. In the normal case of multiple agents serving multiple queues, the observed

inherent randomness of these systems will cause agents to be ahead of schedule with respect to some routing targets, and behind with respect to others, at any single point in time. Recognizing this, we define five routing rules that control agent activity with respect to their pre-set targets in different ways. We then conduct experiments using discrete-event simulations to evaluate the rules along dimensions of expertise development, variation of expertise among agents, and maintenance of waiting-time performance.

Informed by the behavior of expertise objectives from Chapter 3, we observe how the five rules must trade off customer waiting time against fidelity to routing targets. Among our simulation results, we describe a case under which the customer's utility function U_c may be substantially improved using priority routing, and another using the same system parameters under which priority routing is counterproductive. Taken all together, parts one and two provide a contact center manager with a toolkit to set both routing targets and target-implementing rules such that the development of agents' expertise can be fully incorporated in operational planning.

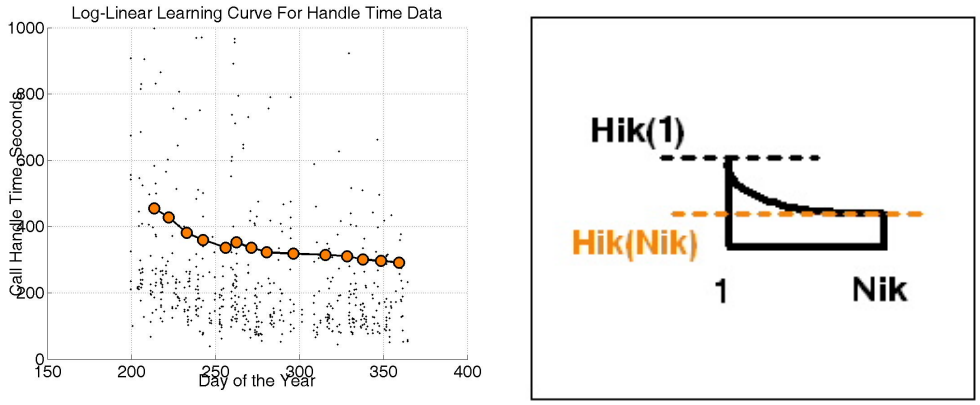


Figure 5.1: **Recap: Empirical Examples of Improving Performance.** Repeated again from Chapter 1: an example of cumulative average handle time performance improving with experience. Left: an agent gets faster and decreases her average handle time from 8 minutes to 5 minutes, a $\approx 40\%$ improvement, after handling about 560 calls over five months. Right: our model of this process, the log-linear learning curve, where the call handle time for agent i and call type k is a decreasing function of cumulative production, $H_{ik}(N_{ik}) = H_{ik}(1) \cdot N_{ik}^{-b_{ik}}$. The learning exponent in this case is $b = 0.07$.

5.2. A Nonlinear Program For Finding Optimal Work Assignments

5.2.1 Design of the Objective Function

Suppose our operations manager is responsible for a total of I agents who handle K different call types. The handle time performance changes by agent and by type, so let subscripts i and k be the indices specifying each agent and type, respectively. More customers typically call in than a single agent can handle, so she seeks to divide N_{total} calls into work assignments N_{ik} for each {agent, type} pair:

$$N_{total} \quad = \quad \sum_{k=1}^{K} N_k = \sum_{i=1}^{I} \sum_{k=1}^{K} N_{ik}. \tag{5.1}$$

Her first step is to define an objective function for the optimization. She notices that her agents perform better with experience, and thus wishes to consider learning-based improvements in efficiency as part of her work allocation model. In Chapter 3, we examined two fundamental functions of expertise that she can use.

- The first is the the expected value of agent expertise seen by a customer, which we call the *customer's utility function of expertise*, or U_c. This function is optimized by an extreme routing rule that concentrates the highest possible volume of work on the smallest feasible subset of agents.

- The second is the sum of expertise present at the call center, which we call the *supervisor's utility function of expertise*, or U_s. This function is optimized by a routing rule that divides work assignments evenly among agents.

In the approach taken here, one of these utility functions becomes the optimization objective, and its influence is represented in the optimization solver output by a set of target percentages for routing calls to agents. Subsequent simulation results show us which of a selection of routing rules characterized by

high to low evenness among call distribution outcomes does best to implement the objective's target values in a stochastic model of the service system.

The agents' circumstances will favor one of these utilities. If the arriving tasks favor specialized expertise—expert agents demonstrate critical efficiency and accuracy improvements compared to unseasoned agents—and if on-the-job learning is important for manifesting that expertise, then U_c is the better choice. Some agents may have strong latent task-specific potential that benefits the group if it can be tapped; specialized routing targets to steer the right calls to those agents are preferred.

On the other hand, the manager may have several reasons for wanting to develop agents' expertise equally. Agents may cover for each other more easily. Server pooling to reduce system congestion becomes easier. Agents perceive their own workload to be more fair when everyone else receives the same assignment. The firm may reclaim some bargaining power with respect to salaries by distributing necessary skills more evenly. If such reasons predominate, objective U_s is the better choice.

An additional subtlety exists in the present context regarding the supervisor's utility U_s, the straight sum of expertise values in a group. Note that Chapter 3 examines these utility functions where all calls are classified into a single type. But Chapter 4 shows that agents face a variety of call types, and each type is characterized by different average performance measures. In fact, we find that each agent-type pair exhibits a unique performance trend. Now we broaden our approach to accommodate both multiple call types and multiple agents when pursuing objective functions for expertise development. *Given differing call types and agent abilities, the optimal value of the sum of expertise may not be produced by a completely even assignment of calls to agents.* But from our experience of how to maximize a concave sum, we can expect that the best assignment for U_s will be more even than the best assignment for U_c.

Our manager knows that during her career she may face a different balance favoring one or the other for a specific agent group and time period. Here we will consider both U_c and U_s as objective functions, and explore the ramifications of both choices.

As a first step, our manager collects data to determine an accurate estimate of her agents' potential to improve, such as the data shown in Figure 5.1 that plots one agent's change in performance on a specific task. She has confidence in her data, and can estimate well the capabilities of untried agents by matching their human resources profiles with past work histories of veterans who were once at the same stage of expertise.

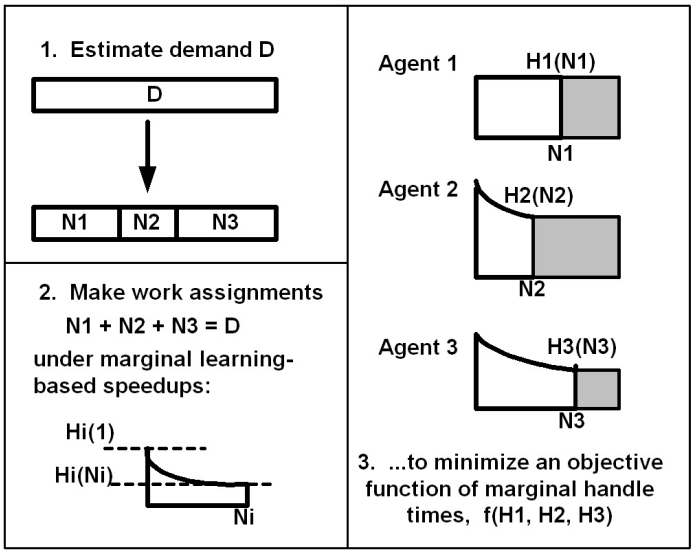


Figure 5.2: **Objective: Minimize a Function of Marginal Handle Times.** Given a demand forecast D, we must assign N_i jobs to agents $i \in \{1, 2, 3\}$, such that $\sum_{i=1}^{3} N_i = D$. We require that our optimization objective take into account the dynamic of agent handle time decreasing through cumulative experience.

The model of expertise development appears at the right of Figure 5.1. Recall that this model is the commonly used log-linear learning curve applied to improvements in an agent's handle time, where the call handle time for agent i and call type k is a decreasing function of cumulative production, $H_{ik}(N_{ik}) = H_{ik}(1) \cdot N_{ik}^{-b_{ik}}$. Figure 5.2 illustrates this dynamic in the presence of multiple agents. The organization also conducts research for quality assurance purposes on agent performance, and finds that improved efficiency often correlates with improved expertise on other dimensions of service quality—by encouraging efficiency improvements, she also hopes to boost quality metrics that are harder to quantify.

Our manager considers whether or not it is useful to specify routing priorities, and works out

a simple way to estimate their importance. Say $H_{ik}(N_M)$ is the handle time after an agent takes some number of calls N_M, where N_M is the maximum number of calls of type k that could be routed to i in one period. Then to attain a percentage p of this improved efficiency, the agent needs to be routed N_p calls, where

$$\begin{aligned} H_{ik}(N_p) &= H_{ik}(N_M) + p \cdot [H_{ik}(1) - H_{ik}(N_M)] \\ H_{ik}(1) \cdot N_p^{-b_{ik}} &= H_{ik}(1) \cdot N_M^{-b_{ik}} + p \cdot [H_{ik}(1) - H_{ik}(1) \cdot N_M^{-b_{ik}}] \\ N_p &= \left((1-p) + p \cdot N_M^{-b_{ik}}\right)^{(-1/b_{ik})} \end{aligned} \tag{5.2}$$

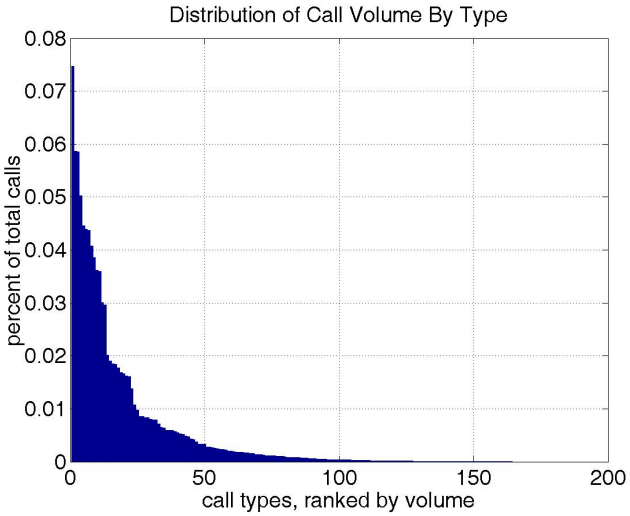


Figure 5.3: **Example of Arrival Rates.** Different call types to have different arrival rates. Routing rules that encourage priorities or specialization are most important when significant learning occurs among the low-volume call types.

For example, if $b_{ik} = 0.01$ and $N_M = 300$, to attain $p = 75\%$ of the potential efficiency improvement this agent must be routed a total of $N_p = 70$ calls. In general it would be easy to route N_p calls to i if the arrival rate λ_k of call type k is large, so that i receives frequent opportunities for service through a policy of agents taking the first waiting customer—then routing priorities may be unnecessary. However, that is not the case for many call types; for instance, see Figure 5.3 for the empirical distribution of call type arrival rates from Chapter 4. In our example, consider the volume of other call types to be so large that they crowd out our agent's favored type, making it unlikely a

no-priority policy will cause agent i to get 70 calls within the optimization period. Were our manager to ascertain that many examples like this exist in her group—that certain of her agents will progress rapidly on specific lower-volume call types—then objective U_c is preferred, because her assignment objective must introduce routing priorities to match capable agents to favored task types that they would otherwise rarely see.

5.2.2 Maximizing the Customer's Utility Function U_c

Here we can also define the total customer's utility U_c over all agents and call types as the *expected value of expertise,* as discussed in Chapter 3. There, a quantity $0 \leq X \leq 1$ represented a positive experience level. Now, consider instead the slowest mean handle time H_k among the agent-type pairs to be zero expertise; and let positive quantities of expertise be represented by the difference between H_k and faster values H_{ik}.

- We can write the expertise level as $X_{ik} = H_k - H_{ik}$.
- In the cumulative production model $H_{ik}(N_{ik}) = H_{ik}(1) \cdot N_{ik}^{-b_{ik}}$.
- Substitution in the first equation gives $X_{ik} = H_k - H_{ik}(1) \cdot N_{ik}^{-b_{ik}}$.
- The customer's utility is defined as $U_c = \sum_j p_j X_j$ for some j, requiring routing proportions p_j. Here, proportions p_{ik} for agent i and call type k may be obtained by splitting the number of calls assigned N_{ik} into $N_{ik} = p_{ik} \cdot D_k$, where D_k is the demand forecast for type k calls.

The customer's objective, the expected value of expertise, may now be expressed as

$$\sum_{i}^{I} \sum_{k}^{K} p_{ik} X_{ik} = \sum_{i=1}^{I} \sum_{k=1}^{K} p_{ik} \cdot \left[H_k \; - \; \left(H_{ik}(1) \cdot D_k^{-b_{ik}} \cdot p_{ik}^{-b_{ik}} \right) \right] \tag{5.3}$$

Note that the routing proportions sum to one for every call type, or $\sum_i^I p_{ik} = 1$. Also note that expertise now has units of time (say, minutes) and its values lie in the range $0 < X_{ik} \leq H_k$.

Our manager observes that Equation (5.3) would better reflect the value of agents' expertise to the contact center if they were weighted by the relative arrival rates of the different call types. She

therefore assigns weights to each term $w_k = \frac{\lambda_k}{I \cdot \sum_{k=1}^{K} \lambda_k}$, where λ_k represents the mean arrival rate of type-k customers to the center. The term I in the denominator here allows the weights to sum to 1 over all terms in the double summation, but it also reduces the numerical accuracy of a solver. Since it is present identically in all terms, it can be discarded. Note that the weights w_k could potentially incorporate other factors, such as the profitability of serving each of k different classes of customers. Weights w_{ik} could also be defined to account for unique costs or other characteristics of individual agents.

We now define the final form of the **customer's utility function** as

$$U_c(p_{ik}) = \sum_{i=1}^{I} \sum_{k=1}^{K} w_k \cdot p_{ik} \cdot \left[H_k \; - \; \left(H_{ik}(1) \cdot D_k^{-b_{ik}} \cdot p_{ik}^{-b_{ik}} \right) \right]. \tag{5.4}$$

When our manager decides the customer's utility—the expected value of expertise—is the right quantity to optimize, she would like to maximize Equation (5.4) as the objective, written

$$\max_{p_{ik}} \; \left(U_c(p_{ik}) \right), \text{ or equivalently } \min_{p_{ik}} \; \left(-1 \cdot U_c(p_{ik}) \right).$$

Unfortunately as we saw before (pages 48 and 53), and show in Lemma 5.2.1 below, $U_c(N_{ik})$ is a convex function, and so it presents the potential for multiple local maxima. It does not have convenient optimality properties—the local maximum that is returned depends on the initial conditions the solver sees, and it may not be the true global maximum.

As a technical matter before stating the next result, we note that all routing proportions p_{ik} are constrained to be strictly greater than zero in the targets we seek from a solver, although p_{ik} is allowed to be very small, so that it is acceptable for $p_{ik} \cdot D_k \approx 0$.

Lemma 5.2.1. *The objective function of Equation (5.4), $U_c(p_{ik})$, is a **convex function** of p_{ik}.*

Proof. To show that $U_c(p_{ik})$ is convex in p_{ik}, we must show that $\vec{v}^T \nabla^2 U_c(p_{ik}) \vec{v} \geq 0$ for any real-valued vector $\vec{v}$ of length $I * K$ (see Boyd and Vandenberghe [2004], page 74, or Nash and Sofer [1996], pages 22-23). Here $\nabla^2 U_c(p_{ik})$ is the Hessian matrix, I is the total number of agents, and K is

the total number of call types. First, note that

$$\begin{aligned}
U_c(p_{ik}) &= \sum_{i=1}^{I}\sum_{k=1}^{K} w_k \cdot p_{ik} \cdot \left[H_k \; - \; \left(H_{ik}(1) \cdot D_k^{-b_{ik}} \cdot p_{ik}^{-b_{ik}}\right)\right] \\
&= \sum_{i}^{I}\sum_{k}^{K} w_k \cdot p_{ik} \cdot H_k \; - \; \sum_{i=1}^{I}\sum_{k=1}^{K} w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}} \cdot p_{ik}^{(1-b_{ik})} \\
\text{Constant } C_1 &= \sum_{i=1}^{I}\sum_{k=1}^{K} w_k \cdot p_{ik} \cdot H_k = \sum_{k}^{K} w_k \cdot H_k \qquad (5.5) \\
U_c(p_{ik}) &= C_1 \; - \; \sum_{i=1}^{I}\sum_{k=1}^{K} w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}} \cdot p_{ik}^{(1-b_{ik})} \\
&= C_1 \; - \; \sum_{i}^{I}\sum_{k}^{K} C_{ik} p_{ik}^{(1-b_{ik})} \qquad (5.6)
\end{aligned}$$

where terms $C_{ik} = w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}}$ are constant with respect to p_{ik}, and Equation (5.5) holds because $\sum_i^I p_{ik} = 1$. $U_c(p_{ik})$ is now a constant minus a sum, and the nonzero Hessian terms are

$$\begin{aligned}
\frac{\partial U_c(p_{ik})}{\partial p_{ik}} &= -(1 - b_{ik}) \cdot C_{ik} \cdot p_{ik}^{-b_{ik}} \\
\frac{\partial^2 U_c(p_{ik})}{\partial p_{ik}^2} &= b_{ik} \cdot (1 - b_{ik}) \cdot C_{ik} \cdot p_{ik}^{-(1+b_{ik})} \geq 0 \\
\nabla^2 U_c(p_{ik}) &= \text{diag}\left(b_{ik} \cdot (1 - b_{ik}) \cdot C_{ik} \cdot p_{ik}^{-(1+b_{ik})}\right). \qquad (5.7)
\end{aligned}$$

The Hessian matrix $\nabla^2 U_c(p_{ik})$ is thus a diagonal matrix of size $(I * K, I * K)$, with strictly positive elements on the diagonal (because $p_{ik} > 0$). Therefore all of its eigenvalues are positive; $\nabla^2 U_c(p_{ik})$ is positive definite (see for example Lay [1994], pages 289 and 416); and the quadratic form $\vec{v}^T \nabla^2 U_c(p_{ik}) \vec{v} > 0$. It is therefore also positive semidefinite. Thus we have shown that $U_c(p_{ik})$ is a convex function of p_{ik}, and the lemma is proved. □

Finding the best solution to objective $U_c(p_{ik})$ involves submitting the problem $(\max_{p_{ik}} U_c(p_{ik}))$ to a nonlinear solver; but as discussed above, the global optimum may not be returned, or the solver may have trouble converging to a solution. Our operations manager devises the following work-around. The first-order Taylor series approximation to the nonlinear term $p_{ik}^{-b_{ik}}$ is just the linear term p_{ik}. Her data reveal the values of the learning exponents to be $b_{ik} \leq 0.1$ among the agent-type combinations, so the maximum linearization error for one p_{ik} term in the sum is about 4% at $p_{ik} = 0.349$—and the

error is much less for other p_{ik} values. She accepts the trade-off of losing a small amount of accuracy to guarantee the solver's convergence to a good solution, and therefore the new **linearized objective function for the customer's utility** becomes

$$U_c(p_{ik}) = \sum_{k}^{K} w_k \cdot H_k \; - \; \sum_{i=1}^{I}\sum_{k=1}^{K} \left(w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}} \right) \cdot p_{ik}. \tag{5.8}$$

5.2.3 Maximizing the Supervisor's Utility Function U_s

To maximize the sum of expertise U_s in the current context, where a high level of expertise is expressed as fast service, we need to minimize the sum of final marginal handle times. Thus our manager starts with an objective function to minimize the sum of $H_{ik}(N_{ik})$ values over all agent-type pairs, and again includes the weighting by call type arrival rate $w_k = \frac{\lambda_k}{I \cdot \sum_{k=1}^{K} \lambda_k}$. This gives

$$U_s(p_{ik}) \; = \; \sum_{i=1}^{I}\sum_{k=1}^{K} \left(w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}} \right) \cdot p_{ik}^{-b_{ik}}. \tag{5.9}$$

We seek the values p_{ik}^* that minimize this function, or $\min_{p_{ik}} \; (U_s(p_{ik}))$. An optimal routing target p_{ik}^* may show some unevenness, and increase assignments according to high-volume call types given by w_k values, and according to high-performing agents as determined by low $H_{ik}(1)$ and high b_{ik} values. Nevertheless as we show in Lemma 5.2.2 below this is a convex function we wish to minimize; in a solver that may be transformed to the problem of maximizing a concave function, which we saw in Chapter 3 was maximized by making **even assignments** when agents were equally capable. Our manager therefore expects the routing targets derived from $U_s(p_{ik})$ and the associated best routing rule to promote more even assignments than would be the case for the U_c objective, which explicitly seeks the most extreme specialized assignments.

Lemma 5.2.2. *The objective function of Equation (5.9), $U_s(p_{ik})$, is a **convex function** of p_{ik}.*

Proof. Proceeding as in Lemma 5.2.1, we have

$$\begin{aligned}
U_s(p_{ik}) &= \sum_{i=1}^{I}\sum_{k=1}^{K}\left(w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}}\right) \cdot p_{ik}^{-b_{ik}} \\
&= \sum_{i=1}^{I}\sum_{k=1}^{K} C_{ik} \cdot p_{ik}^{-b_{ik}} \\
\frac{\partial U_s(p_{ik})}{\partial p_{ik}} &= -b_{ik}C_{ik}p_{ik}^{-(1+b_{ik})} \\
\frac{\partial^2 U_s(p_{ik})}{\partial p_{ik}^2} &= b_{ik}(1+b_{ik})C_{ik}p_{ik}^{-(2+b_{ik})} \geq 0 \\
\nabla^2 U_s(p_{ik}) &= \operatorname{diag}\left(b_{ik}(1+b_{ik})C_{ik}p_{ik}^{-(2+b_{ik})}\right). \qquad (5.10)
\end{aligned}$$

The Hessian matrix $\nabla^2 U_s(p_{ik})$ is thus a diagonal matrix of size $(I*K, I*K)$, with positive elements on the diagonal. Therefore as we saw for the Hessian matrix in Lemma 5.2.1, page 89, $\vec{v}^T \nabla^2 U_s(p_{ik})\vec{v} \geq 0$, and the lemma is proved. □

5.2.4 Design of the Constraints

Our operations manager notes that three constraints are implied by Figure 5.2. First, the sum of the calls must equal the demand forecast D_k, or $\forall\ k, \quad \sum_{i=1}^{I} N_{ik} = D_k$. Second, every agent must get some calls—Agent 2 appears the least efficient, but still receives a nonzero allocation. This is because substantive changes in expertise accrue slowly and in parallel to the daily peaks and valleys of incoming traffic, and even the less promising agents will be needed during the peak times to keep customer waiting times reasonable. Thus some reserve capacity for peak times must be built into the solver's work assignment—and informed by Chapter 4's data, we take that to mean that every agent must be taking calls during at least 20% of the time during the optimization period. This capacity is captured in a set of *minimum utilization constraints*, which we will also refer to simply as *minimum constraints*. This discussion also informs the question of what time span the work assignments should cover. Since meaningful expertise gains occur after completing tens or hundreds of similar tasks, while agents only get a few chances each day to engage in the same task, the time horizon for the manager's expertise-aware routing assignments will be some number of months.

Third, agents must not be overloaded. This is represented by the gray areas next to agent allocations that show unused capacity. Both behavioral and operational issues may be at stake: large differences in volumes of work assigned will strike some agents as unfair, and generating extremely tight work schedules through routing targets leaves agents with no flexibility to cope with day-to-day fluctuations around the average demand forecasts. These ideas are captured in *maximum utilization constraints*, or just *maximum constraints*. Chapter 4's data suggest a maximum utilization constraint where an agent is between 50% and 60% utilized during the optimization period.

To express the minimum and maximum constraints, the manager evaluates the total time taken by agent i to handle all calls in his assignment of type k, for every agent-type pair. This is accomplished by the following steps.

- If an agent's handle time does not change, he takes $T_{ik} = \sum_{n=1}^{N} H_{ik}(n)$ minutes to handle N jobs, where $H_{ik}(n)$ is the time needed for the nth call; if $H_{ik}(n)$ is constant, $H_{ik}(n) = H_{ik}(1)$, and $T_{ik} = N \cdot H_{ik}$.

- Assume a unique log-linear learning curve function exists for every agent-type pair. $H_{ik}(n)$ decreases monotonically as we have seen compared with the first handle time $H_{ik}(1)$ as cumulative production n increases.

- As before, the cumulative production total N_{ik} and the routing target proportion p_{ik} are related by $N_{ik} = p_{ik}D_k$, where D_k is the demand forecast for type k calls. Here it is convenient to work with N_{ik}.

- Specific limits can be placed on the value of the learning exponent b_{ik}, based on observations from Chapter 4: $0 \leq b_{ik} \leq 0.1$, and therefore $0.9 \leq (1 - b_{ik}) \leq 1$.

- In the presence of a learning curve, the time taken to handle N jobs becomes $T_{ik} = \sum_{n=1}^{N} H_{ik}(n) \approx \int_{v=0}^{N} H_{ik}(v)\mathrm{d}v$.

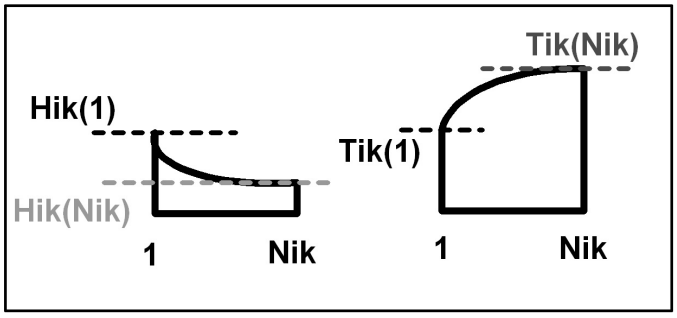


Figure 5.4: **Convex Objective and Concave Constraint Functions in N_{ik}.** Left: plot of a convex log-linear learning curve $H_{ik}(N_{ik}) = H_{ik}(1)N_{ik}^{-b_{ik}}$. The curve shows marginal values of handle time per call decreasing with cumulative production N_{ik}, for agent i and call type k. Right: the sum of all the handle times from the left-hand curve becomes a concave function, the total time $T_{ik}(N_{ik})$—note the scale of the peak is attenuated to fit on the plot.

A good closed-form approximate expression for T_{ik} is:

$$T_{ik} \approx H_{ik}(1)\int_{v=0}^{N} v^{-b_{ik}}\mathrm{d}v = \frac{H_{ik}(1)\cdot N^{(1-b_{ik})}}{(1-b_{ik})}. \tag{5.11}$$

Now the set of maximum utilization constraints may be written as

$$\forall \text{ i } \sum_{k=1}^{K} T_{ik} \leq T_{max}. \tag{5.12}$$

and the set of minimum utilization constraints may be written as

$$\forall \text{ i } \sum_{k=1}^{K} T_{ik} \geq T_{min} \tag{5.13}$$

The manager sets the values of T_{min} and T_{max} by looking at empirical trends of agent utilization, as those reveal the organization's consensus about the sustained rates at which agents should be busy over multiple-month time periods. Again, as her data is similar to the quality regime call center data in Chapter 4, the minimum utilization might be 20%, and the maximum 60%. Multiplying the standard number of minutes worked per day by the number of working days in the optimization period gives the total time T_{work} corresponding to 100% utilization. Let ρ_{max} be the busiest allowed mean utilization rate over the period; then $T_{max} = \rho_{max} \cdot T_{work}$. Similarly, ρ_{min} is the lowest allowed mean utilization rate over the period, and $T_{min} = \rho_{min} \cdot T_{work}$.

Figure 5.4 compares the forms of the convex marginal handle time function $H_{ik}(N_{ik})$ and the concave total time $T_{ik}(N_{ik})$. Note that $T_{ik}(N_{ik})$ appears in two constraint functions with opposite relational symbols—greater than or equal to T_{min}, and less than or equal to T_{max}—but when solving an optimization problem, constraints must be able to be expressed in a standard form where all inequalities point in the same direction. We take that direction to be less-than-or-equal-to, which means the minimum constraint must be inverted. When that occurs, the sum of concave functions in the minimum constraint is also inverted, resulting in the convex function $\frac{T_{min}}{\sum_{k=1}^{K} T_{ik}}$.

The shapes of the constraint functions become important in Section 5.2.6, so we conclude this section with two results that describe their convexity properties. For convenience we refer to the the number of calls N_{ik} as the optimization variable rather than the routing target proportion of calls p_{ik}. Again, these are simply related by $N_{ik} = p_{ik}D_k$, where exogenous constant parameter D_k is the demand forecast of type k calls.

Lemma 5.2.3. *The maximum utilization constraint function in Inequality (5.12), $\sum_{k=1}^{K} T_{ik}(N_{ik}) \leq T_{max}$, is a **concave function** of the number of calls N_{ik}, and thus of the routing target proportion p_{ik}.*

Proof. Let $g(N_{ik}) = \sum_{k=1}^{K} T_{ik}(N_{ik})$. Then Inequality (5.12) may be written $g(N_{ik}) \leq T_{max}$. We prefer to define constraints in a standard form where the right-hand side is zero, so then this becomes $\frac{g(N_{ik})}{T_{max}} - 1 \leq 0$. In this case, for $g(N_{ik})$ to be concave we must show that $\vec{v}^T \nabla^2 g(N_{ik}) \vec{v} \leq 0$ for any real-valued vector $\vec{v}$ of length K. There is a separate constraint for every agent i, so we are concerned only with a single summation over K call types in this case. Note that:

$$
\begin{aligned}
g(N_{ik}) &= \sum_{k=1}^{K} T_{ik}(N_{ik}) \\
\forall \text{ i, k:} \quad T_{ik}(N_{ik}) &= \frac{H_{ik}(1) \cdot N_{ik}^{(1-b_{ik})}}{(1-b_{ik})} \\
\frac{\partial g(N_{ik})}{\partial N_{ik}} &= H_{ik}(1) \cdot N_{ik}^{(-b_{ik})} \\
\frac{\partial^2 g(N_{ik})}{\partial N_{ik}^2} &= -b_{ik} H_{ik}(1) N_{ik}^{-(1+b_{ik})} \leq 0 \qquad (5.14) \\
\nabla^2 g(N_{ik}) &= \operatorname{diag}\left(-b_{ik}(1+b_{ik}) w_k H_{ik}(1) N_{ik}^{-(1+b_{ik})}\right). \qquad (5.15)
\end{aligned}
$$

The Hessian matrix $\nabla^2 g(N_{ik})$ is thus a diagonal matrix of size (K, K) that has strictly negative elements on the diagonal. This latter property is true because $p_{ik} > 0$, we take $D_k > 0$, and so $N_{ik} = p_{ik} \cdot D_k > 0$; therefore terms like (5.14) are strictly negative. Thus we know that all the eigenvalues of $\nabla^2 g(N_{ik})$ are negative, and so $\nabla^2 g(N_{ik})$ is a negative definite matrix (see Lay [1994], pages 289 and 416). It is also a negative semidefinite matrix, and we know for the quadratic form that $\vec{v}^T \nabla^2 g(N_{ik}) \vec{v} \leq 0$ for any vector $\vec{v}$. Therefore $g(N_{ik})$ is a concave function of N_{ik}. Substituting $p_{ik} = N_{ik}/D_k$ does not change the sign of terms like (5.14)—the only change is a constant multiplier—and thus $g(N_{ik})$ is also a concave function of p_{ik}, and the lemma is proved. □

Lemma 5.2.4. *The minimum utilization constraint function in Inequality (5.13), $\sum_{k=1}^{K} T_{ik}(N_{ik}) \geq T_{min}$, is a **convex function** of the number of calls N_{ik}, and thus of the routing proportion p_{ik}.*

Proof. Inequality (5.13) in standard form becomes $\frac{T_{min}}{\sum_{k=1}^{K} T_{ik}(N_{ik})} \leq 0$. Now the function we are concerned with in this case is $g(N_{ik}) = \frac{T_{min}}{\sum_{k=1}^{K} T_{ik}(N_{ik})}$. Note that:

$$\begin{aligned} g(N_{ik}) &= \frac{T_{min}}{\sum_{k=1}^{K} T_{ik}(N_{ik})} \\ \forall \text{ i, k: } \quad T_{ik}(N_{ik}) &= \frac{H_{ik}(1) \cdot N_{ik}^{(1-b_{ik})}}{(1-b_{ik})} \\ \frac{\partial g(N_{ik})}{\partial N_{ik}} &= \frac{-T_{min}}{(\sum_{k=1}^{K} T_{ik})^2} H_{ik}(1) \cdot N_{ik}^{(-b_{ik})} \end{aligned}$$

Diagonal elements of the Hessian are given by:

$$\frac{\partial^2 g(N_{ik})}{\partial N_{ik}^2} = \frac{b_{ik} T_{min} H_{ik}(1) N_{ik}^{-(b_{ik}+1)}}{(\sum_{k=1}^{K} T_{ik})^2} + \frac{2 T_{min} H_{ik}^2(1) N_{ik}^{-2b_{ik}}}{(\sum_{k=1}^{K} T_{ik})^3} \geq 0 \tag{5.16}$$

Off-diagonal elements of the Hessian are given by:

$$\frac{\partial^2 g(N_{ik})}{\partial N_{ik} \partial N_{jm}} = \frac{2 T_{min} H_{ik}(1) N_{ik}^{-b_{ik}} \cdot H_{jm}(1) N_{jm}^{-b_{jm}}}{(\sum_{k=1}^{K} T_{ik})^3} \geq 0 \tag{5.17}$$

The Hessian matrix $\nabla^2 g(N_{ik})$ is thus a square real symmetric matrix of size (K, K) with all elements on and off the diagonal greater than or equal to zero. For conciseness, let $\Sigma_K = \sum_{k=1}^{K} T_{ik}(N_{ik})$, let $F_{ik} = H_{ik}(1) \cdot N_{ik}^{-b_{ik}}$, and define a vector of F_{ik} values to be $\vec{F} = (F_{i1}, F_{i2}, ..., F_{iK})$.

Again, we take $p_{ik} > 0$, $D_k > 0$, and $N_{ik} = p_{ik} \cdot D_k > 0$, so that we have

$$\begin{aligned}
\nabla^2 g(N_{ik}) &= \left(\frac{T_{min}}{(\Sigma_K)^3}\right) \cdot \left[\text{diag}\left(\Sigma_K b_{ik} F_{ik} N_{ik}^{-1}\right) + 2\left(\vec{F} \cdot \vec{F}^T\right)\right] \\
\vec{v}^T \nabla^2 g(N_{ik}) \vec{v} &= \left(\frac{T_{min}}{(\Sigma_K)^3}\right) \cdot \left[\Sigma_K \cdot \left(\sum_{k=1}^{K} v_k^2 b_{ik} F_{ik} N_{ik}^{-1}\right) + 2\left(\sum_{k=1}^{K} v_k F_{ik}\right)^2\right] > 0.
\end{aligned}$$

Therefore we may claim that $\vec{v}^T \nabla^2 g(N_{ik}) \vec{v} \geq 0$, so that $g(N_{ik})$ is a convex function of N_{ik}. As in Lemma 5.2.3, substituting $p_{ik} = N_{ik}/D_k$ does not change the sign of terms like (5.16) or (5.17), so we know that $g(N_{ik})$ is also a convex function of p_{ik}, and the lemma is proved. □

5.2.5 Linearizing the Concave Maximum Constraint

Our operations manager seeks a method for computing work assignments that converges reliably to an optimal solution, and produces repeatable results given the same set of inputs. But those qualities are not guaranteed when using the set of nonlinear constraint functions developed in the previous section. In particular, note that the maximization constraint in standard form is concave. When combined with the convex objective and minimum constraint, that may cause a solver to be trapped in local minimum, hence failing to find the true global minimum. In this section, we show how to overcome this problem and design a *convex program* that consistently terminates in a unique optimal solution value.

Unfortunately, to obtain the convex program we must sacrifice some accuracy in the maximum utilization constraint. This arises because all functions of N_{ik} whose sum is suited to be bounded by a maximum constraint here are increasing in N_{ik}—for example, $T_{ik}(N_{ik})$, $H_{ik}(1) - H_{ik}(N_{ik})$, and so on. (Of course a sum of just the marginal values $H_{ik}(N_{ik})$ is not useful in a maximum constraint.) But learning curves exhibit diminishing returns with increased production, so all the functions we might use that are increasing in N_{ik} happen to be concave functions. Thus we have (concave function $\leq T_{max}$), destroying the convexity property of the nonlinear program.

To resolve this, we linearize the maximum utilization constraint. Instead of $\forall$ i $\sum_{k=1}^{K} T_{ik} \leq T_{max}$, we use the sum $\forall$ i $\sum_{k=1}^{K} \beta_{ik} H_{ik}(1) N_{ik} \leq T_{max}$. Here $\beta_{ik} > 0$ is a discount factor. This lin-

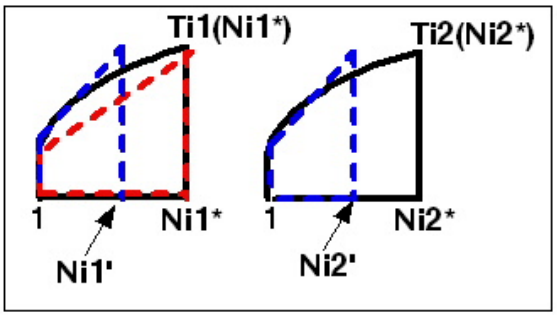


Figure 5.5: **The Effect of Linearization.** Linearization of the terms in the maximum utilization constraint achieves a convex program at the cost of some accuracy.

earization step implicitly assumes the following approximation is sufficiently accurate for each agent-call type pair:

$$\beta_{ik} H_{ik}(1) \cdot N_{ik} \quad \approx \quad T_{ik}(N_{ik}) = \frac{H_{ik}(1) \cdot N_{ik}^{(1-b_{ik})}}{(1-b_{ik})} \tag{5.18}$$

The approximation error increases as the learning exponent b_{ik} increases. Figure 5.5 illustrates the problem.

- Consider an example with two call types, where $T_{i1}(N_{i1}^*) + T_{i2}(N_{i1}^*) = T_{max}$.
- Here the constraint is active at the solution, $T_{i1}(N_{i1}^*) = \frac{H_{i1}(1) \cdot (N_{i1}^*)^{(1-b_{i1})}}{(1-b_{i1})}$, and $T_{i2}(N_{i2}^*) = \frac{H_{i2}(1) \cdot (N_{i2}^*)^{(1-b_{i2})}}{(1-b_{i2})}$.
- In the figure, the value $T_{i1}(N_{i1}^*)$ given by the solid curve is attained early by $H_{i1}(1) \cdot N'_{i1}$, given by the dotted line, and $N_{i1}^* > N'_{i1}$. Thus the new maximum constraint has become active prematurely; agent i could have actually handled more calls of type 1 beyond what the new linearized constraint allowed. A similar effect occurs for type 2 calls.

As a means to control this error, we specify a discount factor $\beta_{ik} > 0$. This reduces the slope of the linear approximation, which is illustrated in the diagram of type 1 calls in Figure 5.5. Here a value of $0 < \beta_{ik} < 1$ is fortuitously chosen to make $N_{i1}^* - N'_{i1} = 0$, as shown by the second dotted

line. But because N_{ik}^{*} values are optimization outputs, and not knowable in advance, the choice of β values must be estimated before the run.

To take full advantage of the discount β_{ik}, the user must devise a preliminary estimation step to judge how large it should be with respect to other agents' H_{ik} and b_{ik} parameters, arrival rates λ_k, and other factors. For the scenarios we examine, we do not see significant errors even when setting $\beta_{ik} = 1$—the maximum constraint is not often active—and leave the development of a suitable method to set β_{ik} as an open estimation problem. If necessary, users can also simply shorten the optimization period to reduce this error, and re-optimize with a new set of updated $H_{ik}(1)$ values; and Appendix 5.A describes another experimental method for reducing some of this linearization error.

5.2.6 A Nonlinear Program to Find Good Call Routing Targets

Below we present two convex programs with linearized maximum constraints, known as Program CP-U_c and Program CP-U_s. *These two programs are used to generate the optimal call routing targets for Section 5.3.* The constraint functions are the same for both objective functions; $U_c(p_{ik})$ maximizes the linearized customer's utility function and $U_s(p_{ik})$ maximizes the supervisor's utility function over all agent-type pairs by varying call routing proportions p_{ik}. Here all functions are written in terms of the optimization variables p_{ik}.

$$\boxed{\textbf{CP-}U_c} \quad \max_{p_{ik}} \; U_c(p_{ik}) = \sum_{k}^{K} w_k \cdot H_k \; - \; \sum_{i=1}^{I}\sum_{k=1}^{K}\left(w_k \cdot H_{ik}(1) \cdot D_k^{-b_{ik}} \cdot p_{ik}\right) \tag{5.19}$$

such that

$$\forall \text{ i, k:} \quad N_{ik} = p_{ik} \cdot D_k \quad \text{“number of calls”}$$

$$\forall \text{ i, k:} \quad T_{ik}(p_{ik}) = \frac{H_{ik}(1) \cdot (p_{ik} \cdot D_k)^{(1-b)}}{(1-b)} \quad \text{“time taken”}$$

$$\forall \text{ k:} \quad w_k = \frac{\lambda_k}{\sum_{k=1}^{K} \lambda_k} \quad \text{“arrival rate weighting”}$$

$$\forall \text{ i} \quad \sum_{k=1}^{K} \beta_{ik} H_{ik}(1)(p_{ik} \cdot D_k) \leq T_{max} \quad \text{“max. utilization”} \tag{5.20}$$

$$\forall \text{ i} \quad \sum_{k=1}^{K} T_{ik}(p_{ik}) \geq T_{min} \quad \text{“min. utilization.”} \tag{5.21}$$

$$\forall \text{ k} \quad \sum_{i=1}^{I} p_{ik} \cdot D_k = D_k \quad \text{“full service”} \tag{5.22}$$

$$\forall \text{ i, k} \quad p_{ik} > 0 \quad \text{“}p_{ik}\text{ positive”} \tag{5.23}$$

$$\boxed{\textbf{CP-}U_s} \quad \min_{p_{ik}} \quad U_s(p_{ik}) = \sum_{i=1}^{I}\sum_{k=1}^{K} w_k H_{ik}(1)(p_{ik} \cdot D_k)^{-b_{ik}} \tag{5.24}$$

such that...(as above)

We adopt the form of a convex program for each objective in order to guarantee the existence of a global minimum objective value. A convex program is defined as an optimization program that minimizes a convex function over a convex set, where the convex set is formed from a set of nonlinear convex inequality constraints and linear equality constraints. See Boyd and Vandenberghe [2004], page 136; Bertsekas [2003], page 208; or Nash and Sofer [1996], page 473 for proofs and examples. It is known that convex programs yield unique global solution values, and commonly used interior point solvers such as Matlab's *fmincon()* function can find these global solutions in a reasonable time by means of a variant of Newton's method (see Coleman and Zhang [2009], pages 3–18). Now we are in a position to prove the result:

Theorem 5.2.5. *Program CP-U_c is a convex program.*

Proof. Note the following properties of Program CP-U_c's components:

1. By Lemma 5.2.1, page 89, the objective function is a linear function, which for the purpose of classifying a nonlinear optimization program may be considered a convex function.

2. By Lemma 5.2.4, page 96, Inequality (5.21) is a convex nonlinear inequality constraint.

3. Inequality (5.20) is a linear inequality constraint, and thus can be considered a convex inequality constraint.

4. The full service constraint is a linear equality constraint.

Together these four properties are sufficient to classify Program CP-U_c as a convex program, completing the proof. □

Theorem 5.2.6. *Program CP-U_s is a convex program.*

Proof. Note the following properties of Program CP-U_s's components:

- By Lemma 5.2.2, page 91, the objective function is a linear function, which for the purpose of classifying a nonlinear optimization program may be considered a convex function.

- Note that properties 2, 3, and 4 of Theorem 5.2.5 above apply to Program CP-U_s as well.

Together these properties are sufficient to classify Program CP-U_s as a convex program, completing the proof. □

5.2.7 Example of Work Assignment Optimization

Our manager knows or can predict the starting handle times $H_{ik}(1)$, and can predict b_{ik} and N_{ik} for future time periods. Then Program CP-U_c and CP-U_s on page 100 generate priority routing target proportions that account for improving handle time trends among agents and call types. Consider the following small example:

Input		*Output CP-U_c*	*Output CP-U_s*
Learning Exponents	Start Handle Times	Routing Targets	Routing Targets
b_{ik}	$H_{ik}(1)$	p_{ik}^{c*}	p_{ik}^{s*}
$\begin{bmatrix} 0.01 & 0.01 & 0 \\ 0.005 & 0.005 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 7.0 & 7.0 & 5.5 \\ 7.0 & 7.0 & 6.0 \\ 8.0 & 7.0 & 6.0 \end{bmatrix}$	$\begin{bmatrix} 0.29 & 0.02 & 1.0 \\ 0.71 & 0.4 & \approx 0 \\ \approx 0 & 0.58 & \approx 0 \end{bmatrix}$	$\begin{bmatrix} 0.35 & 0.09 & 0.83 \\ 0.58 & 0.50 & 0.01 \\ 0.06 & 0.40 & 0.16 \end{bmatrix}$

Table 5.1: **Optimal Work Assignment Example** Rows correspond to agents $i = 1$, 2, and 3; columns correspond to call types $k = 1$, 2, and 3. A larger value for b_{ik} indicates an agent-type pair for which larger learning-based improvement occurs. Call types are evenly weighted in the objectives. Note that $p_{ik} > 0$, but values lower than 0.01 are indicated by ≈ 0 in the routing target tables. As expected, maximized objective $U_c(p_{ik}^{c*}) > U_c(p_{ik}^{s*})$, minimized objective $U_s(p_{ik}^{s*}) < U_c(p_{ik}^{c*})$, and the sum of final handle times under p_{ik}^{s*} is lower than under p_{ik}^{c*}.

- Our call center has three agents, $I = 3$, and three call types, $K = 3$.
- The time span of interest is three months, or 60 work days, and agents take calls for equal shifts of 480 minutes per day; if an agent were 100% utilized, he would work for 28800 minutes during the entire period.
- The traffic forecast predicts an equal number of calls for each type, $D_k = 1660$, so the call types are evenly weighted in the objectives. At the time of the forecast, calls are expected to take around 6-7 minutes to handle, depending on the agent-type pairing.
- Staffing for the center has been set to match the volume forecast, such that the center's mean utilization is about 40%. To keep our nonlinear programming solution in line with this, $T_{max} = 15000$ minutes. We set the lower bound T_{min} to 5760 minutes, which corresponds to a utilization of 20%.

Under these conditions, Table 5.1 shows the optimal routing assignments for a roster of three agents taking three call types. At left are inputs: the learning exponents b_{ik}, and the starting handle times $H_{ik}(1)$. A larger value for b_{ik} indicates an agent-type pair for which larger learning-based improvement occurs. Here the numbers provide a challenge to the two objective functions $U_c(p_{ik})$ and

$U_s(p_{ik})$ from page 100, in that a good solution to one is likely to be a good solution to the other, and the solver must work hard to generate outputs that give superior differentiated results on each objective.

The right side of the table shows the routing proportions output by a solver for these two objectives. Both $U_c(p_{ik})$ and $U_s(p_{ik})$ start at the same initial solution when running in separate trials in the solver, and in each trial they improve their values until succeeding iterations of the total objective differ by no more than 1e-6 minutes.

For call types 1, 2, and 3, the final target routing proportions differ by 25%, 35%, and 34% respectively as the two objectives attempt to fulfill their differing aims. Let p_{ik}^{c*} represent the output under the customer's objective U_c, and let p_{ik}^{s*} represent the output under the supervisor's objective U_s.

- The maximized objective is $U_c(p_{ik}^{c*}) > U_c(p_{ik}^{s*})$ $(0.65 > 0.62)$.
- The minimized objective is $U_s(p_{ik}^{s*}) < U_c(p_{ik}^{c*})$ $(19.73 < 19.77)$.
- The sum of final marginal handle times under p_{ik}^{s*} is slightly lower than under p_{ik}^{c*} $(59.26 < 59.37)$.

This example demonstrates two other properties of the routing target computation. First, given the presence of the learning exponent b_{ik}, the optimal routing policy cannot be deduced simply by observing the starting handle times, $H_{ik}(1)$. Second, the presence of other agents and their individual characteristics changes the composition of the optimal solution. As a result, potential on-the-job learning opportunities may be denied to certain agents because their skills are needed elsewhere to improve the selected objective function's value.

5.3. Simulation of Routing Policies to Implement Optimal Work Assignment Targets

5.3.1 Approach to Simulation

Our nonlinear program outputs precise targets for the agents' workloads. Yet it is unclear how such precise targets may be implemented in the unpredictable daily call volume seen by a typical contact center. To gain insight into methods for implementing our ideal targets, we now describe results generated by a contact center simulator. This simulator generates streams of random numbers that imitate the random time points at which calls arrive, and the random lengths of time required to serve each customer.

The goal here is to develop and analyze routing rules that both show fidelity to our optimization targets, and show high performance on day-to-day operational metrics. Call centers depend on metrics that are not easily amenable to analytical methods and optimization, such as mean customer waiting time, the average service level, the average number of abandonments or dropped calls per period, and so on. All time-based metrics correlate fairly well with waiting time — more waiting time leads to predictably higher abandonment rates, and lower service levels — so we focus primarily on the average waiting time $\bar{W}$ as our key queueing simulation output.

In a single experiment, an optimization solver implementing Program CP-U_c or CP-U_s runs first. It defines the size and scope of the problem, and all relevant parameters. As described in Section 5.2.6, it outputs a list of target proportions for each agent-call type pair. It also generates two critical priority routing tables: the *agent-to-type map*, or just *agent map*, and the *type-to-agent-map*, or just *type map*. The simulator contains a routing subprogram that uses these tables to guide new customers to free agents during the run. When output from Program CP-U_c or CP-U_s, both routing tables follow exactly the ordering of target proportions. For instance, if an agent is assigned tasks A and B in proportions 50.1% and 49.9%, task A comes first in his row of the agent map.

5.3.2 Priority Routing Rules

The simulator uses the proportions and tables differently, depending on a key parameter: the *routing rule* or *policy*. Figure 5.6 illustrates the differences among the routing rules we will be exploring. In the top figure, the system does not prioritize calls by type, but only considers waiting time: a "no priority" policy. When available to take a call, an agent chooses the customer with the longest waiting time according to first-in-first-out order. The figure assumes the agents' mean service times are the same, so in the long run they each end up serving half of the incoming arrivals.

Without priorities, on-the-job learning specific to each call type still takes place, but it is not tracked. As we have seen before, this leads to expertise distributions among agents driven solely by the proportion of call types within the arrival stream, and each agent's potential to learn by serving those types. Agents serve many types, and spend a high proportion of their time in the steep starting phase of each learning curve. This policy both minimizes waiting time from server pooling, and maximizes the sum of expertise over all agents and types in the system.

Yet management may determine that more specialized expertise is needed than can be obtained under a no-priority rule. The bottom of Figure 5.6 illustrates policies that develop specialized expertise. Maximum specialization occurs when agents serve a single type, and sit idle when their preferred customer type is absent. However, this extreme rule is rarely feasible except in the largest systems. Instead, we may use rules that trade-off some waiting time performance to obtain a higher degree of specialized expertise. Such rules employ agent-to-type priority routing schedules that limit idling.

Consider a priority routing rule for Figure 5.6. Call types A and B arrive at the system requiring separate kinds of expertise—in a the financial services setting, A may represent questions about tax forms, and B may represent questions about credit card payments. Agents 1 and 2 have their own preferred call type, but may also assist by taking calls of the other type. Let the service rates μ_1 and μ_2 be constant at one customer per arbitrary time period. Then the arrival rates λ_A and λ_B and the

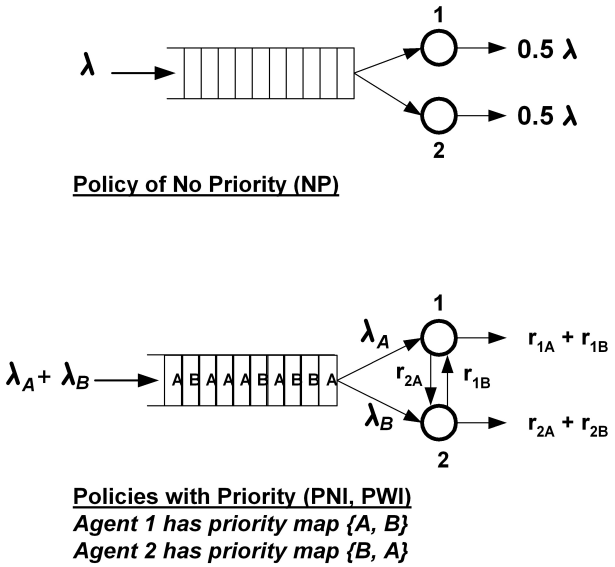


Figure 5.6: **Three Rules for Routing Calls To Agents.** Example of policy-dependent priority maps. Top: When no call type priorities are identified, policy NP, agents take any waiting call. Bottom: When priorities are present, agents choose the first waiting call of their preferred type. If no preferred customers are present, the agent may offer to serve other types of customers (priority-no-idle policies), or he may idle and wait for his preferred type (priority-with-idle policies). Here λ_A and λ_B are Poisson arrival rates for call types A and B; r_{ik} are rates of type k served by agent i.

routing rule determine the degree of task sharing, and thus the trade-off of waiting time for specialized expertise.

• **Routing Rule Example** Assume the lower diagram of Figure 5.6 is a system with Markovian arrival and service processes. Mean arrival rates are $\lambda_A = 0.45$ customers per arbitrary time period, $\lambda_B = 0.25$, and service rates are $\mu_1 = \mu_2 = 1$. The priority schedule, or priority map, for agent 1 is $\{$A, B$\}$. For agent 2 the map is $\{$B, A$\}$. We adopt the routing rule *priority-no-idle* (PNI): serve the highest priority customer from the priority map, if a customer of that type is present in the system; otherwise check if the other agent is busy and a customer of the other type is waiting—if so, then serve the other type.

These parameters suffice to determine the steady-state flows in the system: $r_{1A} = 0.365, r_{1B} = 0.029, r_{2A} = 0.085, r_{2B} = 0.221$. Figure 5.7 shows a simulation procedure's convergence to the final

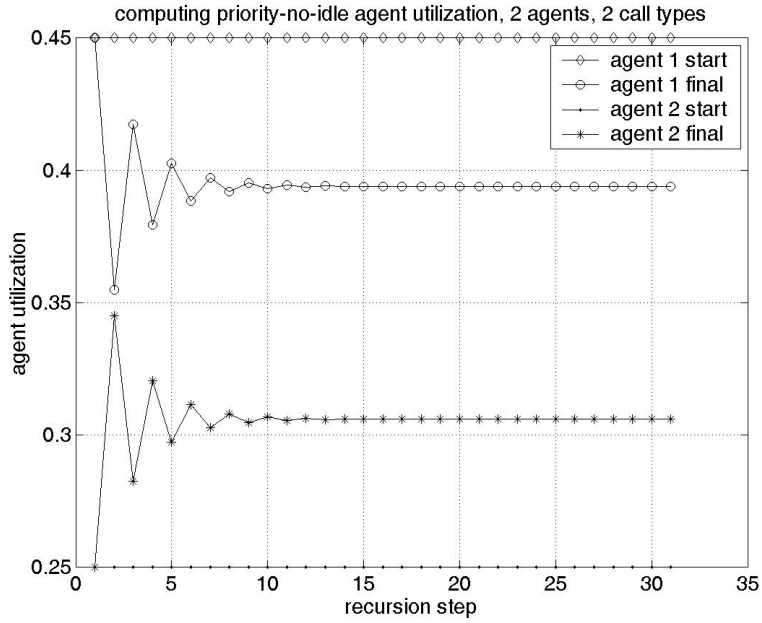


Figure 5.7: **Utilization for Two Agents Under Idle and No-Idle Routing.** Example 5.3.2 results. The two results in the middle are achieved under policy PNI, while the extreme top and bottom results are obtained under PWI.

answer. Agent 2 is free more often, and the task sharing rule allows him to give substantial assistance to Agent 1. Occasionally, Agent 1 assists Agent 2 with type B calls as well — about 7% of Agent 1's service is used to help Agent 2. □

For our system simulations, let us consider the following five operational routing rules. Compared to the first default rule, the other four pay a higher cost in waiting time in return for a better value of the customer's objective function, $U_c(p_{ik})$, as we will measure in Section 5.4.

1. No priority, or NP. Under this rule, on-the-job learning plays no role in workload assignments. We calculate a staffing level sufficient to maintain a desired average service level with respect to a call volume forecast, and do not consider skills or the call type when routing calls to agents.

2. Priority-No-Idle-Constant, or PNI-Constant. When free, an agent will take his next call from the queue that is highest on his priority list, but all call types appear on his list. So he favors

the call types in a certain order, but will not be idle if a customer is ready to be served.

3. Priority-No-Idle-Swap, or PNI-Swap. This is the same as PNI-Constant, with one difference — the priorities of call types in the routing tables are changed dynamically, such that the agent-type pair with the greatest error between the routing target proportion and the currently calculated proportion occupies the top agent map and type map positions. The pair with the second greatest error occupies the second two positions, and so on. The recalculation takes place following every service event.

4. Priority-With-Idle-N-Constant, or PWI-N-Constant. An agent will idle until a call type from his assigned priority list arrives, even at the cost of making other customers wait. The implementation drops all non-priority call types from the rows of the agent map and the type map. If all types were present, this would be the same as PNI-Fixed. The value N indicates the lengths of the rows on the agent and type maps.

5. Priority-With-Idle-N-Swap, or PWI-N-Swap. This is the same as PWI-Constant, except that the priorities among the agent-type pairs still present in the reduced agent and type maps are changed dynamically, as with PNI-Swap.

5.4. Routing Rule Performance

5.4.1 The Efficient Frontier Defined By Five Routing Rules

Figure 5.8 plots the performance of our five rules in a such a way that they define an *efficient frontier* describing the trade-off between the normalized value of the customer's objective function $U_c(p_{ik})$, as defined on page 100, and the normalized mean waiting time per customer $\bar{W}$. An ideal policy, unachievable in practice, would land on the origin: zero waiting time, and maximum customer utility.

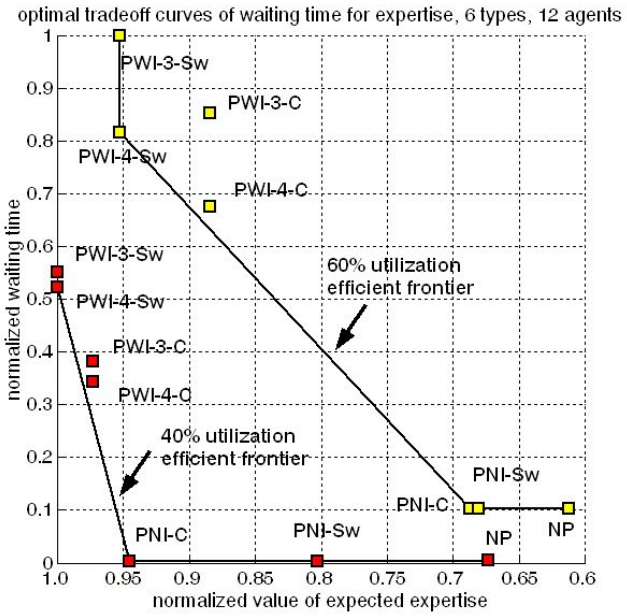


Figure 5.8: **Optimal Trade-Off Curves, Defined by Routing Rules.** Optimal tradeoff curves for 6 call types and 12 agents. Each point marks the performance of one of the types of routing rules. The rules shown define the efficient frontier for average system utilization rates of 40% and 60%. Normalizations are done separately with respect to the values generated by each utilization series.

Each point represents the tradeoff at the final value of a year's operation (240 business days) of a twelve-agent, six-call-type system. All agents begin the year with the same mean service time of seven minutes for every call type; their learning exponents b_{ik} are all set to a moderate value of 0.01, and the system assumes Markovian arrivals and service. The arrival rates for each call type are equal, and are determined by the mean utilization value for the system: 40%, or 60%. Again, for this system the arrival rates among call types are equal, and learning rates among agents are equal.

Under these conditions, we see how the introduction of priorities increases both measures. Rule NP lies at one extreme, providing the lowest waiting time. The forced idling rule PWI-3-Swap lies on the other extreme, providing the largest gain in objective $U_c(p_{ik})$. Both priority-no-idle rules offer good compromises for the 40% utilization case. When utilization increases to 60%, interruptions in agent work patterns caused by prioritizing call types result in longer queues. They also result in more service opportunities for ascending specialized learning curves; PWI-3-Swap at 60% achieves a

higher $U_c(p_{ik})$ gain than PWI-3-Swap at 40%.

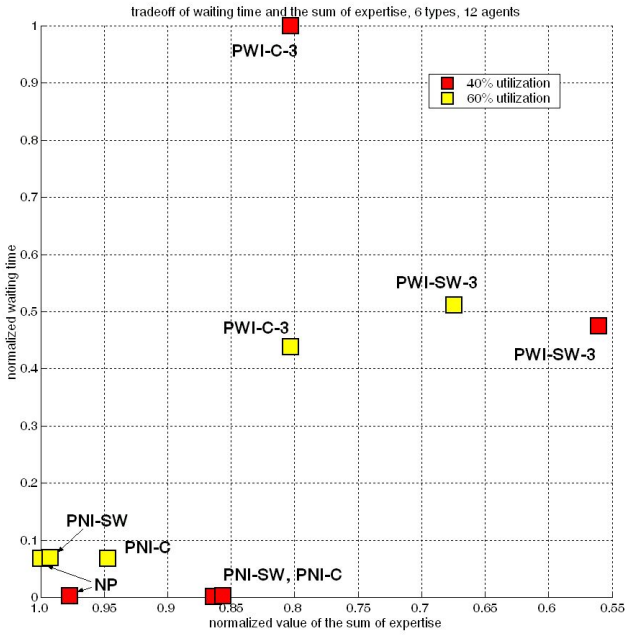


Figure 5.9: **Routing Rules and the the Supervisor's Utility.** Optimal tradeoff curves for 6 call types and 12 agents, with the sum of expertise (the supervisor's utility U_s) on the horizontal axis, and waiting time on the vertical axis. Each point marks the performance of one of the routing rules. Note that the optimal trade-off curves consist of a single point for both the 40% and 60% utilization cases. That point corresponds to routing rule NP, or no-priority.

By contrast, Figure 5.9 shows the equivalent efficient frontier curve for the same system using the supervisor's utility as the objective function, as defined by $U_s(p_{ik})$ on page 100. Note that the curve consists of one point for both 40% and 60% utilization values: rule NP achieves both the lowest total waiting time and the best value of the objective function. This supports the intuition developed in Chapter 3 that the supervisor's utility is optimized by even routing, as rule NP routes calls evenly among agents. It also naturally weights routing proportions according to the arrival rate ratio $\frac{\lambda_k}{\sum_{k=1}^{K} \lambda_k}$, which we also included in the objective $U_s(p_{ik})$. In contrast to rule NP, the priority

$$\begin{bmatrix} 0.0005 & 0.0005 & \boxed{0.02} & 0.0005 & 0.0005 \\ 0.0005 & 0.0005 & 0.0005 & \boxed{0.02} & 0.0005 \\ 0.0005 & 0.0005 & 0.0005 & 0.0005 & \boxed{0.02} \\ 0.0005 & 0.0005 & 0.0005 & 0.0005 & 0.0005 \\ 0.0005 & 0.0005 & 0.0005 & 0.0005 & 0.0005 \\ 0.0005 & 0.0005 & 0.0005 & 0.0005 & 0.0005 \\ 0.0005 & 0.0005 & 0.0005 & 0.0005 & 0.0005 \end{bmatrix}$$

Table 5.2: **Learning Rates for Section 5.4.2.** The set of learning exponent values b_{ik} used for the experiments of Section 5.4.2. Each column represents one of the five call types, and each row represents one of the seven agents. Agents 1, 2, and 3 have high learning rates for call types 3, 4, and 5, respectively, as indicated by the boxes. Note that in the second of the three experiments discussed, the values in the boxes are reduced to the same low learning rates found elsewhere in the table: $b_{ik} = .0005$ for every entry.

routing rules do not appear to be attractive options when our manager chooses the supervisor's utility as her objective.

5.4.2 Routing Rule Performance Given an Asymmetric Distribution of Arrival Rates

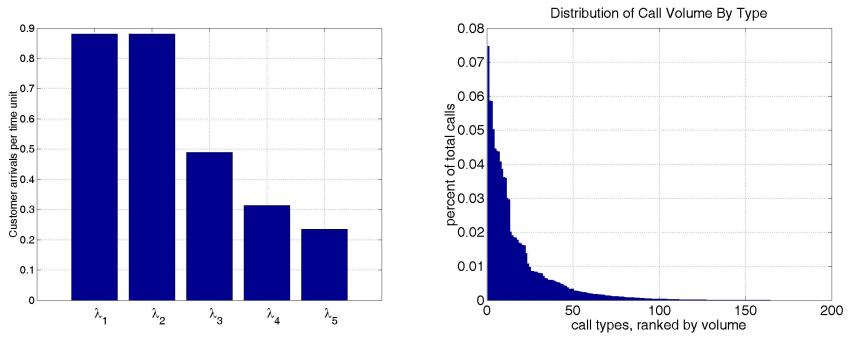


Figure 5.10: **Asymmetric Distribution of Arrival Rates.**Left: relative distribution of arrival rates over five call types, used in Section 5.4.2. Actual arrival rates λ_k for $k = 1...5$ are multiplied by a common factor to obtain a specific system utilization value for each experiment. Right: empirical distribution of call types, from Chapter 4. We may expect different call types to have different arrival rates. Note that routing rules that implement priority routing are most important when learning occurs among the low-volume call types.

Recall our manager's observations on page 87 regarding the role of arrival rates. If on-the-job

learning is important only for the high-volume call types that flood into the center every day, then there is no need for priority rules to distribute those calls. Instead, those high-volume calls will constitute a share of every agent's workload, thus ensuring that everyone gets the cumulative production needed to improve. In such cases we may expect routing rule NP to give the best value for the supervisor's objective $U_s(p_{ik})$, and to give a very attractive point on a efficiency frontier such as Figure 5.8: the lowest waiting time among the rules, and a competitive value of the customer's objective $U_c(p_{ik})$ compared to priority rules.

On the other hand, if learning is important on low-volume call types—and where as in Figure 5.10 traffic is an asymmetric mix of high- and low-volume types—then there is no guarantee that an agent with a talent for improvement on a low-volume type will be exposed to that specific type under a no-priority policy. Under these conditions, priority routing rules deliver better outcomes for the customer's utility $U_c(p_{ik})$ (at the cost of longer customer waiting times) and competitive values for the supervisor's objective $U_s(p_{ik})$. The following three simulation examples illustrate this dynamic.

Consider again the arrival distributions of Figure 5.10. Here, the first two call types λ_1 and λ_2 occur so frequently that all agents must take a share to avoid unreasonable customer waiting. The other three types may be served by one or two agents with acceptably small waiting time increases, allowing for more specialization. We simulate the performance of seven agents serving these five types, with an average system utilization of 40%, given the distribution of learning exponents from Figure 5.2, and for all five distinct routing rules. Each replication reports the outcome for the system at the end of one year's operation. (See Appendix 5.B for more simulation details.)

• **Figure 5.11 represents high learning on the call types with low arrival rates.** This is the scenario outlined above where we expect priority routing rules to do well. Note that among the 35 agent-type pairings, just three demonstrate high talent levels: the first three agents are capable of fast improvement on low-volume call types 3, 4, and 5, respectively. The bar chart on the top shows metrics we desire to maximize: dark green represents the customer's objective $U_c(p_{ik})$; medium green

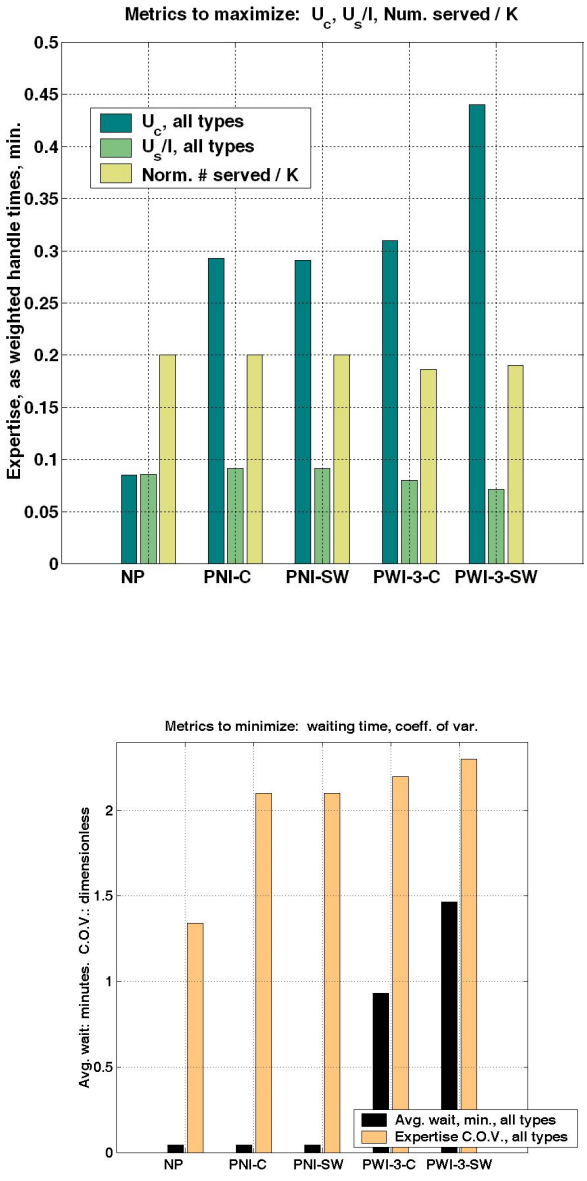
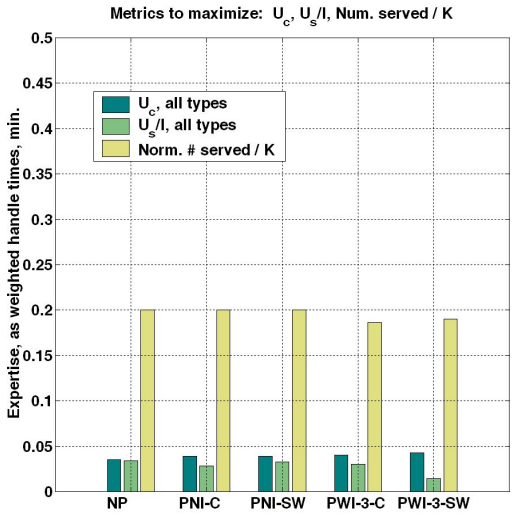


Figure 5.11: **Developing Expertise by Priority Routing.** Results for five routing rules. Here there are 5 call types, 7 agents, and 40% average system utilization with the asymmetric call type arrival distribution of Figure 5.10. Learning improvement exponents b_{jk} are given in Table 5.2. *Top:* metrics we desire to maximize. Dark green bars represent the value of the customer's objective $U_c(p_{ik})$. Medium green bars show improvements in the scaled supervisor's objective $U_s(p_{ik})/I$. *Bottom:* metrics we desire to minimize. Black bars represent the mean waiting time over all customers. Tan bars represent the coefficient of variation of the customer's objective.



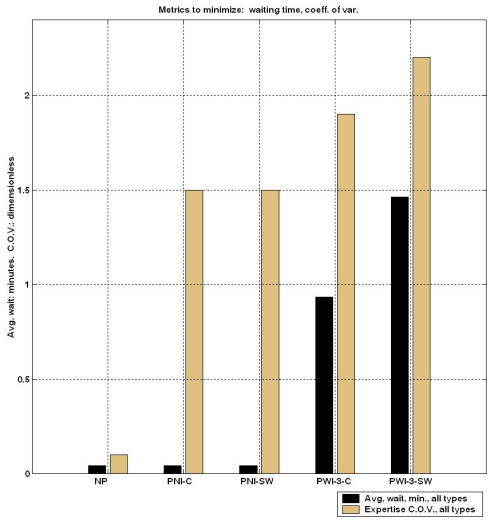


Figure 5.12: **Priority Routing When Learning Rates Are Low.** As Figure 5.11, but low learning: $b_{ik} = 0.0005$ for every agent i and type k. The coefficient of variation calculation for a specific call type includes agents whose routing table prohibits that type; this greatly increases the COV value for the priority-with-idle policies.

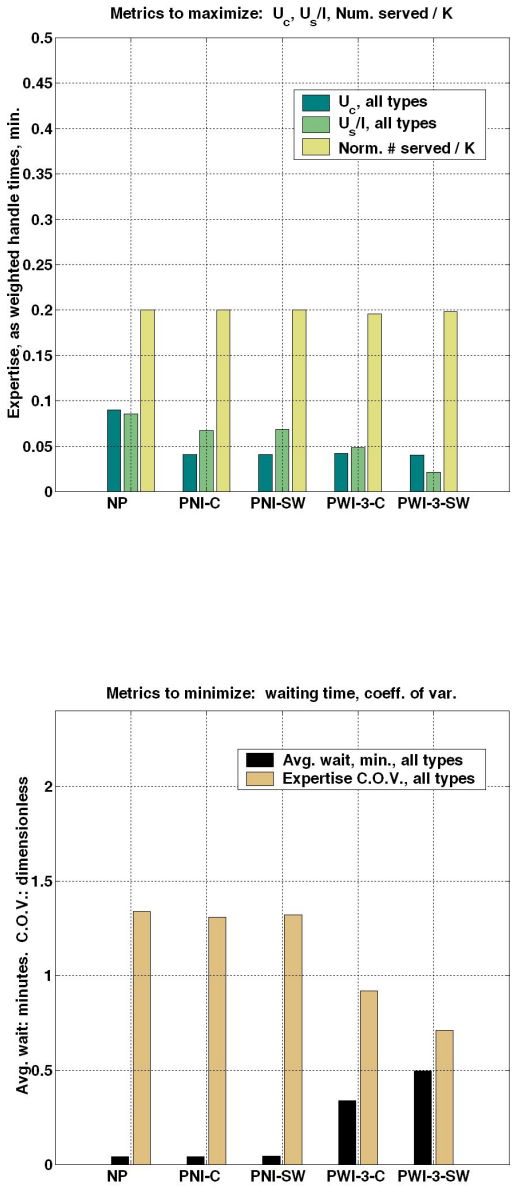


Figure 5.13: **Routing with Incorrect Priorities.** As Figure 5.11, including the same high learning rates. But the highest productivity agent-type pairs are given the lowest priority assignments—so agents are given the wrong priorities from an operational point of view, and priority routing is actually counterproductive on the metrics we wish to maximize.

represents the supervisor's objective $U_s(p_{ik})$ (divided by $I = 7$ agents to put it on the same scale as $U_c(p_{ik})$); and yellow represents the number of calls served by the center during the year, normalized to the number under rule NP (about 233,000 calls).

The plot on the bottom shows metrics we desire to minimize. Black bars represent the mean waiting time under each rule. Tan bars represent the coefficient of variation (COV) of the customer's objective under each rule—the standard deviation divided by the mean. This COV value is first computed separately for each call type, over all agents; and then combined into a single value over all types through a weighted average, as weighted by arrival rate proportions $\frac{\lambda_k}{\sum_{k=1}^{K} \lambda_k}$. Note that the reported COV values are pessimistic, in that agents who took no calls of a certain type are included in the standard deviation; were they to be excluded, the COV values for the priority-with-idle rules would decrease.

Note the clear trade-off shown in the two graphs between increasing the expected value of expertise and customer waiting time—the progression over routing rules of increasing emphasis on priority from NP, to PNI, to PWI increases both $U_c(p_{ik})$ and COV values, and mean waiting time ($\bar{W}$) increases for PWI rules. The value for $U_c(p_{ik})$ is about 5 times higher under PWI-3-Swap than under NP: **this difference illustrates the value of priority routing rules.** Both priority-no-idle policies PNI-C and PNI-Swap also offer improved performance over NP, for almost no cost in increased waiting times. We conclude that if our manager prefers the customer's objective, then the introduction of priority routing here is advantageous; and if the waiting time increase under PWI is too great, PNI is the clear alternative.

• **Figure 5.12** is identical to Figure 5.11 in every respect, except that the three high learning rates highlighted in Table 5.2 have been reduced back to the same negligible levels as all other agent-type pairs: $b_{ik} = 0.0005, \forall\ i,\ k$. Now the routing rules produce about the same value of $U_c(p_{ik})$; and PWI-3-Swap incurs a ten-fold increase in waiting time. There is no useful trade-off here, and NP is preferred.

• **Figure 5.13** repeats the experiment of Figure 5.11, but no priority assignments are made to high productivity agent-type pairs—thus the boxed learning exponents in Table 5.2 are disregarded. We prioritize the extreme opposite of the best choices from an operational perspective: we make all the wrong priority assignments. In this case, policy NP is superior on four of the five metrics, including $U_c(p_{ik})$, the very productivity measure that priority routing exists to enhance. Here priority assignments offer no benefits according to our operational metrics.

This case occurs in practice because other concerns may take priority over management's targets for expertise development. For instance, to increase agent satisfaction and decrease turnover, agents in some centers choose their own priorities. They may select types that seem easier, or more interesting, according to their individual preferences; such choices may not align with priorities that offer optimal productivity increases.

5.4.2.1 Time Series of Expertise and Waiting Time

As an illustration of how the customer's objective $U_c(p_{ik})$ develops as a simulation experiment unfolds, Figure 5.14 provides two time series plots from one replication for quantities of interest in this case. These plots use the same service time (mean 7 minutes) and learning exponents (Table 5.2) as before. This time the PWI rules' routing tables have four entries instead of three.

The top of Figure 5.14 shows the evolution of $U_c(p_{ik})$ in the system, and the bottom shows the mean waiting time for the five policy types. Routing rule PWI-4-Swap appears in the highest position both times—it provides the largest trade-off of waiting time for expected expertise; and policy NP gives the smallest trade-off, as we expect from Figures 5.8 and 5.11.

5.4.3 Performance of Natural Rules

Our results also offer guidance in cases where routing rules and priorities must be chosen without access to a program that sets optimal targets. Consider the following two possibilities:

- **Agent performance characteristics are known, or may be estimated.** Policies PNI-C and

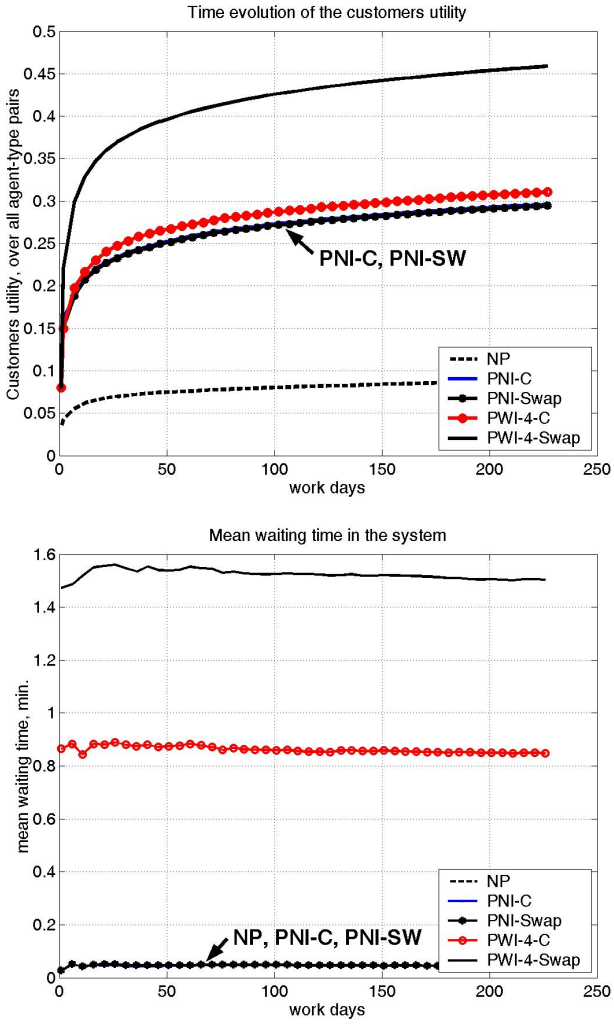


Figure 5.14: **Simulation Time Series.**Top: evolution of the customer's objective $U_c(p_{ik})$ in the system. Priority-With-Idle-Swap performs best. Bottom: cumulative average waiting time in the system.

PWI-C use agent and type routing maps that prioritize agent-type pairings relative to other agent-type pairings. A qualitative assessment of relative performance is sufficient to build these maps. Relative priorities, if accurate, suffice to achieve higher gains in the expected value of expertise.

- **Agent performance characteristics are unknown.** The introduction of priorities in this case may be counterproductive, since as the experiment of Figure 5.13 shows the wrong priorities may be assigned and cause performance to get worse. In our tests we find that a variant of PNI-C may still be viable: *PNI-C with alternating priorities*, or PNI-C-Alt. Here the agent routing map is organized as a Toeplitz matrix, with rotating priority assignments, and the type map is adjusted to match. This routing rule does no worse than NP in boosting $U_c(p_{ik})$, and occasionally does better when an agent's performance characteristics align with her priorities. PNI-C-Alt performed about the same as NP on waiting time and COV metrics, but resulted in a slightly worse value of the supervisor's objective, $U_s(p_{ik})/I$.

5.5. Conclusions from the Optimization-Simulation Studies

We found the log-linear function $H_{ik}(1) \cdot (p_{ik} \cdot D_k)^{-b_{ik}}$ to be convenient for modeling learning effects among contact center agents. Relative to other kinds of learning functions, empirical estimates of the learning exponent are robust, and in addition the function's monomial form makes it suitable for use in the objective and in the constraints of a nonlinear optimization program. In this chapter we showed how to create target routing proportions that take agent expertise into account, and simulated five distinct routing rules to implement the targets in stochastic service systems. Rules that introduce service priorities for specific agent-call type pairs improve the expected value of expertise seen by customers, as long as the assigned priorities accurately reflect the capabilities of the agents. We note that the priorities may actually be set qualitatively by an informed manager, without recourse to an optimization engine; suboptimal but significant expertise gains may still be achieved.

5.A. A Two-Step Nonlinear Program*

Programs CP-U_c and CP-U_s from page 100 are used to generate call routing targets for use in the simulations of Section 5.3. In order to guarantee convergence to a global minimum, the original nonlinear maximum utilization constraint is linearized in these programs—so its convergence properties were obtained at the cost of some accuracy in the upper bound of the call volume allowed for each agent. In some cases, this error may be reduced and the optimization variables p^*_{ik} output by Program CP may be improved according to the following technique.

- Compute p^*_{ik} values normally using Program CP, which is shown again on the left of Table 5.3.
- Use p^*_{ik} as an initial starting solution for Program NLP, shown on the right of Table 5.3. The only difference between Program NLP and Program CP is that the nonlinear maximum utilization constraint has been restored.

The hope here is that the solution p^*_{ik} will guide the solver to the vicinity of Program NLP's true global minimum, bypassing peaks in the objective function that block solvers from escaping local minima traps. Then we guess that iterations of Program NLP from point p^*_{ik} will find this true global minimum. In experiments, we sometimes find that this two-step solver chaining method produces a small improvement in the p^*_{ik} values. Another possibility in cases where other methods fail is to run a metaheuristic simulated annealing, genetic algorithm, or probability collectives solver starting at p^*_{ik} (Ryder and Ross [2005]).

5.B. Simulation Details*

Chapter 5 explores a two-step approach for first defining routing targets, and then for simulating routing rules that implement those targets in a queueing system model of a group of call center agents. In this appendix we discuss some details of interest related to the software environments used to generate these results.

Role	Program CP (Convex)	Program NLP (Not Convex)
Objective	$U_c(p_{ik})$ or $U_s(p_{ik})$, page 100	same
Min. Util.	$\forall$ i $\sum_{k=1}^{K} T_{ik}(p_{ik}) \geq T_{min}$	same
Max. Util.	$\forall$ i $\sum_{k=1}^{K} \beta_{ik} H_{ik}(1) p_{ik} D_k \leq T_{max}$	$\forall$ i $\sum_{k=1}^{K} T_{ik}(p_{ik}) \leq T_{max}$
Full Service	$\forall$ k $\sum_{i=1}^{I} p_{ik} \cdot D_k = D_k$	same
GT Zero	$\forall i, k \;\; p_{ik} \geq 0$	same

Table 5.3: **Convex Program and General Nonlinear Program.** Chaining two optimization programs explore for better results. Program CP sacrifices accuracy in the maximum utilization constraint to guarantee finding a global optimum; Program NLP restores that accuracy, but running by itself from an arbitrary starting solution it may be trapped in a local minimum. By seeding Program NLP with the output p^*_{ik} of Program CP we aim to force Program NLP to start running in the vicinity of the true global minimum for the problem.

The nonlinear optimization programs CP-U_c and CP-U_s from page 100 create the routing targets. These are implemented as Matlab routines that call the constrained nonlinear optimization function *fmincon()*. See Chapter 3 of Coleman and Zhang [2009] for more details.

- Inputs to the nonlinear solver include the optimization time span, agent characteristic matrices, objective and constraint data, and arrival rates and service rates derived from a fixed average utilization rate for all agents.

- Outputs from the solver include agent-to-type and type-to-agent priority routing lists and their associated numerical p_{ik} targets.

Our first prototype routing rule simulations were developed in the feature-rich, user-friendly environment Extend©, from the Imagine That! Corporation; please see *http://www.extendsim.com*.

To be compatible with others in our research community we also developed simulations that consist of Java programs that invoke the ContactCenters simulation library. This open-source library has been developed at the University of Montreal by Professor Pierre L'Ecuyer's contact center research group, with Dr. Eric Buist as the lead designer and programmer. The library contains all the functionality required to run complex discrete-event simulations of contact centers, and is noted for its fast completion times for long runs. Table 5.4 defines the default simulation parameters we use in

Parameter Type	Default Value†
Customer arrivals	Fixed Poisson arrival rate; may differ by call type.
Service time	Exponential service times (an M/M/C system with priorities).
Simulated time span for each experiment	12 months
Working days per month	20 days
Working minutes per day	480 minutes (eight hours)
Working minutes per month	9600 minutes
Mean agent utilization over the entire time span	Fixed for each experiment, at a value between 30% and 60%.
Number of replications for each experiment	30
Sample standard deviation, typical output value	Less than 1% of mean
Sample standard deviation, maximum observed for rare events (such as waiting time values under low utilization), one replication	Less than 11% of mean

Table 5.4: **Chapter 5 Simulation Parameters.**† Unless indicated otherwise in the text.

Chapter 5.

- Inputs to the ContactCenters simulator include all the input parameters for the solver program, the routing target output information from the solver, the choice of routing rule for the current replication—NP, PNI-C, PNI-Swap, PWI-C, or PWI-Swap—and the depth of the priority routing list in the case of priority-with-idle rules.

- Outputs from the simulator include customer waiting times, agent utilization rates, routing proportions observed in simulation $\hat{p}_{ik}$, objective function values $U_c(\hat{p}_{ik})$ and $U_c(\hat{p}_{ik})$, and the coefficients of variation for those objectives.

Note that the queueing system model may be sensitive to the choice of the service time

distribution, and results for other distributions of potential interest have not been pursued here. Gans et al. [2003], page 112 compares the effect of assuming exponential, lognormal, and deterministic service times in an M/G/100 system. Using the exponential distribution (as we do here) gives the longest waiting times. Their model has other parameters we will not describe, but lognormal service times cause waiting times to decrease roughly 10%, and deterministic service times cause waiting times to decrease roughly 20% relative to waiting times under the exponential service time assumption.

Each replication took a year of simulation time, which according to pessimistic assumptions in Equation (53) of Whitt [1989] gives between 5% to 10% of the required simulation time needed to obtain a 95% level of significance in the results ($\pm 2.5\%$ confidence intervals). However, the confidence intervals reported by Contact Centers are better we would expect under this prediction. We believe this is because Contact Centers uses stratified sampling and other variance reduction techniques to reduce uncertainty in the results (L'Ecuyer and Buist [2006]; Ross [2002], pages 131–166). ContactCenters reports its own estimate of its simulation error, and in the worst observed case for a waiting time measurement, the ContactCenters 95% confidence intervals are within 11% of the mean at the end of 240 simulated days (one replication). We then averaged the results of 30 replications with different random seeds to reduce the effect of random simulation errors further.

CHAPTER

SIX

Conclusions, and Recommendations for Future Research

In summary, the principle results of the previous chapters are as follows:

- In Chapter 2, I describe a Markov decision process model shows that the existence of learning will cause changes to the optimal policy for routing work to an agent. I consider parameter ranges of interest for rates of learning, forgetting, arrival and service in the case of one call center agent serving two queues, where learning and forgetting act to change the service rate. The policy *SMax*, also known as the shortest processing time first policy, most closely approximates the optimal policy in the majority of cases. Note that the commonly used μ-C rule is optimal under

the assumption of unchanging mean service rates, but suboptimal when the agent's mean service rate fluctuates.

- In Chapter 3, expertise is quantified as a continuous function of the arrival rate of jobs to an agent. Two key funcions are defined: the customer's utility function U_c, which is the expected value of expertise seen by a customer; and the supervisor's utility function U_s, the sum of all expertise values in the system. As a performance metric, U_c is optimized through specialized work assignments, and U_s is optimized through even or cross-training assignments. In real call centers, management's ability to optimize U_c is limited because server pooling is required to keep service levels high—some call types must be shared among agents, forcing mixed work assignments.

- Chapter 4 describes productivity trends in a new set of empirical call-by-call data from a financial service contact center. Many individual agents demonstrate improvement from on-the-job learning. Specific task types demonstrate significant performance improvement with cumulative production over large groups of agents. The log-linear learning curve function is used due to its robustness in the presence of noisy data. Learning exponents as high as $b_{ik} = 0.1$ are found when agents' work histories are fit by this function.

- In Chapter 5, the research from prior sections informs a new approach to modifying work assignments for contact center agents in order to maximize operational efficiency. A nonlinear program identifies optimal work assignments based on characteristics of each agent–call type pair: these work assignments balance immediate operational performance with the gains realized through learning. Then a contact center simulator identifies the best routing rule to use to achieve these optimal targets in a real system. The routing rules trade-off fidelity to specialized expertise targets with server pooling that reduces customer waiting time; management may use these results to choose the routing rule whose trade-off is best for a specific case. Results include approximate routing rules that do not require the use of nonlinear programming to set targets first.

Interesting extensions of these results may be obtained in three areas. First, new empirical data sets that provide call-by-call details of customer-agent interactions would help to establish how well these learning, service time, and first call resolution parameter assumptions apply to other contact centers. In addition, since operational analyses only consider aspects of service that are quantifiable, contact centers that manage to quantify heretofore elusive aspects of the service encounter have an advantage. An empirical data set that contains additional service quality data—whether or not a call was transferred before terminating; the originating, intermediate, and terminating agent IDs for each unresolved call sequence; whether or not increased call resolution failure is related to management directing the toughest calls to specific agents—would provide a valuable window into the processes analyzed in Chapter 4.

Second, note that the nonlinear programs of Chapter 5 could be adapted to optimize different objective functions in cumulative production. One natural extension is to try to build convex programs that consider the μ-R trends for agents, instead of just optimizing based on service time trends. Then combinations of service time and quality parameters may be used to develop work assignment targets that differ to some degree from the targets set solely using service time, and an appropriate implementation policy can then be chosen from my simulation studies.

Third, the variance in time and quality for calls of the same type handled by the same agent is significant in this data. Advanced studies of how much each side—the customer and the agent—shares responsibility for this variance could reveal how much an agent can learn to reduce variance with more experience. A variance reduction learning curve may then be added to the optimization and simulation framework presented here.

BIBLIOGRAPHY

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

Z. Aksin, F. de Vericourt, and F. Karaesmen. "Call center outsourcing contract analysis and choice.". *Management Science*, 54(2):354–368, February 2008.

A. Avramidis and P. L'Ecuyer. "Modeling and simulation of call centers". In *Proceedings of the 2005 Winter Simulation Conference*, pages 144 – 152, Orlando, FL, December 2005. IEEE Press.

A. Badiru. "Computational survey of univariate and multivariate learning curve models". *IEEE Transactions on Engineering Management*, 39(2):176 – 188, May 1992.

J. Baras, A. Dorsey, and A. Makowski. "Two competing queues with linear costs and geometric service requirements: the mu-c rule is often optimal". *Advances in Applied Probability*, 17:186–209, March 1985.

D.P. Bertsekas. *Convex Analysis and Optimization*. Athena Scientific, PO Box 391, Belmont, MA 02178, 2003.

D.P. Bertsekas. *Dynamic Programming and Optimal Control*, volume II. Athena Scientific, PO Box 391, Belmont, MA 02178, second edition, 1995.

J. Bodreau, W. Hopp, J. McClain, and L. Thomas. "On the interface between operations and human resource management". *Manufacturing and Service Operations Management*, 5(3):179–202, Summer 2003.

S.K. Bordoloi. "Agent recruitment planning in knowledge-intensive call centers". *Journal of Service Research*, 6(4):309 – 323, 2004.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004. ISBN 0 521 83378 7.

L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, and L. Zhao. "Statistical analysis of a telephone call center: a queueing science perspective". Technical Report 03-12, Wharton Financial Institutions Center, November 9, 2002.

F. Caro and J. Gallien. "Dynamic assortment with demand learning for seasonal consumer goods". *Management science*, 53(2):276–292, 2007.

B. Cleveland and J. Mayben. *Call Center Management on Fast Forward: Succeeding in Today's Dynamic Inbound Environment*. Call Center Press, Annapolis, Maryland, first edition, 2000.

T. Coleman and Y. Zhang. *Matlab Optimization Toolbox*. The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760 USA, 2009.

F. de Vericourt and Y. Zhou. "Managing response time in a call-routing problem with service failure". *Operations Research*, 53(6):968 – 981, 2005.

B. Dietrich and T. Harrison. "Serving the services: the emerging science of service management opens opportunities for operations research and management science". OR/MS Today, at http://www.lionhrtpub.com/orms/orms-6-06/frservice.html, June 2006.

G. Eitzen, D. Panton, and G. Mills. "Multi-skilled workforce optimization.". *Annals of Operations Research*, 127(1):359 – 372, 2004.

S. Elaydi. *An Introduction to Difference Equations*. Springer, 233 Spring Street, New York, NY 10013, USA, third edition, 2005.

W. Fischer and K. Meier-Hellstern. "The Markov-modulated Poisson process (MMPP) cookbook". *Performance Evaluation*, 18(2):149–171, September 1993.

C. Froehle. "Service personnel, technology, and their interaction in influencing customer satisfaction". *Decision Sciences*, 37(1):5 – 38, 2006.

N. Gans and Y. Zhou. "Managing learning and turnover in employee staffing". *Operations Research*, 50(6):991–1006, November-December 2002.

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

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

J. George and J. Harrison. "Dynamic control of a queue with adjustable service rate". *Operations Research*, 49(5):720–731, November-December 2002.

P. Ghemawat. "Building strategy on the experience curve". *Harvard Business Review*, pages 143–149, March-April 1985.

W. Grassmann. "The use of eigenvalues for finding equilibrium probabilities of certain Markovian two-dimensional queueing problems". *INFORMS Journal on Computing*, 15(4):412–421, December 2003.

D. Gross and C. Harris. *Fundamentals of Queueing Theory*. Wiley Series in Probability and Statistics. Wiley Interscience, New York, third edition, 1998.

R. Hampshire, M. Harchol-Balter, and W. Massey. "Fluid and diffusion limits for transient sojourn

times of processor sharing queues with time-varying rates". *Queueing Systems*, 53(1-2):19–30, 2006.

J. Harrison. "A priority queue with discounted linear costs". *Operations Research*, 23(2):270–282, March-April 1975.

S. Hasija, E. Pinker, and R. Shumsky. "Staffing and routing in a two-tier call center". *Int. J. Operational Research*, 1(1/2):8 – 29, 2005.

S. Hasija, E. Pinker, and R. Shumsky. "Call center outsourcing contracts under information asymmetry". *Management Science*, 54(4):793 – 807, April 2008.

J. Higgins. *Introduction to Modern Nonparametric Statistics*. Thomson Brooks/Cole, 511 Forest Lodge Road, Pacific Grove, CA 93950, 2004. ISBN 0-534-38775-6.

W. Hopp, S. Iravani, and G. Yuen. "Operations systems with discretionary task completion". *Management Science*, 53(1):61–77, January 2007.

S. Iravani, B. Kolfal, and M. Oyen. "Call-center labor cross-training: it's a small world after all". *Management Science*, 53(7):1102 – 1112, July 2007.

G. Koole. "Assigning a single server to inhomogeneous queues with switching costs". *Theoretical Computer Science*, 182(1-2):203–216, August 1997.

D. Lay. *Linear Algebra and Its Applications*. Addison-Wesley, New York, first edition, 1994.

P. L'Ecuyer and E. Buist. "Variance Reduction in the Simulation of Call Centers". In *Proceedings of the 2006 Winter Simulation Conference*, pages 604 – 613, Monterey, CA, Dec. 3-6 2006.

A. Mandelbaum and A. Stolyar. "Scheduling flexible servers with convex delay costs: heavy traffic optimality of the generalized c-mu rule". *Operations Research*, 52(6):836–855, November-December 2004.

W. Martinez and A. Martinez. *Computational Statistics Handbook with MATLAB*. Chapman and Hall/CRC, 6000 Broken Sound Parkway NW, Suite 300, Boca Raton, FL 33487, 2008. ISBN 978-1-58488-566-5.

J. Mazzola and K. McCardle. "The stochastic learning curve: optimal production in the presence of learning-curve uncertainty". *Operations research*, 45(3):440–450, May-June 1997.

V. Mehrotra and J. Fama. "Call center simulation modeling: methods, challenges, and opportunities". In *Proceedings of the 2003 Winter Simulation Conference*, volume 1, pages 135 – 143, New Orleans, Louisiana, December 2003. IEEE Press.

K. Meier-Hellstern. "A fitting algorithm for Markov-modulated Poisson processes having two arrival rates". *European Journal of Operational Research*, 29(3):370–377, 1987.

S. Misra, E. Pinker, and R. Shumsky. "Salesforce design with experience-based learning". *IIE Transactions*, 36(10):941–952, 2004.

S.G. Nash and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill Series in Industrial Engineering and Management Science. McGraw-Hill Inc., New York, first edition, 1996.

D. Nembhard and N. Osothsilp. "An empirical comparison of forgetting models". *IEEE Transactions on Engineering Management*, 48(3):283 – 291, 2001.

R. Nunez-Queija. "A queueing model with varying service rate for ABR". In *Proceedings of the Tenth International Conference on Computer Performance Evaluation: Modelling Techniques and Tools*, pages 93–104. Springer-Verlag, 1998.

A. Parasuraman, V. Zeithaml, and L. Berry. "SERVQUAL: A multiple-item scale for measuring consumer perceptions of service quality". *Journal of Retailing*, 64(1):12 – 40, 1988.

E. Pinker and R. Shumsky. "The efficiency-quality trade-off of cross-trained workers". *Manufacturing and Service Operations Management*, 2(1):32–48, Winter 2000.

D.A. Reis. ”Learning curves in food services”. *Journal of the Operational Research Society*, 42(8): 623–629, 1991.

Z. Ren and Y. Zhou. ”Call center outsourcing: coordinating staffing level and service quality”. *Management Science*, 54(2):369 – 383, February 2008.

S. Ross. *Simulation*. Academic Press, Elsevier, San Diego, third edition, 2002.

M. Rossiter. ”A switched Poisson model for data traffic”. *Australian Telecommunication Research*, 21 (1):53–57, 1987.

G. Ryder and K. Ross. ”A Probability Collectives Approach to Weighted Clustering Algorithms for Ad hoc Networks”. In *IASTED CCN*, pages 94 – 99, Marina Del Rey, CA, USA, Oct. 24 - 26 2005.

G. Ryder, K. Ross, and J. Musacchio. ”Optimal service policies under learning effects”. *Int. J. Services and Operations Management*, 4(6):631–651, 2008.

V. Ryokov and E. Lembert. ”Optimal dynamic priorities in single-line queueing systems”. *Engineering Cybernetics*, 5(1):21–30, 1967.

S. Sayin and S. Karabati. ”Assigning cross-trained workers to departments: a two-stage optimization model to maximize utility and skill improvement”. *European Journal of Operational Research*, 176 (3):1643 – 1658, February 2007.

M. Schilling, P. Vidal, R. Ployhart, and A. Marangoni. ”Learning by doing something else: variation, relatedness, and the learning curve”. *Management Science*, 49(1):39 – 56, January 2003.

L. Schrage and L. Miller. ”The queue M/G/1 with the shortest remaining processing time discipline”. *Operations Research*, 14(4):670–684, July-August 1966.

S. Shafer, D. Nembhard, and M. Uzumeri. ”The effects of worker learning, forgetting, and heterogeneity on assembly line productivity”. *Management Science*, 47(12):1639–1653, December 2001.

A. Shumsky and E. Pinker. ”Gatekeepers and referrals in services”. *Management Science*, 49(7):839 – 856, July 2003.

S. Sikstrom and M. Jaber. ”The power integration diffusion model for production breaks”. *Journal of Experimental Psychology: Applied*, 8(2):118 – 126, 2002.

D. Stirzaker. *Elementary Probability*. Cambridge University Press, New York, second edition, 2003.

T. Tezcan. *”State Space Collapse in Many-Server Diffusion Limits of Parallel Server Systems and Applications”*. PhD thesis, Georgia Institute of Technology, August 2006.

P. Thompson. ”How much did the Liberty shipbuilders forget?”. *Management Science*, 53(6):908–918, June 2007.

A. Tucker, I. Nembhard, and A. Edmondson. ”Implementing new practices: an empirical study of organizational learning in hospital intensive care units”. *Management Science*, 53(6):894–907, June 2007.

R. Wallace and W. Whitt. ”A staffing algorithm for call centers with skill-based routing”. *Manufacturing and Service Operations Management*, 7(5):276–294, Fall 2005.

W. Whitt. ”The impact of increased employee retention on performance in a customer contact center”. *Manufacturing and Service Operations Management*, 8(3):235–252, 2006.

W. Whitt. ”Planning Queueing Simulations”. *Management Science*, 35(11):1341–1366, November 1989.

J. Zamiska, M. Jaber, and H. Kher. ”Worker deployment in dual resource constrained systems with a task type factor”. *European Journal of Operational Research*, 177(3):1507–1519, March 2007.

E. Zohar, A. Mandelbaum, and N. Shimkin. ”Adaptive behavior of impatient customers in tele-queues: theory and empirical support”. *Management Science*, 48(4):566–583, April 2002.